

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Video surveillance systems for use in security applications –
Part 2-3: Video transmission protocols – IP interoperability implementation
based on Web services**

**Systèmes de vidéosurveillance destinés à être utilisés dans les applications
de sécurité –
Partie 2-3: Protocoles de transmission vidéo – Mise en œuvre de
l'interopérabilité IP en fonction des services Web**



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2013 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester.

If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de la CEI ou du Comité national de la CEI du pays du demandeur.

Si vous avez des questions sur le copyright de la CEI ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de la CEI de votre pays de résidence.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

Useful links:

IEC publications search - www.iec.ch/searchpub

The advanced search enables you to find IEC publications by a variety of criteria (reference number, text, technical committee,...).

It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available on-line and also once a month by email.

Electropedia - www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary (IEV) on-line.

Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

A propos de la CEI

La Commission Electrotechnique Internationale (CEI) est la première organisation mondiale qui élabore et publie des Normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications CEI

Le contenu technique des publications de la CEI est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente. un corrigendum ou amendement peut avoir été publié.

Liens utiles:

Recherche de publications CEI - www.iec.ch/searchpub

La recherche avancée vous permet de trouver des publications CEI en utilisant différents critères (numéro de référence, texte, comité d'études,...).

Elle donne aussi des informations sur les projets et les publications remplacées ou retirées.

Just Published CEI - webstore.iec.ch/justpublished

Restez informé sur les nouvelles publications de la CEI. Just Published détaille les nouvelles publications parues. Disponible en ligne et aussi une fois par mois par email.

Electropedia - www.electropedia.org

Le premier dictionnaire en ligne au monde de termes électroniques et électriques. Il contient plus de 30 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans les langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (VEI) en ligne.

Service Clients - webstore.iec.ch/csc

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: csc@iec.ch.

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Video surveillance systems for use in security applications –
Part 2-3: Video transmission protocols – IP interoperability implementation
based on Web services**

**Systèmes de vidéosurveillance destinés à être utilisés dans les applications
de sécurité –
Partie 2-3: Protocoles de transmission vidéo – Mise en œuvre de
l'interopérabilité IP en fonction des services Web**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

PRICE CODE
CODE PRIX

XH

ICS 13.320

ISBN 978-2-8322-1189-2

**Warning! Make sure that you obtained this publication from an authorized distributor.
Attention! Veuillez vous assurer que vous avez obtenu cette publication via un distributeur agréé.**

CONTENTS

FOREWORD.....	21
INTRODUCTION.....	23
1 Scope.....	24
2 Normative references	26
3 Terms, definitions and abbreviations	29
3.1 Terms and definitions	29
3.2 Abbreviations	32
4 Overview	34
4.1 Web services.....	34
4.2 IP configuration	35
4.3 Device discovery	35
4.4 Device types	36
4.5 Device management.....	36
4.5.1 Capabilities	36
4.5.2 Network	37
4.5.3 System	37
4.5.4 Retrieval of system information.....	37
4.5.5 Firmware upgrade	38
4.5.6 System restore	38
4.5.7 Security	38
4.6 DeviceIO	38
4.7 Imaging configuration.....	39
4.8 Media configuration.....	39
4.8.1 General	39
4.8.2 Media profiles	39
4.9 Real-time streaming	43
4.10 Event handling	44
4.11 PTZ control	44
4.12 Video analytics	45
4.13 Analytics device	47
4.14 Display	47
4.15 Receiver.....	47
4.15.1 General	47
4.15.2 Synchronization points	48
4.16 Storage	48
4.16.1 Storage model	48
4.16.2 Recording.....	49
4.16.3 Search.....	50
4.16.4 Replay	50
4.17 Security.....	50
5 Web Services framework	51
5.1 Services overview	51
5.1.1 General	51
5.1.2 Services requirements	52
5.2 WSDL overview.....	52
5.3 Namespaces	53

5.4	Types	55
5.5	Messages	55
5.6	Operations	56
5.6.1	One-way operation type	57
5.6.2	Request-response operation type	58
5.7	Port types	58
5.8	Binding	58
5.9	Ports	58
5.10	Services	58
5.11	Error handling	58
5.11.1	Protocol errors	59
5.11.2	SOAP errors	59
5.12	Security	62
5.12.1	User-based access control	62
5.12.2	User token profile	63
6	IP configuration	64
7	Device discovery	65
7.1	General	65
7.2	Modes of operation	65
7.3	Discovery definitions	66
7.3.1	Endpoint reference	66
7.3.2	Hello	66
7.3.3	Probe and probe match	68
7.3.4	Resolve and resolve match	68
7.3.5	Bye	68
7.3.6	SOAP fault messages	69
7.4	Remote discovery extensions	69
7.4.1	Network scenarios	69
7.4.2	Discover proxy (DP)	71
7.4.3	Remote Hello and Probe behaviour	72
7.4.4	Client behaviour	73
7.4.5	Security	73
8	Device management	74
8.1	Capabilities	74
8.1.1	Get WSDL URL	74
8.1.2	Capability exchange	75
8.2	Network	78
8.2.1	Get hostname	78
8.2.2	Set hostname	78
8.2.3	Get DNS settings	79
8.2.4	Set DNS settings	80
8.2.5	Get NTP settings	81
8.2.6	Set NTP settings	82
8.2.7	Get dynamic DNS settings	83
8.2.8	Set dynamic DNS settings	84
8.2.9	Get network interface configuration	85
8.2.10	Set network interface configuration	85
8.2.11	Get network protocols	87
8.2.12	Set network protocols	88

8.2.13	Get default gateway.....	88
8.2.14	Set default gateway.....	89
8.2.15	Get zero configuration.....	90
8.2.16	Set zero configuration.....	90
8.2.17	Get IP address filter.....	91
8.2.18	Set IP address filter.....	92
8.2.19	Add an IP filter address.....	92
8.2.20	Remove an IP filter address.....	93
8.2.21	IEEE 802.11 configuration.....	94
8.3	System.....	99
8.3.1	Device Information.....	99
8.3.2	Get System URIs.....	100
8.3.3	Backup.....	101
8.3.4	Restore.....	101
8.3.5	Start system restore.....	102
8.3.6	Get system date and time.....	103
8.3.7	Set system date and time.....	104
8.3.8	Factory default.....	105
8.3.9	Firmware upgrade.....	106
8.3.10	Start firmware upgrade.....	107
8.3.11	Get system logs.....	108
8.3.12	Get support information.....	109
8.3.13	Reboot.....	110
8.3.14	Get scope parameters.....	110
8.3.15	Set scope parameters.....	111
8.3.16	Add scope parameters.....	112
8.3.17	Remove scope parameters.....	113
8.3.18	Get discovery mode.....	113
8.3.19	Set discovery mode.....	114
8.3.20	Get remote discovery mode.....	114
8.3.21	Set remote discovery mode.....	115
8.3.22	Get remote DP addresses.....	115
8.3.23	Set remote DP addresses.....	116
8.4	Security.....	116
8.4.1	Get access policy.....	116
8.4.2	Set access policy.....	117
8.4.3	Get users.....	117
8.4.4	Create users.....	118
8.4.5	Delete users.....	119
8.4.6	Set users settings.....	120
8.4.7	IEEE 802.1X configuration.....	121
8.4.8	Create self-signed certificate.....	126
8.4.9	Get certificates.....	127
8.4.10	Get CA certificates.....	128
8.4.11	Get certificate status.....	128
8.4.12	Set certificate status.....	129
8.4.13	Get certificate request.....	129
8.4.14	Get client certificate status.....	130
8.4.15	Set client certificate status.....	131

8.4.16	Load device certificate	131
8.4.17	Load device certificates in conjunction with its private key	132
8.4.18	Get certificate information request	133
8.4.19	Load CA certificates	134
8.4.20	Delete certificate	135
8.4.21	Get remote user	136
8.4.22	Set remote user	137
8.4.23	Get endpoint reference	138
8.5	Input/Output (I/O)	138
8.5.1	Get relay outputs	139
8.5.2	Set relay output settings	139
8.5.3	Trigger relay output	140
8.5.4	Auxiliary operation	141
8.6	Service specific fault codes	142
9	Device IO Service	148
9.1	VideoOutputs	148
9.1.1	General	148
9.1.2	GetVideoOutputs	148
9.2	VideoOutputConfiguration	149
9.2.1	GetVideoOutputConfiguration	149
9.2.2	SetVideoOutputConfiguration	149
9.2.3	GetVideoOutputConfigurationOptions	150
9.3	VideoSources	151
9.3.1	General	151
9.3.2	GetVideoSources	151
9.4	VideoSourceConfiguration	152
9.4.1	GetVideoSourceConfiguration	152
9.4.2	SetVideoSourceConfiguration	153
9.4.3	GetVideoSourceConfigurationOptions	153
9.5	AudioOutputs	154
9.5.1	General	154
9.5.2	GetAudioOutputs	154
9.6	AudioOutputConfiguration	155
9.6.1	GetAudioOutputConfiguration	155
9.6.2	SetAudioOutputConfiguration	156
9.6.3	GetAudioOutputConfigurationOptions	156
9.7	AudioSources	157
9.7.1	General	157
9.7.2	GetAudioSources	157
9.8	AudioSourceConfiguration	158
9.8.1	GetAudioSourceConfiguration	158
9.8.2	SetAudioSourceConfiguration	159
9.8.3	GetAudioSourceConfigurationOptions	159
9.9	Relay outputs	160
9.9.1	Get relay outputs	160
9.9.2	Set relay output settings	161
9.9.3	Trigger relay output	162
9.10	Service specific fault codes	163
10	Imaging configuration	164

10.1	Imaging settings	164
10.1.1	Get imaging settings	165
10.1.2	Set imaging settings	166
10.1.3	Get options	167
10.1.4	Move	168
10.1.5	Get move options	169
10.1.6	Stop	170
10.1.7	Get imaging status	171
10.2	Service specific fault codes	172
11	Media configuration	173
11.1	Audio and video codecs	173
11.2	Media profile	173
11.2.1	Create media profile	174
11.2.2	Get media profiles	175
11.2.3	Get media profile	176
11.2.4	Add video source configuration to a profile	177
11.2.5	Add video encoder configuration to a profile	177
11.2.6	Add audio source configuration to a profile	178
11.2.7	Add audio encoder configuration to a profile	179
11.2.8	Add PTZ configuration to a profile	180
11.2.9	Add video analytics configuration to a profile	181
11.2.10	Add metadata configuration to a profile	182
11.2.11	Add audio output configuration	183
11.2.12	Add audio decoder configuration	184
11.2.13	Remove video source configuration from a profile	185
11.2.14	Remove video encoder configuration from a profile	186
11.2.15	Remove audio source configuration from a profile	187
11.2.16	Remove audio encoder configuration from a profile	188
11.2.17	Remove PTZ configuration from a profile	189
11.2.18	Remove video analytics configuration from a profile	190
11.2.19	Remove metadata configuration from a profile	191
11.2.20	Remove audio output configuration	192
11.2.21	Remove audio decoder configuration	193
11.2.22	Delete media profile	194
11.3	Video source	195
11.3.1	General	195
11.3.2	GetVideoSources	195
11.4	Video source configuration	196
11.4.1	Get video source configurations	196
11.4.2	Get video source configuration	197
11.4.3	Get compatible video source configurations	197
11.4.4	Get video source configuration options	198
11.4.5	Modify a video source configuration	199
11.5	Video encoder configuration	200
11.5.1	Get video encoder configurations	201
11.5.2	Get video encoder configuration	201
11.5.3	Get compatible video encoder configurations	202
11.5.4	Get video encoder configuration options	203
11.5.5	Modify a video encoder configuration	204

11.5.6	Get guaranteed number of video encoder instances	205
11.6	Audio source	206
11.6.1	General	206
11.6.2	Get audio sources	206
11.7	Audio source configuration	207
11.7.1	Get audio source configurations	207
11.7.2	Get audio source configuration	208
11.7.3	Get compatible audio source configurations	209
11.7.4	Get audio source configuration options	210
11.7.5	Modify an audio source configuration	211
11.8	Audio encoder configuration	212
11.8.1	Get audio encoder configurations	213
11.8.2	Get audio encoder configuration	213
11.8.3	Get compatible audio encoder configurations	214
11.8.4	Get audio encoder configuration options	215
11.8.5	Modify audio encoder configurations	217
11.9	Video analytics configuration	217
11.9.1	Get video analytics configurations	218
11.9.2	Get video analytics configuration	218
11.9.3	Get compatible video analytics configurations	219
11.9.4	Modify a video analytics configuration	220
11.10	Metadata configuration	222
11.10.1	Get metadata configurations	222
11.10.2	Get metadata configuration	223
11.10.3	Get compatible metadata configurations	223
11.10.4	Get metadata configuration options	224
11.10.5	Modify a metadata configuration	225
11.11	Audio outputs	226
11.11.1	General	226
11.11.2	Get audio outputs	226
11.12	Audio output configuration	227
11.12.1	Get audio output configurations	227
11.12.2	Get audio output configuration	228
11.12.3	Get compatible audio output configurations	229
11.12.4	Get audio output configuration options	229
11.12.5	Modify audio output configuration	231
11.13	Audio decoder configuration	231
11.13.1	Get audio decoder configurations	232
11.13.2	Get audio decoder configuration	232
11.13.3	Get compatible audio decoder configurations	233
11.13.4	Get audio decoder configuration options	234
11.13.5	Modify audio decoder configuration	235
11.14	Audio channel modes	236
11.15	Stream URI	237
11.15.1	General	237
11.15.2	Request stream URI	237
11.16	Snapshot	239
11.16.1	General	239
11.16.2	Request snapshot URI	239

11.17 Multicast.....	239
11.17.1 Start multicast streaming	240
11.17.2 Stop multicast streaming	240
11.18 Synchronization points	241
11.18.1 General	241
11.18.2 Set synchronization point.....	241
11.19 Service specific fault codes	242
12 Real time streaming.....	243
12.1 Media stream protocol	244
12.1.1 Transport format	244
12.1.2 Media transport	244
12.1.3 Synchronization point	249
12.1.4 JPEG over RTP	249
12.2 Media control protocol	252
12.2.1 Stream control	252
12.3 Back channel connection.....	257
12.3.1 RTSP Require – Tag	257
12.3.2 Connection setup for a bi- directional connection.....	258
12.3.3 Multicast streaming.....	260
12.4 Error handling	260
13 Receiver configuration.....	260
13.1 Persistence	260
13.2 Receiver modes	260
13.3 Receiver commands.....	261
13.3.1 Get receivers.....	261
13.3.2 Get receiver.....	261
13.3.3 Create receiver.....	262
13.3.4 Delete receiver.....	262
13.3.5 Configure receiver	263
13.3.6 SetReceiverMode	263
13.3.7 GetReceiverState	264
13.4 Events.....	264
13.4.1 ChangeState	264
13.4.2 Connection Failed.....	265
13.5 Service specific fault codes	265
14 Display service	265
14.1 Panes.....	266
14.1.1 GetPaneConfigurations.....	266
14.1.2 GetPaneConfiguration	267
14.1.3 SetPaneConfigurations	268
14.1.4 SetPaneConfiguration.....	269
14.1.5 CreatePaneConfiguration	270
14.1.6 DeletePaneConfiguration	271
14.2 Layout	272
14.2.1 GetLayout.....	272
14.2.2 SetLayout.....	273
14.3 Display options.....	274
14.3.1 General	274
14.3.2 LayoutOptions	274

14.3.3	Coding Capabilities.....	274
14.3.4	GetDisplayOptions.....	274
14.4	Events.....	275
14.4.1	General	275
14.4.2	Decoding error event	275
14.5	Service specific fault codes	276
15	Event handling	277
15.1	Basic notification interface.....	277
15.1.1	General	277
15.1.2	Requirements	278
15.2	Real-time Pull-Point Notification Interface	279
15.2.1	Create pull point subscription	280
15.2.2	Pull messages	280
15.3	Notification streaming interface	281
15.4	Properties	281
15.4.1	Property example	282
15.5	Notification structure	282
15.5.1	Notification information.....	283
15.5.2	Message format.....	284
15.5.3	Property example, continued.....	285
15.5.4	Message description language.....	286
15.5.5	Message content filter.....	287
15.6	Synchronization point.....	288
15.7	Topic structure	289
15.7.1	ONVIF topic namespace.....	289
15.7.2	Topic type information.....	290
15.7.3	Topic filter	290
15.8	Get event properties.....	292
15.9	SOAP fault messages.....	293
15.10	Notification example.....	293
15.10.1	GetEventPropertiesRequest	293
15.10.2	GetEventPropertiesResponse.....	293
15.10.3	CreatePullPointSubscription	294
15.10.4	CreatePullPointSubscriptionResponse.....	295
15.10.5	PullMessagesRequest	295
15.10.6	PullMessagesResponse.....	296
15.10.7	UnsubscribeRequest	297
15.11	Service specific fault codes	297
16	PTZ control.....	297
16.1	PTZ Model.....	298
16.2	PTZ Node.....	299
16.2.1	GetNodes	300
16.2.2	GetNode.....	300
16.3	PTZ configuration	301
16.3.1	GetConfigurations.....	302
16.3.2	GetConfiguration	303
16.3.3	GetConfigurationOptions	303
16.3.4	SetConfiguration.....	304
16.4	Move operations.....	305

16.4.1	AbsoluteMove.....	306
16.4.2	RelativeMove.....	307
16.4.3	ContinuousMove.....	309
16.4.4	Stop	312
16.4.5	GetStatus	312
16.5	Preset operations	313
16.5.1	SetPreset	313
16.5.2	GetPresets	315
16.5.3	GotoPreset.....	316
16.5.4	RemovePreset.....	317
16.6	Home position operations	317
16.6.1	GotoHomePosition.....	317
16.6.2	SetHomePosition	318
16.7	Auxiliary operations.....	319
16.7.1	General	319
16.7.2	SendAuxiliaryCommand.....	319
16.8	Predefined PTZ spaces	320
16.8.1	Absolute position spaces	320
16.8.2	Relative translation spaces.....	321
16.8.3	Continuous velocity spaces	323
16.8.4	Speed spaces.....	324
16.9	Service specific fault codes	325
17	Video analytics	327
17.1	Scene description interface.....	327
17.1.1	Overview	327
17.1.2	Frame related content.....	327
17.1.3	Scene elements.....	330
17.2	Rule interface.....	334
17.2.1	Rule representation.....	335
17.2.2	Rule description language	335
17.2.3	Standard rules.....	336
17.2.4	Operations on rules	338
17.3	Analytics modules interface.....	342
17.3.1	Analytics module configuration	342
17.3.2	Analytics module description language	343
17.3.3	Operations on analytics modules	343
17.4	Service-specific fault codes	347
18	Analytics device	349
18.1	Overview	349
18.2	Analytics engine input	349
18.2.1	GetAnalyticsEngineInputs.....	350
18.2.2	GetAnalyticsEngineInput.....	350
18.2.3	SetAnalyticsEngineInput.....	351
18.2.4	CreateAnalyticsEngineInputs	352
18.2.5	DeleteAnalyticsEngineInputs	353
18.3	Video analytics configuration.....	354
18.3.1	GetVideoAnalyticsConfiguration	354
18.3.2	SetVideoAnalyticsConfiguration.....	355
18.4	Analytics engines	356

18.4.1	GetAnalyticsEngines	356
18.4.2	GetAnalyticsEngine	357
18.5	Analytics engine control	358
18.5.1	GetAnalyticsEngineControls	358
18.5.2	GetAnalyticsEngineControl	359
18.5.3	SetAnalyticsEngineControl	359
18.5.4	CreateAnalyticsEngineControl	360
18.5.5	DeleteAnalyticsEngineControl	362
18.6	GetAnalyticsState	362
18.7	Output streaming configuration	363
18.7.1	General	363
18.7.2	Request stream URI	363
19	Recording control	364
19.1	General	364
19.2	General requirements	366
19.3	Data structures	366
19.3.1	RecordingConfiguration	366
19.3.2	TrackConfiguration	366
19.3.3	RecordingJobConfiguration	366
19.4	CreateRecording	368
19.5	DeleteRecording	369
19.6	GetRecordings	369
19.7	SetRecordingConfiguration	371
19.8	GetRecordingConfiguration	371
19.9	CreateTrack	372
19.10	DeleteTrack	374
19.11	GetTrackConfiguration	374
19.12	SetTrackConfiguration	375
19.13	CreateRecordingJob	376
19.14	DeleteRecordingJob	377
19.15	GetRecordingJobs	378
19.16	SetRecordingJobConfiguration	378
19.17	GetRecordingJobConfiguration	379
19.18	SetRecordingJobMode	380
19.19	GetRecordingJobState	381
19.20	Events	382
19.20.1	Recording job state changes	382
19.20.2	Configuration changes	383
19.20.3	Data deletion	383
19.20.4	Recording and track creation and deletion	383
19.21	Examples	384
19.21.1	Example 1: Setup recording of a single camera	384
19.21.2	Example 2: Record multiple streams from one camera to a single recording	385
20	Recording search	386
20.1	General	386
20.2	Concepts	387
20.2.1	Search direction	387
20.2.2	Recording event	387

20.2.3	Search session	387
20.2.4	Search scope	388
20.2.5	Search filters	388
20.3	Data structures	388
20.3.1	RecordingInformation structure	388
20.3.2	RecordingSourceInformation structure	388
20.3.3	TrackInformation structure	389
20.3.4	SearchState enumeration	389
20.3.5	MediaAttributes structure	389
20.3.6	FindEventResult structure	389
20.3.7	FindPTZPositionResult structure	390
20.3.8	PTZPositionFilter structure	390
20.3.9	MetadataFilter structure	390
20.3.10	FindMetadataResult structure	390
20.4	GetRecordingSummary	390
20.5	GetRecordingInformation	391
20.6	GetMediaAttributes	391
20.7	FindRecordings	392
20.8	GetRecordingSearchResults	393
20.9	FindEvents	394
20.10	GetEventSearchResults	395
20.11	FindPTZPosition	396
20.12	GetPTZPositionSearchResults	397
20.13	FindMetadata	398
20.14	GetMetadataSearchResults	399
20.15	GetSearchState	400
20.16	EndSearch	401
20.17	Recording Event Descriptions	402
20.18	XPath dialect	404
21	Replay control	405
21.1	Use of RTSP	405
21.2	RTP header extension	406
21.2.1	NTP timestamps	406
21.2.2	Compatibility with the JPEG header extension	407
21.3	RTSP feature tag	407
21.4	Initiating Playback	407
21.4.1	Range header field	408
21.4.2	Rate-Control header field	408
21.4.3	Frames header field	409
21.4.4	Synchronization points	409
21.5	Reverse replay	410
21.5.1	Packet transmission order	410
21.5.2	RTP sequence numbers	410
21.5.3	RTP timestamps	410
21.6	RTSP keepalive	411
21.7	Currently recording footage	411
21.8	End of footage	411
21.9	Go to time	411
21.10	Use of RTCP	411

21.11 Replay service commands	412
21.11.1 Request replay URI	412
21.11.2 ReplayConfiguration	413
21.11.3 SetReplayConfiguration	413
21.11.4 GetReplayConfiguration	413
21.11.5 Service specific fault codes	414
22 Security	414
22.1 Transport level security	414
22.1.1 Supported cipher suites	415
22.1.2 Server authentication	415
22.1.3 Client authentication	415
22.2 Message level security	416
22.3 IEEE 802.1X	416
Annex A (informative) Notification topics	417
Annex B (informative) Scene descriptions	420
Annex C (normative) Video IP network interface XML schemata	422
Bibliography	649
Figure 1 – Web Services based development principles	35
Figure 2 – A media profile	40
Figure 3 – Complete profile configuration	41
Figure 4 – Layer structure	43
Figure 5 – PTZ control model	45
Figure 6 – Video analytics architecture	46
Figure 7 – Storage model with tracks	49
Figure 8 – A device, for example a NVT, in an administrative domain (private) and the client (NVC) in a public network	70
Figure 9 – A device, for example a NVT, in public network and the client (NVC) in an administrative domain (private)	70
Figure 10 – A device, for example a NVT, in an administrative domain (private) and the client (NVC) in another administrative domain (private)	70
Figure 11 – Both a device, for example a NVT, and the client (NVC) in a public network	71
Figure 12 – Remote discovery message exchange pattern between a device (for example a NVT) and a Home DP	72
Figure 13 – Message sequence for clients (NVC) pre-configured with home DP address	73
Figure 14 – RTP header	245
Figure 15 – RTCP sequence	247
Figure 16 – RTCP Sender Report	248
Figure 17 – Media synchronization	249
Figure 18 – RTP/JPEG packet structure (only the typical content is listed for the extension payload)	250
Figure 19 – Stream control	253
Figure 20 – Keep Alive	255
Figure 21 – Sequence diagram for the base notification interface	278
Figure 22 – Sequence diagram for the Real-time Pull-Point Notification Interface	279

Figure 23 – Default frame coordinate system	329
Figure 24 – Example of recordings and tracks	365
Figure 25 – Diagram of the RecordingJobConfiguration elements	367
Figure 26 – Diagram of the RecordingJobStateInformation elements	381
Table 1 – Service requirements for the device types	52
Table 2 – Defined namespaces in this standard	53
Table 3 – Referenced namespaces (with prefix).....	54
Table 4 – Referenced namespaces (without prefix).....	55
Table 5 – Operation description outline used in this standard	57
Table 6 – Generic faults.....	61
Table 7 – HTTP errors	62
Table 8 – Scope parameters	67
Table 9 – Get WSDL URL command	74
Table 10 – Get Capabilities command.....	75
Table 11 – The capabilities in the GetCapabilities command.....	76
Table 12 – GetHostname command	78
Table 13 – SetHostname command.....	79
Table 14 – GetDNS command.....	80
Table 15 – Set DNS command.....	81
Table 16 – GetNTP command.....	82
Table 17 – SetNTP command	83
Table 18 – GetDynamicDNS command.....	84
Table 19 – SetDynamicDNS command.....	85
Table 20 – GetNetworkInterfaces command.....	85
Table 21 – SetNetworkInterfaces command	86
Table 22 – GetNetworkProtocols command.....	87
Table 23 – SetNetworkProtocols command	88
Table 24 – GetNetworkDefaultGateway command.....	89
Table 25 – SetNetworkDefaultGateway command	89
Table 26 – GetZeroConfiguration command	90
Table 27 – SetZeroConfiguration command	91
Table 28 – GetIPAddressFilter command	91
Table 29 – SetIPAddressFilter command	92
Table 30 – AddIPAddressFilter command	93
Table 31 – RemoveIPAddressFilter command.....	94
Table 32 – GetDot11Capabilities.....	97
Table 33 – IEEE 802.11 capabilities.....	97
Table 34 – GetDot11Status.....	98
Table 35 – ScanAvailable802.11Networks	99
Table 36 – GetDeviceInformation command.....	100
Table 37 – GetSystemUri command	101
Table 38 – GetSystemBackup command	101

Table 39 – RestoreSystem command	102
Table 40 – StartSystemRestore command	103
Table 41 – GetSystemDateAndTime command	104
Table 42 – SetSystemDateAndTime command	105
Table 43 – SetSystemFactoryDefault command	106
Table 44 – UpgradeSystemFirmware command	107
Table 45 – StartFirmwareUpgrade command	108
Table 46 – GetSystemLog command	109
Table 47 – GetSystemSupportInformation command	110
Table 48 – SystemReboot command	110
Table 49 – GetScopes command	111
Table 50 – SetScopes command	112
Table 51 – AddScopes command	112
Table 52 – RemoveScopes command	113
Table 53 – GetDiscoveryMode command	114
Table 54 – SetDiscoveryMode command	114
Table 55 – GetRemoteDiscoveryMode command	115
Table 56 – SetRemoteDiscoveryMode command	115
Table 57 – GetDPAddresses command	116
Table 58 – SetDPAddresses command	116
Table 59 – GetAccessPolicy command	117
Table 60 – SetAccessPolicy command	117
Table 61 – GetUsers command	118
Table 62 – CreateUsers command	119
Table 63 – DeleteUsers command	120
Table 64 – SetUser command	121
Table 65 – CreateDot1XConfiguration command	123
Table 66 – SetDot1XConfigurationRequest command	124
Table 67 – GetDot1XConfiguration command	125
Table 68 – GetDot1XConfigurations command	125
Table 69 – DeleteDot1XConfigurations command	126
Table 70 – CreateCertificate command	127
Table 71 – GetCertificates command	128
Table 72 – GetCACertificates command	128
Table 73 – GetCertificatesStatus command	129
Table 74 – SetCertificatesStatus command	129
Table 75 – GetPkcs10Request command	130
Table 76 – GetClientCertificateMode command	131
Table 77 – SetClientCertificateMode command	131
Table 78 – LoadCertificates command	132
Table 79 – LoadCertificateWithPrivateKey command	133
Table 80 – GetCertificateInformation command	134
Table 81 – LoadCACertificates command	135

Table 82 – DeleteCertificates command	136
Table 83 – GetRemoteUser command	137
Table 84 – SetRemoteUser command	138
Table 85 – GetEndpointReference command	138
Table 86 – GetRelayOutputs command	139
Table 87 – SetRelayOutputSettings command	140
Table 88 – SetRelayOutputState command	141
Table 89 – Send auxiliary command	142
Table 90 – Device service specific fault codes	143
Table 91 – GetVideoOutputs command	149
Table 92 – GetVideoOutputConfiguration command	149
Table 93 – SetVideoOutputConfiguration command	150
Table 94 – GetVideoOutputConfigurationOptions command	151
Table 95 – GetVideoSources command	152
Table 96 – GetVideoSourceConfiguration command	152
Table 97 – SetVideoSourceConfiguration command	153
Table 98 – GetVideoSourceConfiguartionOptions command	154
Table 99 – GetAudioOutputs command	155
Table 100 – GetAudioOutputConfiguration command	155
Table 101 – SetAudioOutputConfiguration command	156
Table 102 – GetAudioOutputConfigurationOptions command	157
Table 103 – GetAudioSources command	158
Table 104 – GetAudioSourceConfiguration command	158
Table 105 – SetAudioSourceConfiguration command	159
Table 106 – GetAudioSourceConfigurationOptions command	160
Table 107 – GetRelayOutputs command	161
Table 108 – SetRelayOutputSettings command	162
Table 109 – SetRelayOutputState command	163
Table 110 – DeviceIO service specific fault codes	164
Table 111 – GetImagingSettings command	166
Table 112 – SetImagingSettings command	167
Table 113 – GetOptions command	168
Table 114 – Move (focus) command	169
Table 115 – GetMoveOptions (focus) command	170
Table 116 – Stop (focus) command	171
Table 117 – GetStatus (focus) command	172
Table 118 – Imaging specific fault codes	172
Table 119 – CreateProfile command	175
Table 120 – GetProfiles command	176
Table 121 – GetProfile command	176
Table 122 – AddVideoSourceConfiguration command	177
Table 123 – AddVideoEncoderConfiguration command	178
Table 124 – AddAudioSourceConfiguration command	179

Table 125 – AddAudioEncoderConfiguration command	180
Table 126 – AddPTZConfiguration command	181
Table 127 – AddVideoAnalytics command.....	182
Table 128 – AddMetadataConfiguration command	183
Table 129 – AddAudioOutputConfiguration	184
Table 130 – AddAudioDecoderConfiguration.....	185
Table 131 – RemoveVideoSourceConfiguration command	186
Table 132 – RemoveVideoEncoderConfiguration command	187
Table 133 – RemoveAudioSourceConfiguration command	188
Table 134 – RemoveAudioEncoderConfiguration command	189
Table 135 – RemovePTZConfiguration command.....	190
Table 136 – RemoveVideoAnalyticsConfiguration command	191
Table 137 – RemoveMetadataConfiguration command.....	192
Table 138 – RemoveAudioOutputConfiguration.....	193
Table 139 – RemoveAudioDecoderConfiguration	194
Table 140 – DeleteProfile command	195
Table 141 – GetVideoSources command	196
Table 142 – GetVideoSourceConfigurations command.....	196
Table 143 – GetVideoSourceConfiguration command	197
Table 144 – GetCompatibleVideoSourceConfigurations command	198
Table 145 – GetVideoSourceConfigurationOptions command	199
Table 146 – SetVideoSourceConfiguration command.....	200
Table 147 – GetVideoEncoderConfigurations command.....	201
Table 148 – GetVideoEncoderConfiguration command.....	202
Table 149 – GetCompatibleVideoEncoderConfigurations command	203
Table 150 – GetVideoEncoderConfigurationOptions command.....	204
Table 151 – SetVideoEncoderConfiguration command.....	205
Table 152 – GetGuaranteedNumberOfVideoEncoderInstances command	206
Table 153 – GetAudioSources command	207
Table 154 – GetAudioSourceConfigurations command.....	208
Table 155 – GetAudioSourceConfiguration command	209
Table 156 – GetCompatibleAudioSourceConfigurations command	210
Table 157 – GetAudioSourceConfigurationOptions command	211
Table 158 – SetAudioSourceConfiguration command.....	212
Table 159 – GetAudioEncoderConfigurations command.....	213
Table 160 – GetAudioEncoderConfiguration command.....	214
Table 161 – GetCompatibleAudioEncoderConfigurations command	215
Table 162 – GetAudioEncoderConfigurationOptions command.....	216
Table 163 – SetAudioEncoderConfiguration command	217
Table 164 – GetVideoAnalyticsConfigurations command.....	218
Table 165 – GetVideoAnalyticsConfiguration command	219
Table 166 – GetCompatibleVideoAnalyticsConfigurations command	220
Table 167 – SetVideoAnalyticsConfiguration command.....	221

Table 168 – GetMetadataConfigurations command	222
Table 169 – GetMetadataConfiguration command	223
Table 170 – GetCompatibleMetadataConfigurations command	224
Table 171 – GetMetadataConfigurationOptions command	225
Table 172 – etMetadataConfiguration command	226
Table 173 – GetAudioOutputs	227
Table 174 – GetAudioOutputConfiguration	228
Table 175 – GetAudioOutputConfiguration	228
Table 176 – GetCompatibleAudioOutputConfiguration	229
Table 177 – GetAudioOutputConfigurationOptions	230
Table 178 – SetAudioOutputConfiguration	231
Table 179 – GetAudioDecoderConfigurations	232
Table 180 – GetAudioDecoderConfiguration	233
Table 181 – GetCompatibleAudioDecoderConfigurations	234
Table 182 – GetAudioDecoderConfigurationOptions	235
Table 183 – SetAudioDecoderConfiguration	236
Table 184 – GetStreamUri command	238
Table 185 – GetSnapshotUri command	239
Table 186 – StartMulticastStreaming command	240
Table 187 – StopMulticastStreaming command	241
Table 188 – SetSynchronizationPoint command	242
Table 189 – Media service specific fault codes	243
Table 190 – RTP header value	245
Table 191 – RTSP methods	254
Table 192 – GetReceivers command	261
Table 193 – GetReceiver command	261
Table 194 – CreateReceiver command	262
Table 195 – DeleteReceiver command	263
Table 196 – ConfigureReceiver command	263
Table 197 – SetReceiverMode command	264
Table 198 – GetReceiverState command	264
Table 199 – Service specific fault codes	265
Table 200 – GetPaneConfigurations	267
Table 201 – GetPaneConfiguration	268
Table 202 – SetPaneConfigurations	269
Table 203 – SetPaneConfiguration	270
Table 204 – CreatePaneConfiguration	271
Table 205 – DeletePaneConfiguration	272
Table 206 – GetLayout	273
Table 207 – SetLayout	273
Table 208 – GetDisplayOptions	275
Table 209 – Service specific fault codes	276
Table 210 – CreatePullPointSubscription command	280

Table 211 – PullMessages command	281
Table 212 – SetSynchronizationPoint command	289
Table 213 – GetEventProperties command	292
Table 214 – GetNodes command	300
Table 215 – GetNode command	301
Table 216 – GetConfigurations command	302
Table 217 – GetConfiguration command	303
Table 218 – GetConfigurationOptions command	304
Table 219 – SetConfiguration command	305
Table 220 – AbsoluteMove command	307
Table 221 – RelativeMove command	309
Table 222 – ContinuousMove command	311
Table 223 – Stop (PTZ) command	312
Table 224 – GetStatus (PTZ) command	313
Table 225 – SetPreset command	314
Table 226 – GetPresets command	315
Table 227 – GotoPreset command	316
Table 228 – RemovePreset command	317
Table 229 – GotoHomePosition command	318
Table 230 – SetHomePosition command	319
Table 231 – Send Auxiliary command	320
Table 232 – PTZspecific fault codes	325
Table 233 – GetSupportedRules command	338
Table 234 – GetRules command	339
Table 235 – CreateRules command	340
Table 236 – ModifyRules command	341
Table 237 – DeleteRules command	342
Table 238 – GetSupportedAnalyticsModules command	344
Table 239 – GetAnalyticsModules command	344
Table 240 – CreateAnalyticsModules command	345
Table 241 – ModifyAnalyticsModules command	346
Table 242 – DeleteAnalyticsModules command	347
Table 243 – The analytics-specific fault codes	348
Table 244 – GetAnalyticsEngineInputs command	350
Table 245 – GetAnalyticsEngineInput command	351
Table 246 – SetAnalyticsEngineInput command	352
Table 247 – CreateAnalyticsEngineInputs command	353
Table 248 – DeleteAnalyticsEngineInputs command	354
Table 249 – GetVideoAnalyticsConfiguration command	355
Table 250 – SetVideoAnalyticsConfiguration command	356
Table 251 – GetAnalyticsEngines command	357
Table 252 – GetAnalyticsEngine command	357
Table 253 – GetAnalyticsEngineControls command	358

Table 254 – GetAnalyticsEngineControl command	359
Table 255 – SetAnalyticsEngineControl command	360
Table 256 – CreateAnalyticsEngineControl command	361
Table 257 – DeleteAnalyticsEngineControl command	362
Table 258 – GetAnalyticsState	363
Table 259 – GetAnalyticsDeviceStreamUri command	364
Table 260 – CreateRecording command	368
Table 261 – DeleteRecording command	369
Table 262 – GetRecordings command	370
Table 263 – SetRecordingConfiguration command	371
Table 264 – GetRecordingConfiguration command	372
Table 265 – CreateTrack command	373
Table 266 – DeleteTrack command	374
Table 267 – GetTrackConfiguration command	375
Table 268 – SetTrackConfiguration command	376
Table 269 – CreateRecordingJob command	377
Table 270 – DeleteRecordingJob command	378
Table 271 – GetRecordingJobs command	378
Table 272 – SetRecordingJobConfiguration command	379
Table 273 – GetRecordingJobConfiguration command	380
Table 274 – SetRecordingJobMode command	380
Table 275 – GetRecordingJobState command	381
Table 276 – GetRecordingSummary command	391
Table 277 – GetRecordingInformation command	391
Table 278 – GetMediaAttributes command	392
Table 279 – FindRecordings command	393
Table 280 – GetRecordingSearchResults command	394
Table 281 – FindEvents command	395
Table 282 – GetEventSearchResults command	396
Table 283 – FindPTZPosition command	397
Table 284 – GetPTZPositionSearchResults command	398
Table 285 – FindMetadata command	399
Table 286 – GetMetadataSearchResults command	400
Table 287 – GetSearchState command	401
Table 288 – EndSearch command	401
Table 289 – RTP packet layout	406
Table 290 – RTP packet with JPEG header layout	407
Table 291 – GetReplayUri command	412
Table 292 – SetReplayConfiguration command	413
Table 293 – GetReplayConfiguration command	413
Table 294 – Replay service specific fault codes	414

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**VIDEO SURVEILLANCE SYSTEMS FOR USE
IN SECURITY APPLICATIONS –****Part 2-3: Video transmission protocols –
IP interoperability implementation based on Web services**

FOREWORD

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as “IEC Publication(s)”). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62676-2-3 has been prepared by IEC technical committee 79: Alarm and electronic security systems.

The text of this standard is based on the following documents:

FDIS	Report on voting
79/437/FDIS	79/450/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts in the IEC 62676 series, published under the general title *Video surveillance systems for use in security applications*, can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

The IEC Technical Committee 79 in charge of alarm and electronic security systems together with many governmental organisations, test houses and equipment manufacturers have defined a common framework for video surveillance transmission in order to achieve interoperability between products.

The IEC 62676 series of standards on video surveillance system is divided into 4 independent parts:

- Part 1 System requirements
- Part 2: Video transmission protocols
- Part 3: Analog and digital video interfaces
- Part 4 : Application guidelines (to be published)

Each part has its own clauses on scope, references, definitions and requirements.

This IEC 62676-2 series consists of 3 subparts, numbered parts 2-1, 2-2 and 2-3 respectively:

IEC 62676-2-1, *Video transmission protocols – General requirements*

IEC 62676-2-2, *Video transmission protocols – IP interoperability implementation based on HTTP and REST services*

IEC 62676-2-3, *Video transmission protocols – IP interoperability implementation based on Web services*

This third subpart of IEC 62676-2 covers IP Interoperability Implementation Based Web Services. It is based on the requirements for IP video transmission protocols covered in IEC 62676-2-1, which defines protocol requirements to be fulfilled by any high-level IP video device interface.

VIDEO SURVEILLANCE SYSTEMS FOR USE IN SECURITY APPLICATIONS –

Part 2-3: Video transmission protocols – IP interoperability implementation based on Web services

1 Scope

This part 2-3 of IEC 62676 defines procedures for communication between network video clients and video transmitter devices based on Web Services. This new set of specifications makes it possible to build network video systems with devices and receivers from different manufacturers using common and well defined interfaces. These interfaces cover functions such as device management, real-time streaming of audio and video, event handling, Pan, Tilt and Zoom (PTZ) control, video analytics as well as control, search and replay of recordings.

The management and control interfaces defined in this standard are described as Web services. This international standard also contains full XML schema and Web Service Description Language (WSDL) definitions for the introduced network video services.

In order to offer full plug-and-play interoperability, the standard defines procedures for device discovery. The device discovery mechanisms in the standard are based on the WS-Discovery specification with extensions. These extensions have been introduced in order to cover the specific network video discovery needs.

This standard is not limited to discovery, configuration and control functions, but defines precise formats for media and metadata streaming in IP networks using suitable profiling of IETF standards. Furthermore, appropriate protocol extensions have been introduced in order to make it possible for network video manufacturers to offer a fully standardized network video transfer solution to its customers and integrators.

A video transmission device supporting compliance to the requirements of this standard with the help of Web services according to the specification of this part is declared as compatible to IEC 62676-2 Web service Interoperability.

The goal of this standard is to realize a fully interoperable network video implementation comprised of products from different network video vendors. This standard describes the network video model, interfaces, data types and data exchange patterns. The standard reuses existing relevant standards where available, and introduces new specifications only where necessary to support the specific requirements for network video surveillance. This is the Open Network Video Interface Forum (ONVIF) core specification. In addition, ONVIF has released the following related specifications:

- ONVIF Schema [see C.15]
- ONVIF Analytics Service WSDL [see C.1]
- ONVIF Analytics Device Service [see C.2]
- ONVIF Device Service WSDL [see C.4]
- ONVIF DeviceIO Service WSDL [see C.3]
- ONVIF Display Service WSDL [see C.5]
- ONVIF Event Service WSDL [see C.6]
- ONVIF Imaging Service WSDL [see C.7]
- ONVIF Media Service WSDL [see C.8]

- ONVIF PTZ Service WSDL [see C.9]
- ONVIF Receiver Service WSDL [see C.10]
- ONVIF Recording Service WSDL [see C.11]
- ONVIF Remote Discovery WSDL [see C.12]
- ONVIF Replay Service WSDL [see C.13]
- ONVIF Search Service WSDL [see C.14]
- ONVIF Topic Namespace XML [see C.16]

The purpose of this standard is to define the ONVIF specification framework, and is divided into the following sections:

Specification overview: Gives an overview of the different specification parts and how they are related to each other.

Web Services Framework: Offers a brief introduction to Web Services and the Web Services basis for the ONVIF specifications.

IP configuration: Defines the ONVIF network video IP configuration requirements.

Device discovery: Describes how devices are discovered in local and remote networks.

Device management: Defines the network video transmitter management commands.

DeviceIO: Defines commands to handle physical inputs and outputs.

Display: Defines commands to deal with display devices.

Imaging and media: Defines the configuration commands related to imaging and media settings.

Real time streaming: Provides requirements for interoperable video, audio and metadata streaming.

Event handling: Defines how to subscribe to and receive data from network video events (notifications).

PTZ control: Provides commands for pan, tilt and zoom control.

Video analytics: Defines the ONVIF analytics model, analytics object description and analytics rules configurations.

Video analytics device: Defines commands to deal with a video analytics device.

Recording control: Defines mechanism for the configuring of recordings.

Recording search and replay control: Provides commands for retrieval of recorded media including metadata.

Security section: Defines the transport and message level security requirements on ONVIF compliant implementations.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ITU-T G.711, *Pulse code modulation (PCM) of voice frequencies*

< http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.711-198811-I!!PDF-E&type=items >

[X.680] ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*

[X.681] ITU-T Recommendation X.681 (1997) | ISO/IEC 8824-2:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*

[X.682] ITU-T Recommendation X.682 (1997) | ISO/IEC 8824-3:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification*

[X.683] ITU-T Recommendation X.683 (1997) | ISO/IEC 8824-4:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*

[X.690] ITU-T Recommendation X.690 (1997) | ISO/IEC 8825-1:2008, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*

NIST FIPS 180-2, *SECURE HASH STANDARD*

<<http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>>

RFC1305, *Network Time Protocol (Version 3), Specification, Implementation and Analysis*

<<http://www.ietf.org/rfc/rfc1305.txt>>

IETF RFC 2131, *Dynamic Host Configuration Protocol*

<<http://www.ietf.org/rfc/rfc2131.txt>>

IETF RFC 2136, *Dynamic Updates in the Domain Name System (DNS UPDATE)*

<<http://www.ietf.org/rfc/rfc2136.txt>>

IETF RFC 2246, *The TLS Protocol Version 1.0*

<<http://www.ietf.org/rfc/rfc2246.txt>>

IETF RFC 2326, *Real Time Streaming Protocol (RTSP)*

<<http://www.ietf.org/rfc/rfc2326.txt>>

IETF RFC 2435, *RTP Payload Format for JPEG-compressed Video*

<<http://www.ietf.org/rfc/rfc2435.txt>>

IETF RFC 2616, *Hypertext Transfer Protocol – HTTP/1.1*

[<http://www.ietf.org/rfc/rfc2616.txt>](http://www.ietf.org/rfc/rfc2616.txt)

IETF RFC 2617, *HTTP Authentication: Basic and Digest Access Authentication*

[<http://www.ietf.org/rfc/rfc2617.txt>](http://www.ietf.org/rfc/rfc2617.txt)

IETF RFC 2782, *A DNS RR for specifying the location of services (DNS SRV)*

[<http://www.ietf.org/rfc/rfc2782.txt>](http://www.ietf.org/rfc/rfc2782.txt)

IETF RFC 3268, *Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)*

[<http://www.ietf.org/rfc/rfc3268.txt>](http://www.ietf.org/rfc/rfc3268.txt)

IETF RFC 3315, *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*

[<http://www.ietf.org/rfc/rfc3315.txt>](http://www.ietf.org/rfc/rfc3315.txt)

IETF RFC 3550, *RTP: A Transport Protocol for Real-Time Applications*

[<http://www.ietf.org/rfc/rfc3550.txt>](http://www.ietf.org/rfc/rfc3550.txt)

IETF RFC 3551, *RTP Profile for Audio and Video Conferences with Minimal Control*

[<http://www.ietf.org/rfc/rfc3551.txt>](http://www.ietf.org/rfc/rfc3551.txt)

IETF RFC 3927, *Dynamic Configuration of IPv4 Link-Local Addresses*

[<http://www.ietf.org/rfc/rfc3927.txt>](http://www.ietf.org/rfc/rfc3927.txt)

IETF RFC 3984, *RTP Payload Format for H.264 Video*

[<http://www.ietf.org/rfc/rfc3984.txt>](http://www.ietf.org/rfc/rfc3984.txt)

IETF RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*

[<http://www.ietf.org/rfc/rfc3986.txt>](http://www.ietf.org/rfc/rfc3986.txt)

IETF RFC 4514, *Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names*

[<http://www.ietf.org/rfc/rfc4514.txt>](http://www.ietf.org/rfc/rfc4514.txt)

IETF RFC 4566, *SDP: Session Description Protocol*

[<http://www.ietf.org/rfc/rfc4566.txt>](http://www.ietf.org/rfc/rfc4566.txt)

IETF RFC 4571, *Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport*

[<http://www.ietf.org/rfc/rfc4571.txt>](http://www.ietf.org/rfc/rfc4571.txt)

IETF RFC 4702, *The Dynamic Host Configuration Protocol (DHCP) Client Fully Qualified Domain Name (FQDN) Option*

<<http://www.ietf.org/rfc/rfc4702.txt>>

IETF RFC 4861, *Neighbor Discovery for IP version 6 (IPv6)*

<<http://www.ietf.org/rfc/rfc4861.txt>>

IETF RFC 4862, *IPv6 Stateless Address Autoconfiguration*

<<http://www.ietf.org/rfc/rfc4862.txt>>

W3C SOAP 1.2, Part 1, *Messaging Framework*

<<http://www.w3.org/TR/soap12-part1/>>

W3C SOAP Version 1.2 Part 2: Adjuncts (Second Edition)

<<http://www.w3.org/TR/2007/REC-soap12-part2-20070427/>>

OASIS Web Services Base Notification 1.3

<http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf>

OASIS Web Services Security UsernameToken Profile 1.0

<<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf>>

W3C XML Schema Part 1: Structures Second Edition

<<http://www.w3.org/TR/xmlschema-1/>>

W3C XML Schema Part 2: Datatypes Second Edition

<<http://www.w3.org/TR/xmlschema-2/>>

W3C XML-binary Optimized Packaging

<<http://www.w3.org/TR/2005/REC-xop10-20050125/>>

W3C XML Path Language (XPath) Version 1.0

<<http://www.w3.org/TR/xpath/>>

IEEE 802.11:2007, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*

<<http://standards.ieee.org/getieee802/download/802.11-2007.pdf>>

IEEE 802.1X, Port-Based Network Access Control

[<http://standards.ieee.org/getieee802/download/802.1X-2004.pdf>](http://standards.ieee.org/getieee802/download/802.1X-2004.pdf)

[UDDI API ver2] “*UDDI Version 2.04 API Specification UDDI Committee Specification, 19 July 2002*”, OASIS standard, 19 July 2002

[<http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf>](http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf)

[UDDI Data Structure ver2] “*UDDI Version 2.03 Data Structure Reference UDDI Committee Specification*”, OASIS standard, 19 July 2002.

[<http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf>](http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf)

[WS-X.509Token] “*Web Services Security X.509 Certificate Token Profile 1.1*”, OASIS Standard, 1 February 2006.

[<http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf>](http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf)

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1.1

ad-hoc network

an independent basic service set

[SOURCE: IEEE 802.11:2007]

3.1.2

basic service set

a set of IEEE 802.11 stations that have successfully joined in a common network

[SOURCE: IEEE 802.11:2007]

3.1.3

capability

ability of a device that allows a client to ask for its services

3.1.4

configuration entity

a network video device media abstract component that is used to produce a media stream on the network

Note 1 to entry: The media stream is a video and/or an audio stream.

3.1.5

control plane

plane consisting of media control functions

Note 1 to entry: Media control functions are device control, media configuration and PTZ commands.

3.1.6

digital PTZ

function that diminishes or crops an image to adjust the image position and ratio

3.1.7
input/output
IO

external connectors

EXAMPLES: Digital inputs, relay ports and video/audio inputs/outputs.

3.1.8
layout

arrangement of display areas (panes) on a monitor

3.1.9
media plane

plane consisting of media stream

Note 1 to entry: The media stream is video, audio and metadata.

3.1.10
media profile

set of configurations defining a subset of audio/video source, encoder and decoder settings including PTZ and analytics parameters

3.1.11
metadata

set of all streaming data except video and audio, including video analytics results, PTZ position data and other metadata

Note 1 to entry: The other metadata include textual data from POS applications.

3.1.12
network video transmitter

network video server that sends media data over an IP network to a client

Note 1 to entry: A network video server is an IP network camera or an encoder device.

3.1.13
network video display

network video receiver that receives media data over an IP network from e.g. an NVT

EXAMPE: An IP network video monitor.

3.1.14
network video storage

a device that records media and metadata received from a streaming device, such as an NVT, over an IP network to a permanent storage medium

Note 1 to entry: The NVS also enables clients to review the stored data.

3.1.15
network video analytics

a device that performs analysis on data received from a streaming device, such as an NVT, or a storage device, such as an NVS

3.1.16
pane

area of the physical display

3.1.17
public Key cryptography standards

group of public key cryptography standards devised and published by RSA Security

3.1.18**pre shared key**

a static key that is distributed to the device

3.1.19**PTZ node**

low-level PTZ entity that maps to the PTZ device and its capabilities

3.1.20**pullpoint**

resource for pulling messages

Note 1 to entry: By pulling messages, notifications are not blocked by firewalls.

3.1.21**recording**

currently stored media (if any) and metadata on the NVS from a single data source

Note 1 to entry: A recording comprises one or more tracks. A recording can have more than one track of the same type e.g. two different video tracks recorded in parallel with different settings.

3.1.22**recording event**

an event associated with a recording, represented by a notification message in the APIs

3.1.23**recording job**

a job performs the transfer of data from a data source to a particular recording using a particular configuration

3.1.24**remote discovery proxy****remote DP**

the remote DP allows a NVT to register at the remote DP and at the NVC to find registered NVTs through the remote DP even if the NVC and NVT resides in different administrative network domains

3.1.25**scene description**

metadata output by video analytics describing object location and behaviour

3.1.26**track**

an individual data channel consisting of video, audio, or metadata

3.1.27**video analytics**

algorithms or programs used to analyze video data and to generate data describing object location and behaviour

3.1.28**Wi-Fi protected access**

a certification program created by the Wi-Fi Alliance to indicate compliance with the security protocol covered by the program

3.2 Abbreviations

AAC	Advanced Audio Coding
API	Application Programming Interface
ASCII	7 Bit Character Encoding
ASN	Abstract Syntax Notation
AVP	Audio/Video Profile
AVPF	Audio/Video Profile for rtcp Feedback
BLC	Back Light Compensation
BSSID	Basic Service Set Identifier
CA	Certificate Authority
CBC	Cipher-Block Chaining
CCMP	Counter mode with Cipher-block chaining Message authentication code Protocol
DER	Distinguished Encoding Rules
DHCP	Dynamic Host Configuration Protocol
DHT	Define Huffman Table
DM	Device Management
DNS	Domain Name Server
DQT	Define Quantization Table
DP	Discovery Proxy
DRI	Define Restart Interval
EAP	Extensible Authentication Protocol
EOI	End Of Image
FIPS	Federal Information Processing Standard
FOV	Field Of View
GW	Gateway
HMAC	Hash-based Message Authentication Code
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol over Secure Socket Layer
IO, I/O	Input/Output
IP	Internet Protocol
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
Ir	Infrared
JFIF	JPEG File Interchange Format
JPEG	Joint Photographic Expert Group
LAN	Local Area Network
MPEG-4	Moving Picture Experts Group - 4
MTOM	Message Transmission Optimization Mechanism
NAT	Network Address Translation
NFC	Near Field Communication
NTP	Network Time Protocol
NVA	Network Video Analytics Device

NVC	Network Video Client
NVD	Network Video Display
NVT	Network Video Transmitter
NVS	Network Video Storage Device
OASIS	Organization for the Advancement of Structured Information Standards
ONVIF	Open Network Video Interface Forum
PLI	Picture Loss Indication
POSIX	Portable Operating System Interface
PKCS	Public Key Cryptography Standards
PSK	Pre Shared Key
PTZ	Pan/Tilt/Zoom
QVGA	Quarter Video Graphics Array (320 x 240 Pixel)
REL	Rights Expression Language
RSA	Rivest, Sharmir and Adleman
RTCP	RTP Control Protocol
RTP	Realtime Transport Protocol
RTSP	Real Time Streaming Protocol
SAML	Security Assertion Markup Language
SDP	Session Description Protocol
SHA	Secure Hash Algorithm
SOAP	Simple Object Access Protocol
SOI	Start Of Image
SOF	Start Of Frame
SOS	Start Of Scan
SR	Sender Report
SSID	Service Set Identifier
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TKIP	Temporal Key Integrity Protocol
TTL	Time To Live
UDDI	Universal Description, Discovery and Integration
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URN	Uniform Resource Name
USB	Universal Serial Bus
UDDI	Universal Description Discovery and Integration
UTC	Coordinated Universal Time
UTF	Unicode Transformation Format
UUID	Universally Unique Identifier
WDR	Wide Dynamic Range
WPA	Wi-Fi Protected Access
WS	Web Services
WSDL	Web Services Description Language

WS-I	Web Services Interoperability
XML	eXtensible Markup Language
XPath	XML Path Language

4 Overview

This standard is based on network video use cases covering both local and wide area network scenarios. The specification starts from a core set of interface functions for configuration and operation of network video devices by defining their server side interfaces. The set of network video devices includes Network Video Transmitter (NVT), Network Video Display (NVD), Network Video Storage (NVS) and Network Video Analytics (NVA). The framework is designed to be extended and enhanced in future versions.

The framework covers procedures from the network video device deployment and the configuration phase to the real time streaming phase for different network video scenarios.

This standard covers device discovery, device configuration, events, PTZ control, video analytics and real time streaming functionality for live video, as well as search, replay and recording management functionality for recorded video.

All services share a common XML schema and all data types are provided in [ONVIF Schema]. The different services are defined in the respective sections and service WSDL documents.

4.1 Web services

The term Web services is the name of a standardized method of integrating applications using open, platform independent Web Services standards such as XML, SOAP 1.2 [Part 1] and WSDL1.1 over an IP network. XML is used as the data description syntax, SOAP is used for message transfer and WSDL is used for describing the services.

This framework is built upon Web Services standards. All configuration services defined in the standard are expressed as Web Services operations and defined in WSDL with HTTP as the underlying transport mechanism.

Figure 1 gives an overview of the basic principles for development based on Web Services. The service provider (device) implements the ONVIF service or services.

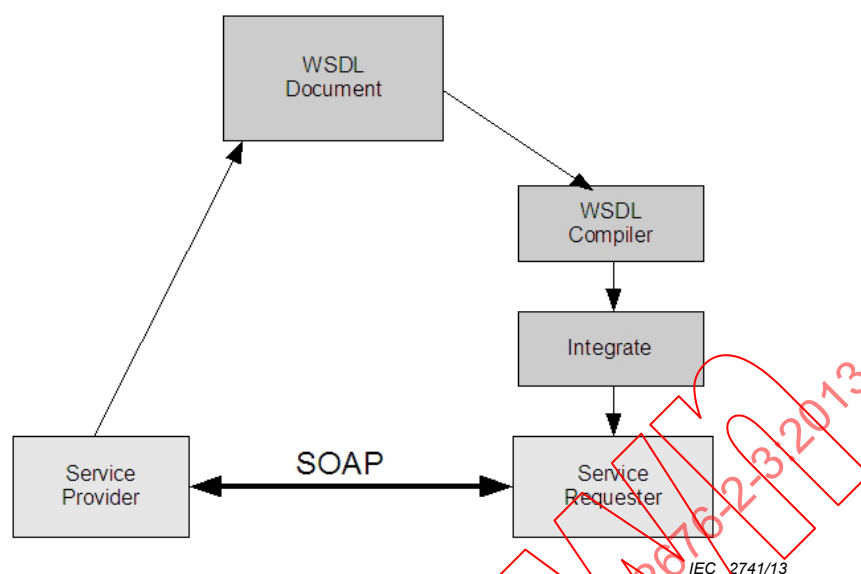


Figure 1 – Web Services based development principles

The service is described using the XML-based WSDL. Then, the WSDL is used as the basis for the service requester (client) implementation/integration. Client-side integration is simplified through the use of WSDL compiler tools that generate platform specific code that can be used by the client side developer to integrate the Web Service into an application.

The Web Service provider and requester communicate using the SOAP message exchange protocol. SOAP is a lightweight, XML-based messaging protocol used to encode the information in a Web Service request and in a response message before sending them over a network. SOAP messages are independent of any operating system or protocol and may be transported using a variety of Internet protocols. This ONVIF standard defines conformant transport protocols for the SOAP messages for the described Web Services.

The Web Service overview section defines the different ONVIF services, the command definition syntax in the specification, error handling principles and the adopted Web Service security mechanisms.

To ensure interoperability, all defined services follow the Web Services Interoperability Organization (WS-I) basic profile 2.0 recommendations and use the document/literal wrapped pattern.

4.2 IP configuration

The IP configuration section defines the IP configuration compliance requirements and recommendations. IP configuration includes:

- IP network communication capability;
- static IP configuration;
- dynamic IP configuration.

4.3 Device discovery

The configuration interfaces defined in this standard are Web Services interfaces that are based on the WS-Discovery standard. This use of this standard makes it possible to reuse a suitable existing Web Service discovery framework, instead of requiring a completely new service or service addressing definition.

This standard introduces a specific discovery behaviour suitable for video surveillance purposes. For example, a fully interoperable discovery requires a well defined service definition and a service searching criteria. The specification covers device type and scopes definitions in order to achieve this.

A successful discovery provides the device service address. Once a client has the device service address it can receive detailed device information through the device service, see 4.5 below.

In addition to the standard web services discovery protocol this standard supports remote discovery proxies to find registered devices through the remote discovery proxy even if the client and the device reside in different administrative network domains.

4.4 Device types

The device type signals the primary function of a device. This standard specifies the following set of device types:

- network Video Transmitter (NVT);
- network Video Display (NVD);
- network Video Storage (NVS);
- network Video Analytics (NVA).

For each device type a number of services are mandatory which are defined in 5.1.2. A device may support other optional services and device signals availability of optional services via the device discovery.

4.5 Device management

Device management functions are handled through the device service. The device service is the entry point to all other services provided by a device. WSDL for the device service is provided in the Device Management WSDL file. The device management interfaces consist of these subcategories:

- capabilities;
- network;
- system;
- security

4.5.1 Capabilities

The capability commands allow a client to ask for the services provided by a device and to determine which general and vendor specific services are offered by the device. The capabilities are structured as the different device services and are further divided into subcategories (when applicable) as follows:

- analytics,
- device,
 - capabilities,
 - network,
 - system,
 - I/O,
 - security,
- event,
- imaging,

- media,
- PTZ,
- device IO,
- display,
- recording,
- search,
- replay,
- analytics device.

The capabilities for the different categories indicate those commands and parameter settings that are available for the particular service or service subcategory.

4.5.2 Network

The following set of network commands allows standardized management of functions:

- get and set hostname;
- get and set DNS configurations;
- get and set NTP configurations;
- get and set dynamic DNS;
- get and set network interface configurations;
- enable/disable and list network protocols;
- get and set default gateway;
- get and set zero configuration;
- get, set, add and delete IP address filter.

4.5.3 System

The system commands are used to manage the following device system settings:

- get device information;
- make system backups;
- get and set system date and time;
- factory default reset;
- upgrade firmware;
- get system log;
- get device diagnostics data (support information);
- reboot;
- get and set device discovery parameters.

4.5.4 Retrieval of system information

System Information, such as system logs, vendor-specific support information and configuration backup images, may be retrieved using either MTOM or HTTP.

The MTOM method is supported by the GetSystemLog, GetSystemSupportInformation and GetSystemBackup commands. The HTTP method is supported by the GetSystemUri command; this retrieves URIs from which the files may be downloaded using an HTTP GET operation.

4.5.5 Firmware upgrade

Two mechanisms are provided for upgrading the firmware on a device. The first uses the UpgradeSystemFirmware command to send the new firmware image using MTOM.

The second is a two stage process; first the client sends the StartFirmwareUpgrade command to instruct the device to prepare for upgrade, then it sends the firmware image using HTTP POST.

The HTTP method is designed for resource-limited devices that may not be capable of receiving a new firmware image in its normal operating state.

4.5.6 System restore

The System Restore capability allows a device's configuration to be restored from a backup image. Again two mechanisms are provided. The first uses the RestoreSystem command to send the backup image using MTOM. The second uses the StartSystemRestore command followed by an HTTP POST operation to send the backup image.

4.5.7 Security

The following security operations are used to manage the device security configurations:

- get and set access security policy;
- handle user credentials and settings;
- handle HTTPS server certificates;
- enable/disable HTTPS client authentication;
- key generation and certificate download functions;
- handle IEEE 802.1X supplicant certificate;
- handle IEEE 802.1X CA certificate;
- IEEE 802.1X configuration.

4.6 DeviceIO

The DeviceIO service offers commands to retrieve and configure the settings of physical inputs and outputs of a device.

The DeviceIO service supports the configuration of the following device interfaces:

- VideoOutputs;
- VideoSources;
- AudioOutputs;
- AudioSources;
- RelayOutputs.

The following commands list existing interfaces:

- GetVideoOutputs – gets all existing video outputs of the device;
- GetVideoSources – gets all existing video sources of the device;
- GetAudioOutputs – gets all existing audio outputs of the device;
- GetAudioSources – gets all existing audio sources of the device;
- GetRelayOutputs – gets all existing relay outputs of the device.

For VideoOutputs, VideoSources, AudioOutputs and AudioSources the following commands are supported:

- *Set<device name>Configuration* – modifies the configuration of a specific interface;
- *Get<device name>Configuration* – gets the configuration of a specific interface;
- *Get<device name>ConfigurationOptions* – gets the supported property values for a specific interface.

RelayOutputs supports following commands:

- *SetRelayOutputSettings* – modifies the configuration of a relay output;
- *SetRelayOutputState* – sets the logical state.

WSDL for the DeviceIO service is specified in [Annex C.3].

4.7 Imaging configuration

The imaging service provides configuration and control data for imaging specific properties. WSDL is part of the framework and provided in the Imaging WSDL file.

The service includes the following operations:

- get and set imaging configurations (exposure time, gain and white balance, for example);
- get imaging configuration options (valid ranges for imaging parameters);
- move focus lens;
- stop ongoing focus movement;
- get current position and move status for focus.

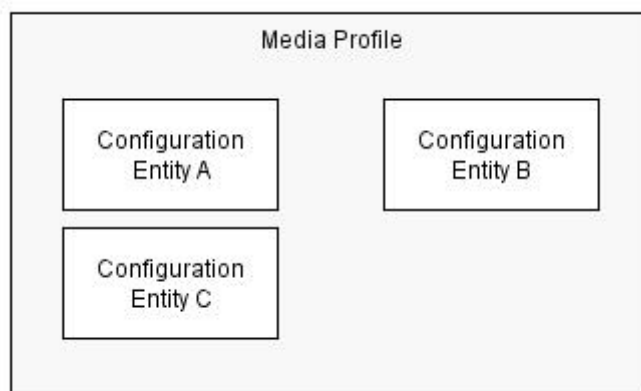
4.8 Media configuration

4.8.1 General

Media configurations are handled through the media service. Media configurations are used to determine the streaming properties of requested media streams as defined in this standard. The device provides media configuration through the media service. WSDL for the media service is provided in the Media WSDL file.

4.8.2 Media profiles

Real-time video and audio streaming configurations are controlled using media profiles. A media profile maps a video and/or audio source to a video and/or an audio encoder, PTZ and analytics configurations (see Figure 2). The NVT presents different available profiles depending on its capabilities (the set of available profiles might change dynamically though).



IEC 2742/13

Figure 2 – A media profile

A device having Media configuration service provides at least one media profile at boot. A device may provide “ready to use” profiles for the most common media configurations that the device offers.

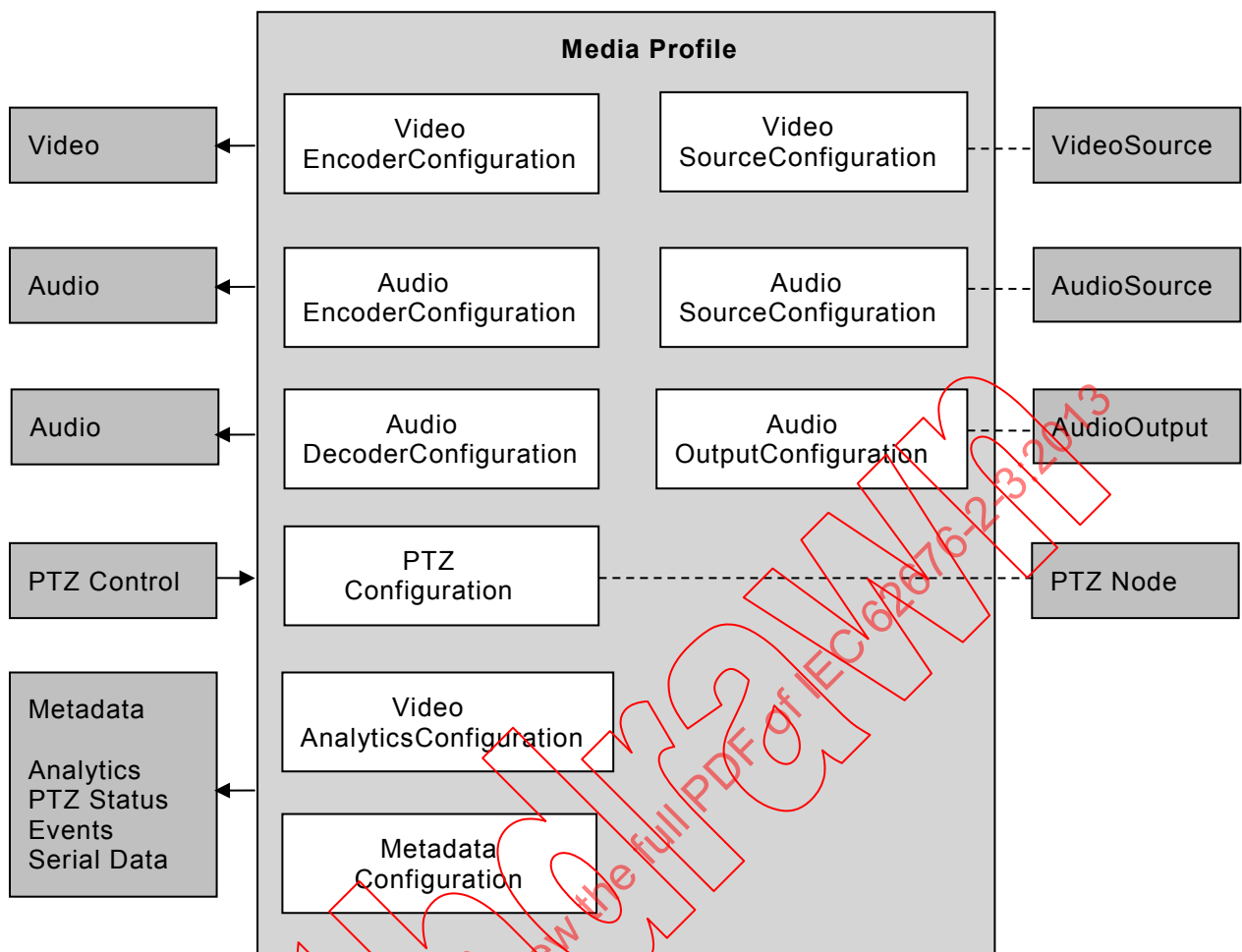
The Profile contains a “fixed” attribute that indicates if a profile can be deleted or not. If a profile is fixed or not is defined by the NVT.

A profile consists of a set of interconnected *configuration entities*. Configurations are provided by the NVT and can be either static or created dynamically by the NVT. For example, the dynamic configurations can be created by the NVT depending on current available encoding resources. A configuration entity is one of the following:

- video source configuration;
- audio source configuration;
- video encoder configuration;
- audio encoder configuration;
- PTZ configuration;
- video analytics configuration;
- metadata configuration;
- audio output configuration;
- audio decoder configuration.

A profile consists of all or a subset of these configuration entities. Depending on the capabilities of the NVT, a particular configuration entity can be part of a profile or not. For example, a profile with an audio source and an audio encoder configuration can exist only in a device with audio support.

An example of a complete profile configuration is illustrated in Figure 3.



IEC 2743/13

Figure 3 – Complete profile configuration

A media profile describes how and what to present to the client in a media stream as well as how to handle PTZ input and Analytics.

The following commands list existing sources:

- *GetVideoSources* – gets all existing video sources in the device;
- *GetAudioSources* – gets all existing audio sources in the device;
- *GetAudioOutputs* – gets all existing audio outputs in the device.

The following commands manage Media Profiles:

- *CreateProfile* – creates a new media profile;
- *GetProfiles* – gets all existing media profiles;
- *GetProfile* – gets a specific media profile;
- *DeleteProfile* – deletes a specific media profile;
- *Add<configuration entity>* – adds a specific configuration entity to the media profile;
- *Remove<configuration entity>* – removes a specific configuration entity from a media profile.

The following commands manage Configuration Entities:

- *Get<configuration entity>Options* – gets the valid property values for a specific configuration entity;
- *Set<configuration entity>* – sets a configuration entity configuration;
- *Get<configuration entity>s* – gets all existing configuration entities of the type;
- *Get<configuration entity>* – gets a specific configuration entity;
- *GetCompatible<configuration entity>s* – gets all configuration entities compatible with a specific media profile,

where *<configuration entity>* is the type of configuration entity. For example, the complete command to get a video encoder configuration is:

GetVideoEncoderConfiguration.

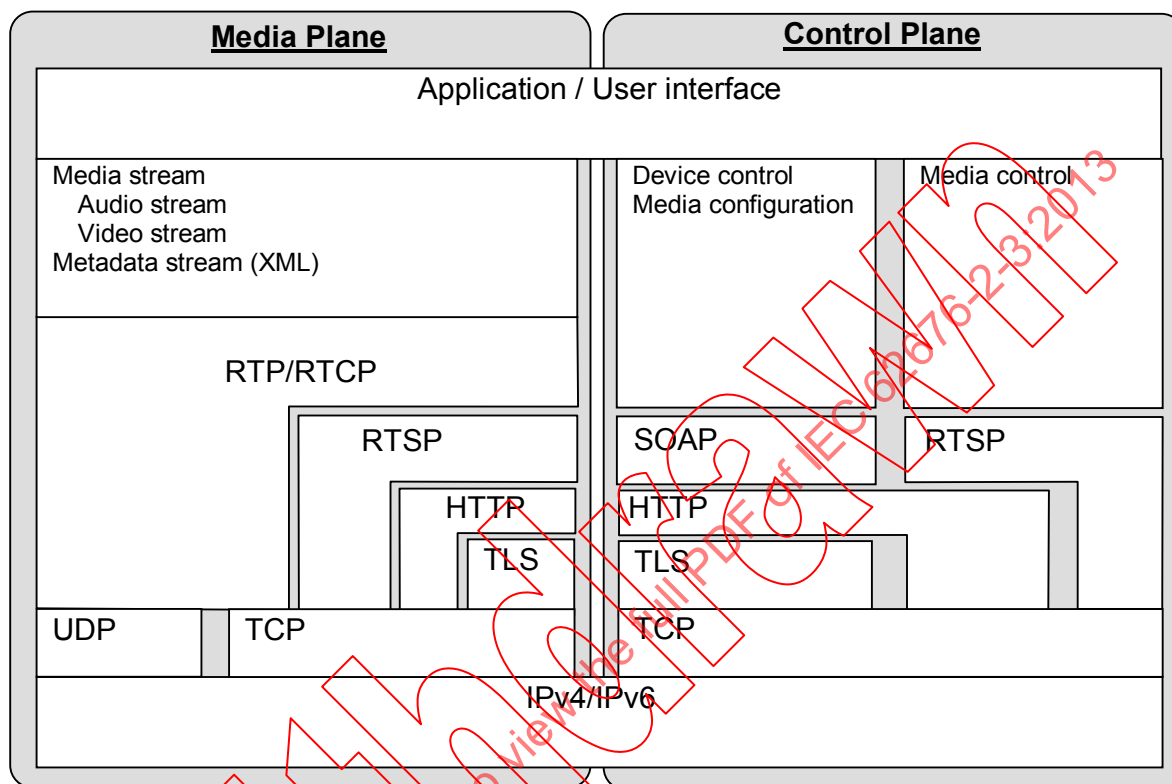
The following commands initiate and manipulate a video/audio stream:

- *GetStreamUri* – requests a valid RTSP or HTTP stream URI for a specific media profile and protocol;
- *StartMulticastStreaming* – starts multicast streaming using a specified media profile;
- *StopMulticastStreaming* – stops a multicast stream;
- *SetSynchronizationPoint* – inserts a synchronization point (I-frame, etc) in active streams;
- *GetSnapshotUri* – requests a valid HTTP URI for a specific media profile that can be used to obtain a JPEG snapshot.

Refer to Clause 5 for examples of how the profiles are used in a client implementation.

4.9 Real-time streaming

This standard defines media streaming options and formats. A distinction is made between *media plane* and *control plane*, as illustrated in Figure 4. A set of media streaming (audio, video and meta data) options, all based on RTP [RFC 3550], are described in order to provide interoperable media streaming services.



IEC 2744/13

Figure 4 – Layer structure

The metadata streaming container format allows well-defined, real-time streaming of analytics, PTZ status and notification data.

Media configuration is done over SOAP/HTTP and is covered by the media configuration service as discussed in 4.8.

Media control is accomplished over RTSP as defined in RFC 2326. This standard utilizes RTP, RTCP and RTSP profiling, as well as JPEG over RTP extensions and multicast control mechanisms.

The standard introduces extensions to the RTSP standard to allow bi-directional streaming connections.

Streaming configurations for the following video codecs are provided:

- JPEG (over RTP), see 12.1.3;
- MPEG-4, Simple Profile (SP) [ISO/IEC 14496-2];
- MPEG-4, Advanced Simple Profile (ASP) [ISO/IEC 14496-2];
- H.264, baseline [ISO/IEC 14496-10];

- H.264, main [ISO/IEC 14496-10];
- H.264, extended [ISO/IEC 14496-10];
- H.264, high [ISO/IEC 14496-10];

and for the following audio codecs:

- G.711 [ITU-T G.711];
- G.726 [ITU-T G.726];
- AAC [ISO/IEC 14496-3].

4.10 Event handling

Event handling is based on the OASIS WS-BaseNotification and WS-Topics specifications. These specifications allow the reuse of a rich notification framework without the need to redefine event handling principles, basic formats and communication patterns.

Firewall traversal, according to WS-BaseNotification, is handled through a *PullPoint* notification pattern. This pattern, however, does not allow real-time notification. Hence, this standard defines an alternative *PullPoint* communication pattern and service interface. The *PullPoint* pattern allows a client residing behind a firewall to receive real-time notifications while utilizing the WS-BaseNotification framework.

A fully standardized event requires standardized notifications. However, the notification topics will, to a large extent, depend on the application needs. This standard defines a set of basic notification topics that a device is recommended to support, see Annex A. In addition, for some services, this standard extends the basic notification topics with mandatory events.

WSDL for the event service including extensions is provided in the Event WSDL file.

4.11 PTZ control

The PTZ service is used to control a Pan Tilt and Zoom (PTZ) video encoder device. WSDL for the PTZ service is provided in the PTZ WSDL file.

The PTZ control principle follows the *MediaProfile* model (see 4.8) and consists of three major blocks:

- PTZ Node – low-level PTZ entity that maps to the PTZ device and its capabilities;
- PTZ Configuration – holds the PTZ configuration for a specific PTZ Node;
- PTZ Control Operation – PTZ, preset and status operations.

The relationship of the three items mentioned above is depicted in Figure 5.

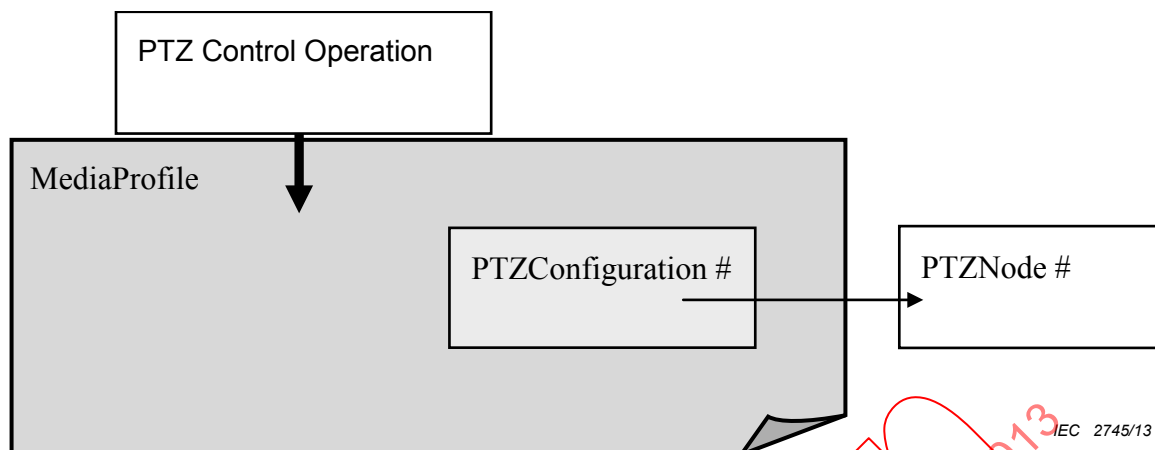


Figure 5 – PTZ control model

A PTZ capable NVT may have one or many PTZ nodes. The PTZ node may be a mechanical PTZ driver, an uploaded PTZ driver on a video encoder or a digital PTZ driver. The PTZ node is the lowest level entity of the PTZ Control and it specifies the supported PTZ capabilities.

PTZ configurations are set *per media profile* and are handled through these configuration commands:

- get and set configurations for pan, tilt and zoom;
- get configuration options for pan, tilt and zoom.

This standard defines the following PTZ control operations:

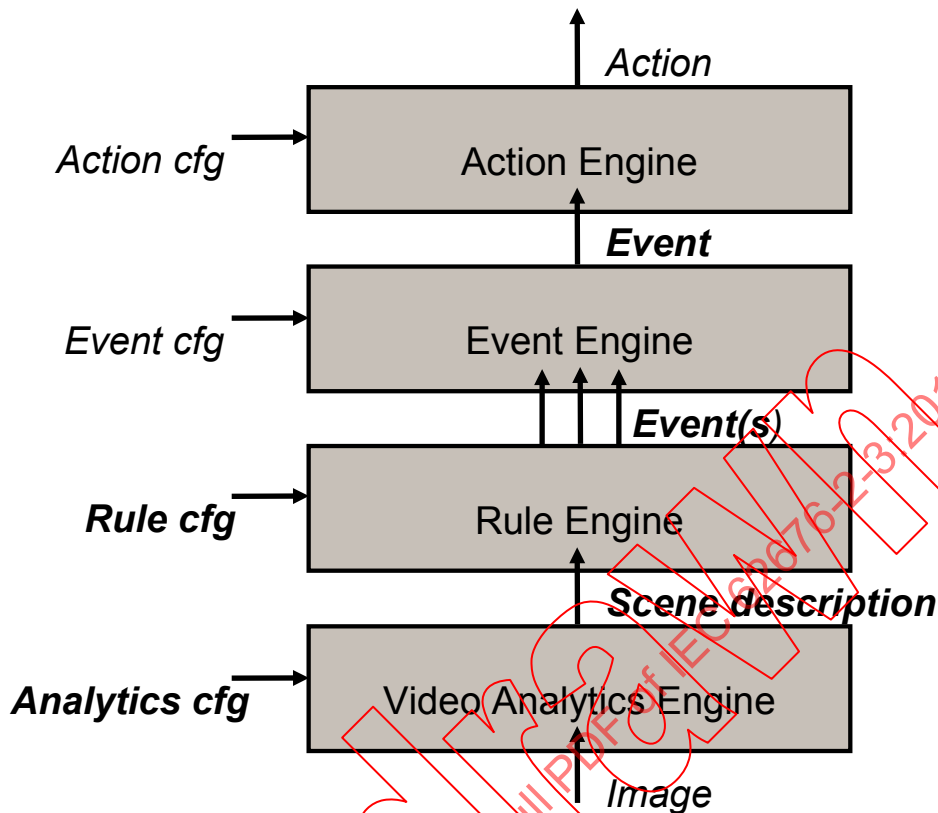
- PTZ absolute, relative and continuous move operations;
- stop operation;
- get PTZ status information (position, error and move status, for example);
- get, set, remove and move to preset position;
- get, set and move to home position.

4.12 Video analytics

Video analytic applications are divided into image analysis and application-specific parts. The interface between these two parts produces an abstraction that describes the scene based on the objects present. Video analytic applications are reduced to a comparison of the scene descriptions and of the scene rules (such as virtual lines that are prohibited to cross, or polygons that define a protected area). Other rules may represent intra-object behaviour such as objects following other objects (to form a tailgating detection). Such rules can also be used to describe prohibited object motion, which may be used to establish a speed limit.

These two separate parts, referred to as the video analytics engine and as the rule engine, together with the events and actions, form the video analytics architecture according to this standard as illustrated in Figure 6.

The video analytics architecture consists of elements and interfaces. Each element provides a functionality corresponding to a semantically unique entity of the complete video analytics solution. Interfaces are unidirectional and define an information entity with a unique content. Only the interfaces are subject to this specification. Central to this architecture is the ability to distribute any elements or sets of adjacent elements to any device in the network.



IEC 2746/13

Figure 6 – Video analytics architecture

The following interfaces are defined in this standard:

- analytics configuration interface;
- scene description;
- rule configuration interface;
- event interface.

The standard defines a configuration framework for the Video Analytics Engine. This framework enables a client to ask the device for supported analytics modules responsible for their configurations. Configurations of such modules can be dynamically added, removed or modified by a client, allowing a client to run multiple Video Analytics Modules in parallel if supported by the device.

The output from the Video Analytics Engine is called a *Scene Description*. The Scene Description represents the abstraction of the scene in terms of the objects, either static or dynamic, that are part of the scene. This standard defines an XML-based Scene Description Interface including data types and data transport mechanisms.

Rules describe how the scene description is interpreted and how to react on that information. The standard defines standard rule syntax and methods to communicate these rules from the application to the device.

An event signals the state of the analysis of the scene description and the associated rules. The event interface is both the input and the output of the event engine element. The event interface is handled through the general notification and topics framework (see 4.10).

WSDL for the video analytics service is part of the framework and provided in the Analytics WSDL file.

4.13 Analytics device

The Analytics Device Service has to be used for stand alone analytics devices which perform evaluation processes on media streams or metadata enhanced media streams. Evaluations may involve more than one media stream or metadata enhanced media stream at a time.

The Analytics Device Service receives media streams or metadata enhanced media streams from live-generating or storing devices. It could comprise decoder capabilities if analysis is being performed on uncompressed data.

The Analytics Device Service is being used by a Client to configure properties and functionality of a stand alone analytics device. Backchannel capabilities are not provided by stand alone analytics devices.

The output of the Analytics Device Service can be obtained using the Event Service, additionally the GetStreamUri command is supported.

4.14 Display

The display service provides functions to enable a client to control and configure display devices. The service introduces panes, each of which occupies an area of the physical display. The configuration of the pane maps audio inputs and outputs to a video output. The configuration also references a Receiver Object which receives the data to be displayed. Functions to retrieve and configure the configuration of a pane are provided.

A layout defines how these panes are visible on the display (e.g. single view or quad view). The service introduces commands to retrieve the current layout of a display and change the layout.

The service also introduces commands to request the encoding and decoding capabilities of a video output as well as the layout options.

4.15 Receiver

4.15.1 General

A receiver is an object that acts as an RTSP client endpoint. Receivers are used by other services that consume media streams, such as the Display, Recording and Analytics Device services. A receiver has a configuration that determines the RTSP endpoint to which it should connect and the connection parameters it should use.

A receiver can operate in three distinct modes:

- Always Connect. The receiver attempts to maintain a persistent connection to the configured endpoint.
- Never Connect. The receiver does not attempt to connect.
- Auto Connect. The receiver connects on demand, as required by consumers of the media streams.

A single receiver may be used by more than one consumer. For example, in order to record a stream and also perform analytics on it, both a recording job and an analytics engine could be attached to the same receiver. If the receiver uses the “Auto Connect” mode, it will connect whenever either the recording job or the analytics engine is active, and disconnect when neither of them are active.

Receivers may be created and deleted either manually, by calling the CreateReceiver and DeleteReceiver operations in the Receiver Service, or automatically by other services. For example, if a recording job is created with the “AutoCreateReceiver” option, it will automatically create and attach to a Receiver. Deleting the recording job will also delete the receiver.

4.15.2 Synchronization points

Because receivers use RTSP addresses to specify the source of the stream, they do not necessarily have access to the web services interface of the transmitter. This means that they cannot use the SetSynchronizationPoint command described in 11.18.2.

Instead, receivers should use the PLI message described in [RFC 4585] to request a synchronization point.

4.16 Storage

This standard provides a set of interfaces that enable the support of interoperable network storage devices, such as network video recorders (NVR), digital video recorders (DVR) and cameras with embedded storage.

The following functions are supported:

- recording control;
- search;
- replay.

These functions are provided by three interrelated services:

Recording service enables a client to manage recordings, and to configure the transfer of data from data sources to recordings. Managing recordings includes creation and deletion of recordings and tracks.

Search service enables a client to find information about the recordings on the storage device, for example to construct a “timeline” view, and to find data of interest within a set of recordings. The latter is achieved by searching for events that are included in the metadata track recording.

Replay service enables a client to play back recorded data, including video, audio and metadata. Functions are provided to start and stop playback and to change speed and direction of the replayed stream. It also enables a client to download data from the storage device so that export functionality can be provided.

4.16.1 Storage model

The storage interfaces in this standard present a logical view of the data on the storage device. This view is completely independent of the way data might be physically stored on disk.

The key concept in the storage model is that of a *recording*. The term *recording* is used in this standard to denote a container for a set of related audio, video and metadata *tracks*, typically from the same data source e.g. a camera. A *recording* could hold any number of tracks. A *track* is viewed as an infinite timeline that holds data at certain times.

At a minimum, a recording is capable of holding three tracks, one for audio, one for video and one for metadata. Some implementations of the recording service may support multiple tracks of each type. For example the same recording could hold two video tracks, one containing a low resolution or low frame rate stream and one containing a high resolution or high frame

rate stream. Figure 7 shows an example with three recordings and related track information over time.

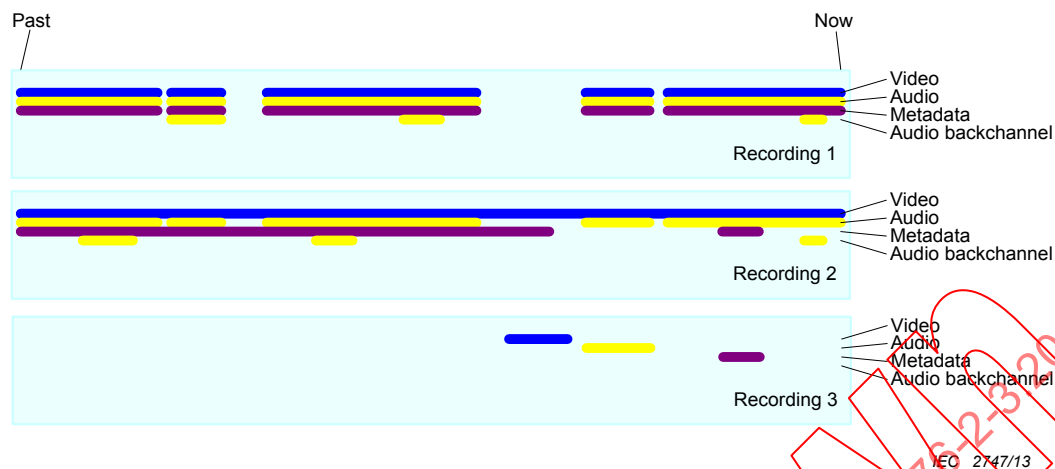


Figure 7 – Storage model with tracks

It is important to note that the storage interfaces do not expose the internal storage structures on the device. In particular, a recording is not intended to represent a single file on disk although in many storage device implementations a recording is physically stored in a series of files. For instance, some camera implementations realise alarm recording by creating a distinct file for each alarm that occurs. Although each file could be represented as a different *recording*, the intent of the model in this standard is that all these files are aggregated into a single recording.

Within a recording the regions where data is actually recorded are represented by pairs of events, where each pair comprises an event when recording started and an event when recording stopped. A client can construct the logical view of the recordings by using the FindRecordings and FindEvents methods of the search service.

If metadata is recorded, the metadata track can hold all the events generated by the data source (see the clause on event handling and the MetadataConfiguration object). In addition, a device also conceptually records ONVIF defined historical events (see Recording Event Descriptions in the search service), this includes information like start and end of a recorded data range. A device may also conceptually record vendor specific historical events. Events generated by the device are not inserted in existing metadata tracks of recordings. The FindEvents method in the search service can find all the recorded events.

4.16.2 Recording

The recording service enables a client to manage recordings, and to configure the transfer of data from data sources to recordings. Managing recordings includes creation and deletion of recordings and tracks.

Recording jobs transfer data from a recording source to a recording. A recording source can be a receiver object created with the receiver service, or it can be a media profile that encodes data on a local device. The media profile could be used as a source on a camera with embedded storage.

To save data to a recording, a client first creates a recording and ensures that the recording has the necessary tracks. Then the client creates a recording job that pulls data from one or more sources and stores the data to the tracks in the recording.

Clients may set up multiple recording jobs that all record into the same recording. If multiple recording jobs are active, the device uses a priority scheme to select between the tracks

defined in the recording jobs. Clients may change the mode of recording jobs at any time, thereby providing means to implement features like alarm recording or manual recording.

The recording job relies on the receiver service for receiving the data from other devices through receiver objects identified by ReceiverTokens.

4.16.3 Search

The search service enables a client to find information about the recordings on the storage device, for example to construct a “timeline” view, and to find data of interest within a set of recordings. The latter is achieved by searching for events and other information that is included in the metadata track recording.

The search service provides the following functionality:

- find recordings and information about each recording;
- find events in the metadata and among the historical events;
- find PTZ positions in the metadata;
- find other information in the metadata e.g. text from electronic point-of-sale systems.

The actual searching is done by coupled find and result operations and is asynchronous. Each find operation initiates a search session. The client can then acquire the results from the search session in increments, or all at once, depending on implementation and the scale of the search. There are four pairs of search operations for recordings, recording events, PTZ positions and metadata.

FindRecordings and GetRecordingSearchResults

FindEvents and GetEventSearchResults

FindPTZPosition and GetPTZPositionSearchResults

FindMetadata and GetMetadataSearchResults

4.16.4 Replay

The replay service provides a mechanism for replay of stored video, audio and metadata. This mechanism may also be used to download data from the storage device so that export functionality can be provided.

The replay protocol is based on RTSP [RFC 2326]. However because RTSP does not directly support all of the requirements for replay, several extensions have been added to the protocol. In particular, an RTP header extension is defined to allow an absolute timestamp to be associated with each access unit (e.g. video frame), and to convey information about stream continuity.

The GetReplayUri command in the replay service returns the RTSP URL of a recording to allow it to be replayed using RTSP.

4.17 Security

This subclause describes network video security requirements. This standard defines security mechanism on two different communication levels:

- transport-level security;
- message-level security.

This standard also defines port-based network security as follows.

- IEEE 802.1X

The general security requirements, definitions and transport security requirements are specified in Clause 0. Message level security requirements are specified in 5.12. IEEE 802.1X requirements are specified in 0 Security management is handled through the device management service as listed above in 4.5.7.

5 Web Services framework

All management and configuration commands are based on Web Services.

For the purpose of this standard:

- the device (NVT, NVD, NVS, NVA) is a service provider;
- the client is a service requester.

A typical network video system does not have a single client that handles all device configuration and device management operations for one device. Instead, a distinction between *control* and network video *receiver* functionality may exist. A device providing services may also act as a client. Future editions of the standard may introduce additional entities and interfaces in the system.

Web Services also require a common way to discover service providers. This discovery is achieved using the Universal Discovery, Description and Integration Registry (UDDI) specifications [UDDI API ver2], [UDDI Data Structure ver2]. The UDDI specifications utilize service brokers for service discovery. This specification targets devices while the UDDI model is *not* device oriented. Consequently, UDDI and service brokers are *outside the scope* of this standard.

According to this standard, devices (service providers) are discovered using WS-Discovery [WS-Discovery] based techniques. The service discovery principles are described in Clause 7.

Web Services allow developers the freedom to define services and message exchanges, which may cause interoperability problems. The Web Services interoperability organization (WS-I) develops standard profiles and guidelines to create interoperable Web Services. The devices and the clients shall follow the guidelines in the WS-I Basic Profile 2.0 [WS-I BP 2.0]. The service descriptions in this standard follow the WS-I Basic Profile 2.0 recommendations.

5.1 Services overview

5.1.1 General

An ONVIF compliant device shall support a number of Web Services which are defined in this standard. Examples for web services defined by this standard are:

- device service;
- media service;
- event service.

The device service is the *target service* of an ONVIF compliant device and the *entry point* for all other services of the device.

The entry point for the device management is fixed to:

`http://onvif_host/onvif/device_service`

5.1.2 Services requirements

A device shall provide the device management and event service. A device MAY support any of the other services depending on the capabilities of the device. Depending on the device type (NVT, NVD, NVS, NVA) additional services are required. The exact compliance requirements are defined as part of the different service definitions in this standard.

If a device supports a certain service, the device shall respond to all commands defined in the corresponding service WSDL. If the specific command is not required for that service and the device does not support the command, the device shall respond to a request with the error codes:

env:Receiver,

ter:ActionNotSupported,

see 5.11.2 for the definitions of the error codes.

Table 1 shows which services are required for the different device types. Mandatory services are marked with 'M' and services that are mandatory if a related feature is supported by the device are marked with 'C'.

Table 1 – Service requirements for the device types

	NVT	NVS	NVD	NVA
Device	M	M	M	M
Event	M	M	M	M
Media	M			
PTZ	C			
Imaging				
Analytics				M
Recording Control		C		
Recording Search		M		
Replay Control		M		
Device IO	M		M	
Receiver		C	M	M
Display			M	
Analytics Device				M

5.2 WSDL overview

“WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate” [WSDL1.1].

This standard follows the WSDL 1.1 specification and uses the document/literal wrapped pattern.

A WSDL document consists of the following sections:

- types – definition of data types using XML schema definitions;
- message – definition of the content of input and output messages;
- operation – definition of how input and output messages are associated with a logical operation;
- portType – groups a set of operations together;
- binding – specification of which protocols that are used for message exchange for a particular portType;
- port – specifies an address for a binding;
- service – used to group a set of related ports.

5.3 Namespaces

Prefix and namespaces used in this standard are listed in Table 2. These prefixes are not part of the standard and an implementation can use any prefix.

Table 2 – Defined namespaces in this standard

Prefix	Namespace URI	Description
tt	http://www.onvif.org/ver10/schema	XML schema descriptions in this standard.
tds	http://www.onvif.org/ver10/device/wsdl	The namespace for the WSDL device service.
trt	http://www.onvif.org/ver10/media/wsdl	The namespace for the WSDL media service.
timg	http://www.onvif.org/ver20/imaging/wsdl	The namespace for the WSDL imaging service.
tev	http://www.onvif.org/ver10/events/wsdl	The namespace for the WSDL event service.
tptz	http://www.onvif.org/ver20/ptz/wsdl	The namespace for the PTZ control service.
tan	http://www.onvif.org/ver20/analytics/wsdl	The namespace for the analytics service.
ter	http://www.onvif.org/ver10/error	The namespace for ONVIF defined faults.
dn	http://www.onvif.org/ver10/network/wsdl	The namespace used for the <i>remote</i> device discovery service in this standard.
tns1	http://www.onvif.org/ver10/topics	The namespace for the ONVIF topic namespace
tad	http://www.onvif.org/ver10/analyticsdevice/wsdl	The namespace for the WSDL analytics device service.
tmd	http://www.onvif.org/ver10/deviceIO/wsdl	The namespace for the WSDL deviceIO service.
tls	http://www.onvif.org/ver10/display/wsdl	The namespace for the WSDL display service
trv	http://www.onvif.org/ver10/receiver/wsdl	The namespace for the WSDL receiver service.
trc	http://www.onvif.org/ver10/recording/wsdl	The namespace for the WSDL recording service.
trp	http://www.onvif.org/ver10/replay/wsdl	The namespace for the WSDL replay service.
tse	http://www.onvif.org/ver10/search/wsdl	The namespace for the WSDL search service

The namespaces listed in Table 3 are referenced by this standard.

Table 3 – Referenced namespaces (with prefix)

Prefix	Namespace URI	Description
wsdl	http://schemas.xmlsoap.org/wsdl/	WSDL namespace for WSDL framework.
wssoap12	http://schemas.xmlsoap.org/wsdl/soap12/	WSDL namespace for WSDL SOAP 1.2 binding.
http	http://schemas.xmlsoap.org/wsdl/http/	WSDL namespace for WSDL HTTP GET & POST binding.
soapenc	http://www.w3.org/2003/05/soap-encoding	Encoding namespace as defined by SOAP 1.2 [SOAP 1.2, Part 2]
soapenv	http://www.w3.org/2003/05/soap-envelope	Envelope namespace as defined by SOAP 1.2 [SOAP 1.2, Part 1]
xs	http://www.w3.org/2001/XMLSchema	Instance namespace as defined by XS [XML-Schema, Part 1] and [XML-Schema, Part 2]
xsi	http://www.w3.org/2001/XMLSchema-instance	XML schema instance namespace.
d	http://schemas.xmlsoap.org/ws/2005/04/discovery	Device discovery namespace as defined by [WS-Discovery].
wsadis	http://schemas.xmlsoap.org/ws/2004/08/addressing	Device addressing namespace referred in WS-Discovery [WS-Discovery].
wsa	http://www.w3.org/2005/08/addressing	Device addressing namespace as defined by [WS-Addressing].
wstop	http://docs.oasis-open.org/wsn/t-1	Schema namespace of the [WS-Topics] specification.
wsnt	http://docs.oasis-open.org/wsn/b-2	Schema namespace of the [WS-BaseNotification] specification.
xop	http://www.w3.org/2004/08/xop/include	XML-binary Optimized Packaging namespace as defined by [XOP]

In addition this standard refers without prefix to the namespaces listed in Table 4.

Table 4 – Referenced namespaces (without prefix)

Namespace URI	Description
http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete	Topic expression dialect defined for topic expressions.
http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet	The ONVIF dialect for the topic expressions.
http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter	The ONVIF filter dialect used for message content filtering.
http://www.onvif.org/ver10/tptz/ZoomSpaces/PositionGenericSpace	The ONVIF standard zoom position space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/PositionGenericSpace	The ONVIF standard pan/tilt position space for PTZ control.
http://www.onvif.org/ver10/tptz/ZoomSpaces/TranslationGenericSpace	The ONVIF standard zoom translation space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/TranslationGenericSpace	The ONVIF standard pan/tilt translation space for PTZ control.
http://www.onvif.org/ver10/tptz/ZoomSpaces/VelocityGenericSpace	The ONVIF standard zoom velocity space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/VelocityGenericSpace	The ONVIF standard pan/tilt velocity space for PTZ control.
http://www.onvif.org/ver10/tptz/ZoomSpaces/SpeedGenericSpace	The ONVIF standard zoom speed space for PTZ control.
http://www.onvif.org/ver10/tptz/PanTiltSpaces/SpeedGenericSpace	The ONVIF standard pan/tilt speed space for PTZ control.

5.4 Types

Data types are defined using XML schema descriptions Part 1 and Part 2. All data types defined in this standard are included in [ONVIF Schema] and can be downloaded from:

- <http://www.onvif.org/onvif/ver10/schema/onvif.xsd>

5.5 Messages

According to WSDL 1.1 operations are described using input and output messages in XML. The message section contains the message content.

A message in this specification contains two main elements:

- message name,
- message parts.

The message name specifies the name of the element and that name is used in the operation definition in the WSDL document. The message name defines the name of the message.

The WSDL message part element is used to define the actual format of the message. Although there can be multiple parts in a WSDL message, this specification follows the WS-I basic profile [WS-I BP 2.0] and does not allow more than one part element in a message. Hence we always use the same name (“parameters”) for the message part name.

The following WSDL notation is used for the messages in this standard:

```
<message name="Operation_Name'Request">
    <part name="parameters" element="prefix':Operation_Name'"/>
</message>
```


respective,

```
<message name="" Operation_Name' Response">  
  
    <part name="parameters" element="" prefix':' Operation_Name' Response"/>  
  
</message>
```

where 'prefix' is the prefix for the namespace in which the message is defined.

This specification uses message specific types that encapsulate multiple parts to allow multiple arguments (or data) in messages.

5.6 Operations

Operations are defined within the WSDL portType declaration. An operation can be one of these two types:

- One-way – The service provider receives a message.
- Request-response – The service provider receives a message and sends a corresponding message.

Depending on the operation, different port types can be used.

The operation name defines the name of the operation.

Operations in the specification are defined using the following table format outlined in Table 5.

Table 5 – Operation description outline used in this standard

Operation_Name		Operation type
Message name	Description	
'Operation_Name'Request	<i>Description of the request message.</i> $Type_{r1} Name_{r1} [a_{r1}][b_{r1}]$ $Type_{r2} Name_{r2} [a_{r2}][b_{r2}]$: $Type_{rn} Name_{rn} [a_{rn}][b_{rn}]$	
'Operation_Name'Response	<i>Description of the response message.</i> $Type_{s1} Name_{s1} [a_{s1}][b_{s2}]$ $Type_{s2} Name_{s2} [a_{s2}][b_{s2}]$: $Type_{sn} Name_{sn} [a_{sn}][b_{sn}]$	
'FaultMessage_Name'	<i>In the case that operation specific faults are defined, this field describes the structure of the defined fault message.</i>	
Fault codes	Description	
Code	Description of the operation specific fault.	
Subcode		
Subcode		

The description column includes a list of the elements (if applicable) included in the request and response messages respectively. The value between brackets defines the lower and upper limits of the number of occurrences that can be expected for the element of the specified type. For example, $Name_{s2}$ in the table above occurs at least a_{s2} times and at most b_{s2} times.

Most commands *do not* define any specific fault messages. If a message is defined, it follows in the table directly after the response message.

The fault codes listed in the tables are the *specific fault* codes that can be expected from the command, see 5.11.2.2. *Any command can return a generic fault*, see 5.11.2.2.

5.6.1 One-way operation type

A one-way operation type is used when the service provider receives a control message *and does not* send any explicit acknowledge message or confirmation. This standard makes use of one-way operations for discovery and event purposes only.

This operation type is defined by a single input message.

5.6.2 Request-response operation type

A request-response operation type is used when a service provider receives a message and responds with a corresponding message.

This operation type is defined by one input, one output and multiple fault message.

5.7 Port types

A port type is a named set of abstract operations and the abstract messages involved. One single port type is a collection of several different operations.

All operation names in this standard are sorted into categories. Each operation category contains one or more operations. Each category holds only *one type* of operation and is grouped into a single *port type*. A one-way operation and a request response operation can never exist for the same port type.

5.8 Binding

A binding defines concrete protocol and transport data format specification for a particular port type. There may be any number of bindings for a given port type.

“Port_type” is a previously defined type and “Binding” is a character string starting with an upper case letter that defines the name of the binding.

Binding definitions for a device according to this standard shall follow the requirements in [WS-I BP 2.0]. This implies that the WSDL SOAP 1.2 bindings shall be used.

The SOAP binding can have different styles. A device shall use the style ‘document’ specified at the operation level.

The bindings are defined in the WSDL specifications for respective services.

5.9 Ports

The individual endpoint is specified by a single address for a binding. Each port shall be given a unique name. A port definition contains a name and a binding attribute.

This standard does not mandate any port naming principles.

5.10 Services

A service is a collection of related ports. This standard does not mandate any service naming principles.

5.11 Error handling

As with any other protocol, errors can occur during communications, protocol or message processing.

The specification classifies error handling into the following categories:

- protocol errors;
- SOAP errors;
- application errors.

5.11.1 Protocol errors

Protocol Errors are the result of an incorrectly formed protocol message, which could contain illegal header values, or be received when not expected or experience a socket timeout. To indicate and interpret protocol errors, HTTP and RTSP protocols have defined a set of standard status codes [e.g., 1xx, 2xx, 3xx, 4xx, 5xx]. According to this standard, devices and the clients shall use appropriate RTSP and HTTP protocol defined status codes for error reporting and when received handle accordingly.

5.11.2 SOAP errors

SOAP Errors are generated as a result of Web Services operation errors or during SOAP message processing. All such SOAP errors shall be reported and handled through SOAP fault messages. The SOAP specification provides a well defined common framework to handle errors through SOAP fault.

A SOAP fault message is a normal SOAP message with a single well-known element inside the body (soapenv:Fault). To understand the error in more detail, SOAP has defined SOAP fault message structure with various components in it.

- fault code;
- subcode;
- reason;
- node and role;
- fault details.

Subcode and **Fault Detail** elements information items are intended for carrying application specific error information.

This standard uses a separate name space for specific faults (see 5.11.2.2):

ter = "http://www.onvif.org/ver10/error".

SOAP fault messages for different Web Services are defined as part of the different Web Services definitions. Server and client shall use SOAP 1.2 fault message handling as specified in this standard and shall follow the WS-I Basic Profile 2.0 fault handling recommendations.

The following example is an error message (SOAP 1.2 fault message over HTTP). The values in *italics* are placeholders for actual values.

```
HTTP/1.1 500 Internal Server Error
CONTENT-LENGTH: bytes in body
CONTENT-TYPE: application/soap+xml; charset="utf-8"
DATE: when response was generated
<?xml version="1.0" ?>
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-
envelope"
  xmlns:ter="http://www.onvif.org/ver10/error"
  xmlns:xs="http://www.w3.org/2000/10/XMLSchema">
<soapenv:Body>
  <soapenv:Fault>
    <soapenv:Code>
      <soapenv:Value>fault code</soapenv:Value>
      <soapenv:Subcode>
        <soapenv:Value>ter:fault subcode</soapenv:Value>
        <soapenv:Subcode>
          <soapenv:Value>ter:fault subcode</soapenv:Value>
        </soapenv:Subcode>
      </soapenv:Subcode>
    </soapenv:Code>
  </soapenv:Fault>
</soapenv:Body>
```

```

    </soapenv:Subcode>
  </soapenv:Code>
  <soapenv:Reason>
    <soapenv:Text xml:lang="en">fault reason</soapenv:Text>
  </soapenv:Reason>
  <soapenv:Node>http://www.w3.org/2003/05/soap-
envelope/node/ultimateReceiver</soapenv:Node>
  <soapenv:Role>http://www.w3.org/2003/05/soap-
envelope/role/ultimateReceiver</soapenv:Role>
  <soapenv:Detail>
    <soapenv:Text>fault detail</soapenv:Text>
  </soapenv:Detail>
</soapenv:Fault>
</soapenv:Body>
</soapenv:Envelope>

```

Table 6 summarizes the general SOAP fault codes (fault codes are defined in SOAP version 1.2 Part 1: Messaging Framework). Server and client MAY define additional fault subcodes for use by applications.

We distinguish between generic faults and specific faults. Any command can generate a generic fault. Specific faults are related to a specific command or set of commands. Specific faults that apply to a particular command are defined in the command definition table.

The fault tables of this standard define the Fault Code, Subcode and Fault Reason are normative values. The description column is added for information.

5.11.2.1 Generic faults

Table 6 lists the generic fault codes and, if applicable, subcodes. All server and client implementations shall handle all the faults listed below. Any web service command may return one or several of the generic faults.

The faults listed without *subcode* do not have any *subcode* value.

Table 6 – Generic faults

Fault Code	Subcode	Fault Reason	Description
env:VersionMismatch		SOAP version mismatch	The device found an invalid element information item instead of the expected <i>Envelope</i> element information item.
env:MustUnderstand		SOAP header blocks not understood	One or more mandatory SOAP header blocks were not understood.
env:DataEncodingUnknown		Unsupported SOAP data encoding	SOAP header block or SOAP body child element information item is scoped with data encoding that is not supported by the device.
env:Sender	ter:WellFormed	Well-formed Error	XML Well-formed violation occurred.
env:Sender	ter:TagMismatch	Tag Mismatch	There was a tag name or namespace mismatch.
env:Sender	ter:Tag	No Tag	XML element tag was missing.
env:Sender	ter:Namespace	Namespace Error	SOAP Namespace error occurred.
env:Sender	ter:MissingAttr	Required Attribute not present	There was a missing required attribute.
env:Sender	ter:ProhibAttr	Prohibited Attribute	A prohibited attribute was present.
env:Sender	ter:InvalidArgs	Invalid Args	An error due to any of the following: missing argument too many arguments arguments are of the wrong data type.
env:Sender	ter:InvalidArgVal	Argument Value Invalid	The argument value is invalid.
env:Sender	ter:UnknownAction	Unknown Action	An unknown action is specified.
env:Sender	ter:OperationProhibited	Operation not Permitted	The requested operation is not permitted by the device.
env:Sender	ter:NotAuthorized	Sender not Authorized	The action requested requires authorization and the sender is not authorized.
env:Receiver	ter:ActionNotSupported	Optional Action Not Implemented	The requested action is optional and is not implemented by the device.
env:Receiver	ter:Action	Action Failed	The requested SOAP action failed.
env:Receiver	ter:OutOfMemory	Out of Memory	The device does not have sufficient memory to complete the action.
env:Receiver	ter:CriticalError	Critical Error	The device has encountered an error condition which it cannot recover by itself and needs reset or power cycle.

5.11.2.2 Specific faults

Specific faults apply only to a specific command or set of commands. The specific faults are declared as part of the service definitions in this standard.

5.11.2.3 HTTP errors

If the server waits for the start of the inbound message and no SOAP message is received, the server shall NOT generate a SOAP fault and instead sends an HTTP error response according to Table 7.

Table 7 – HTTP errors

HTTP Error	HTTP Error Code	HTTP Reason
Malformed Request	400	Bad Request
Requires Authorization	401	Unauthorized
HTTP Method is neither POST or GET	405	Method Not Allowed
Unsupported message encapsulation method	415	Unsupported media

A server should avoid reporting internal errors as this can expose security weaknesses that can be misused.

5.12 Security

The services defined in this standard shall be protected using the WS-Security framework. The WS-Security specification defines a standard set of SOAP extensions that can be used to provide Web Services message integrity and confidentiality. The framework allows several different security models using tokens. The following tokens are currently defined:

- User name token profile [WS-UsernameToken];
- X.509 security token profile [WS-X.509Token];
- SAML token profile [WS-SAMLToken];
- Kerberos token profile [WS-KerberosToken];
- Rights Expression Language (REL) Token Profile [WS-RELTOKEN].

Server and client shall support the user name token profile as specified in WS-Security and 5.12.2 and MAY support any of the other WS-security defined profiles.

The user name token profile gives only a rudimentary level of security. In a system where security is important, it is recommended to always configure the device for TLS-based access (see 22.1). The user name token message level security combined with TLS, with client and server authentication, protected transport level security gives an acceptable level of security in many systems.

A ONVIF compliant device shall when authenticating in RTSP and HTTP use credentials from the same set of credentials that are defined for the web service part. For user defined with the user name token profile, digest authentication [RFC 2617] shall be used for RTSP and HTTP.

An ONVIF compliant device should authenticate a WS request at the WS level, HTTP should only be used as a transport protocol and the device shall not authenticate a WS request at this level.

An ONVIF compliant device should authenticate a RTSP request at the RTSP level, if HTTP is used to tunnel the RTSP request the device shall not authenticate at this level.

An ONVIF compliant device shall when authentication RTSP and HTTP methods use user / credentials from the same set of users / credentials that are used for the WS part. For user defined with the user name token profile, digest authentication [RFC 2617] shall be used for RTSP and HTTP.

5.12.1 User-based access control

The WS-Security framework allows protection and authentication on the SOAP message level. These authentication mechanisms are used to build an access security policy for an ONVIF

service. This specification allows security policy configuration based on four different user levels:

- 1) administrator;
- 2) operator;
- 3) media user;
- 4) anonymous.

A detailed access policy for different user classes can be defined using these categories. Unauthenticated users are placed into the anonymous category and a device shall not allow users to be added to the anonymous user level category.

It shall be possible to define the *exact access security policy* by the device user or by a system administrator. The exact format of the policy configuration file is *outside the scope* of this standard.

Commands to get and set an access security policy in arbitrary format are defined in 8.4.

5.12.2 User token profile

5.12.2.1 General

The only mandatory WS-security token profile is the user token profile [WS-UsernameToken].

A client shall use both nonce and timestamps as defined in [WS-UsernameToken]. The server shall reject any Username Token not using *both nonce and creation* timestamps.

This standard defines a set of command for managing the user name token profile credentials, see 0. These commands allow associating users with the different user levels defined in 5.12.1.

5.12.2.2 Password derivation

The use of the same credentials on several devices introduces a certain security risk. To require the user to supply a unique credential for each device is not feasible, instead a client should implement the following password derivation algorithm.

Denote by UA an arbitrary user. Denote by P-UA the password value used by user UA to access the devices in the system. Furthermore, denote, by NEP, the end device service point reference value for a particular device in the system. Finally, denote by PE-UA the password equivalent used by the client to access a particular device in the system. The client should calculate the PE-UA as follows:

$$PE\_UA = \text{base64}(\text{HMAC\_SHA-1}(UA + P\_UA, \text{NEP} + \text{"ONVIF password"})),$$

where "+" denotes concatenation and where the "ONVIF password" is an ASCII string. It should be included in the exact form it is given without a length byte or trailing null character, i.e., the following hexadecimal value: 4F 4E 56 49 46 20 70 61 73 73 77 6F 72 64.

HMAC_SHA-1 is the algorithm specified in [RFC 2104] using SHA-1 [FIPS 180-2] as the underlying algorithm. The key value to use for the HMAC function is the user password, P-UA, directly mapped to its binary equivalent. Similar, the value PE-UA should be mapped to its ASCII equivalent before transmitting it to the device.

base64 is described in [RFC 3548], note that the result of the base64 operation is the actual password equivalent and shall be used as it is.

Example

Assume the following password and password is used by the client (ASCII): “user” and “VRxuNzpqR”, i.e.,

UA = 75 73 65 72

P_UA = 56 52 78 75 4E 7A 70 71 72 58

Next, assume the device has the following device service end point reference value:

Urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6.

Then the password equivalent to be used will be then calculated as:

PE_UA = base64(HMAC_SHA-1(P_UA, NEP+“ONVIF password”)) =
 base64(HMAC_SHA-1(75736572565278754E7A70717258,
 F81D4fAE7DEC11D0A76500A0C91E6BF6+4F4E5649462070617373776F7264)) =
 base64(16 E5 C5 A9 4D DE 8A 97 6D D7 2F 55 78 5F C2 D0 6B DA 53 4A)=
 FuXFqU3eipdt1y9VeF/C0GvaU0o=

The resulting password equivalence “FuXFqU3eipdt1y9VeF/C0GvaU0o=” is the password that shall be used by a client both for configuring the user credential on the particular device and then also for accessing the device.

6 IP configuration

The device and client communicate over an open or closed IP network. This standard does not place any general restrictions or requirements on the network type. It shall be possible, however, to establish communication links between the entities according to the architectural framework specified in Clause 4. Device IP configuration includes parameters such as IP addresses and a default gateway.

The device shall have at least one network interface that gives it IP network connectivity. Similarly, the client shall have at least one network interface that gives IP connectivity and allows data communication between the device and the client.

The device and client shall support IPv4 based network communication. The device and client should support IPv6 based network communication.

It shall be possible to make static IP configuration on the device using a network or local configuration interface.

The device should support dynamic IP configuration of link-local addresses according to [RFC 3927]. A device that supports IPv6 shall support stateless IP configuration according to [RFC 4862] and neighbour discovery according to RFC 4861.

The device shall support dynamic IP configuration according to [RFC 2131]. A device that supports IPv6 shall support stateful IP configuration according to [RFC 3315].

The device MAY support any additional IP configuration mechanism.

Network configuration of a device is accomplished through an IP network interface or through a local interface such as USB, serial port, Bluetooth or NFC. IP configuration through a local interface is *outside* the scope of this standard. It shall be possible to make device IP configurations through the parameter configuration interface specified in 0. A device user can enable or disable any of the IP address configuration options according to this standard through a network configuration interface. The default device configuration shall be to have both DHCP and dynamic link-local (stateless) address configuration enabled. Even if the device is configured through a static address configuration it should have the link-local address default enabled.

When a device is connected to an IPv4 network, address assignment priorities (link local versus routable address) should be done as recommended in [RFC 3927].

Further details regarding how the IP connectivity is achieved are *outside* the scope of this standard.

7 Device discovery

7.1 General

A client searches for available devices using the dynamic Web Services discovery protocol [WS-Discovery].

A device compliant with this standard shall implement the Target Service role as specified in [WS-Discovery].

A client compliant with this standard shall implement the Client role as specified in [WS-Discovery].

The Discovery Proxy role as described in [WS-Discovery] shall NOT be supported by a device or a client (an alternative Discovery Proxy role is introduced in this standard, see 7.4). A device that implements the client role ignores the interaction scheme with the Discovery Proxy as described in Clause 3 in [WS-Discovery]. Instead, this standard defines a new Discovery Proxy role that allows remote discovery. The remote discovery relies on the presence of a Discovery Proxy and a system provider that would like to offer remote discovery in the system should implement the Discovery Proxy role as specified in 7.4.

[WS-Discovery] describes the Universally Unique Identifier (UUID): URI format recommendation for endpoint references in 2.6, but this specification overrides this recommendation. Instead, the Uniform Resource Name: Universally Unique Identifier (URN:UUID) format is used [RFC 4122] (see 7.3.1).

7.2 Modes of operation

The device shall be able to operate in *two* modes:

- discoverable;
- non-discoverable.

A device in discoverable mode sends multicast Hello messages once connected to the network or sends its Status changes according to [WS-Discovery]. In addition it always listens for Probe and Resolve messages and sends responses accordingly. A device in non-discoverable shall not listen to [WS-Discovery] messages or send such messages.

The devices *default* behaviour shall be the discoverable mode. In order to thwart denial-of-service attacks, it shall be possible to set a device into non-discoverable mode through the operation defined in 0.

7.3 Discovery definitions

7.3.1 Endpoint reference

A device or an endpoint that takes the client role should use a URN:UUID [RFC 4122] as the address property of its endpoint reference.

The device or an endpoint that takes the client role shall use a stable, globally unique identifier that is constant across network interfaces as part of its endpoint reference property. The combination of an wsadis:Address and wsadis:ReferenceProperties provide a stable and globally-unique identifier.

7.3.2 Hello

7.3.2.1 Types

A device shall include the device management service port type, i.e. tds:Device, in the <d:Types> declaration.

For backward compatibility reason an ONVIF compliant device shall also include dn:NetworkVideoTransmitter in the <d:Types> declaration.

The following example shows how the type is encoded in the SOAP Hello body:

```
<d:Types>tds:Device</d:Types>.
```

The Hello message MAY include additional types.

7.3.2.2 Scopes

A device shall include the scope <d:Scopes> attribute with the scopes of the device in the Hello message.

The device scope is set by using [RFC 3986] URIs. This specification defines scope attributes as follows:

The scheme attribute: onvif

The authority attribute: www.onvif.org

This implies that all ONVIF defined scope URIs have the following format:

```
onvif://www.onvif.org/<path>
```

The device MAY have other scope URIs. These URIs are not restricted of ONVIF defined scopes.

Table 8 defines the basic capabilities and other properties of the device. Apart from these standardized parameters, it shall be possible to set any scope parameter as defined by the device owner. Scope parameters can be listed and set through the commands defined in 0. Future editions of the standard might introduce additional standardized scope parameters.

A device MAY have other scope URIs. These URIs are not restricted of ONVIF defined scopes.

Table 8 – Scope parameters

Category	Defined values	Description
type	video_encoder	A video_encoder indicates that this device is a network video encoder device. A device with network video support, shall include the video_encoder type in its scope list.
	Ptz	A ptz scope indicates that the device is a ptz device. A device with PTZ support shall include a scope entry with this value in its scope list.
	audio_encoder	The audio_encoder scope indicates that this device is an audio encoder and a device with audio encoder support shall include a scope entry with this value in its scope list.
	video_analytics	The video analytics scope indicates that this device supports video analytics as defined in Clause 17. A device with video analytics support shall include a scope entry with this value in its scope list.
	Network_Video_Transmitter	The network video transmitter scope indicates if the device is an NVT compliant device. An NVT shall include a scope entry with this value in its scope list.
	Network_Video_Decoder	The network video display scope indicates if the device is an NVD compliant device. An NVD shall include a scope entry with this value in its scope list.
	Network_Video_Storage	The network video storage scope indicates if the device is an NVS compliant device. An NVS shall include a scope entry with this value in its scope list.
	Network_Video_Analytic	The network video analytic scope indicates if the device is an NVA compliant device. An NVA shall include a scope entry with this value in its scope list.
location	Any character string or path value.	The location defines the physical location of the device. The location value might be any string describing the physical location of the device. A device shall include at least one location entry into its scope list.
hardware	Any character string or path value.	A string or path value describing the hardware of the device. A device shall include at least one hardware entry into its scope list.
name	Any character string or path value.	The searchable name of the device. A device shall include at least one name entry into its scope list.

A device shall include at least one entry of the type, location, hardware and name categories respectively in the scopes list. A device MAY include any other additional scope attributes in the scopes list.

A device might include an *arbitrary* number of scopes in its scope list. This implies that one unit might for example define *several different* location scopes. A probe is matched against *all* scopes in the list.

Example

The following example illustrates the usage of the scope value. This is *just an example*, and not at all an indication of what type of scope parameter to be part of an NVT configuration. In this example we assume that the NVT is configured with the following scopes:

```
onvif://www.onvif.org/type/Network_Video_Transmitter
```

```
onvif://www.onvif.org/type/video_encoder
```

```
onvif://www.onvif.org/type/ptz
```

```
onvif://www.onvif.org/type/audio_encoder
```

```
onvif://www.onvif.org/type/video_analytics
```

```
onvif://www.onvif.org/hardware/D1-566
onvif://www.onvif.org/location/country/china
onvif://www.onvif.org/location/city/beijing
onvif://www.onvif.org/location/building/headquarter
onvif://www.onvif.org/location/floor/R5
onvif://www.onvif.org/name/ARV-453
```

A client that probes for the device with scope `onvif://www.onvif.org` will get a match. Similarly, a probe for the device with scope:

```
onvif://www.onvif.org/location/country/china
```

will give a match. A probe with:

```
onvif://www.onvif.org/hardware/D1
```

will *not* give a match.

7.3.2.3 Addresses

A device shall include the `<d:XAddr>` element with the address(es) for the device service in the Hello message. A URI shall be provided for each protocol (http, https) and externally available IP address.

The IP addressing configuration principles for a device are defined in 5.12.2.2.1.

7.3.3 Probe and probe match

For the device probe match types, scopes and addresses definitions, see 7.3.2 Hello.

The device shall at least support the `http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc3986` scope matching rule. This scope matching definitions differs slightly from the definition in [WS-Discovery] as [RFC 2396] is replaced by [RFC 3986].

A device shall include the `<d:XAddr>` element with the addresses for the device service in a matching probe match message. The `<d:XAddr>` element will in most cases only contain one address to the management and configuration interfaces as defined in 5.1.

7.3.4 Resolve and resolve match

This standard requires end point address information to be included into Hello and Probe Match messages. In most cases, there is no need for the resolve and resolve match exchange. To be compatible with the [WS-Discovery] specification, however, a device should implement the resolve match response.

7.3.5 Bye

A device should send a one-way Bye message when it prepares to leave a network as described in WS-Discovery.

7.3.6 SOAP fault messages

If an error exists with the multicast packet, the device and client should silently discard and ignore the request. Sending an error response is not recommended due to the possibility of packet storms if many devices send an error response to the same request. For completeness, unicast packet error handling is described below.

If a device receives a unicast Probe message and it does not support the matching rule, then the device MAY choose not to send a Probe Match, and instead generate a SOAP fault bound to SOAP 1.2 as follows:

[Action] `http://schemas.xmlsoap.org/ws/2005/04/discovery/fault`

[Code] `s12:Sender`

[Subcode] `d:MatchingRuleNotSupported`

[Reason] E.g., the matching rule specified is not supported

[Detail] `<d: SupportedMatchingRules>`

`List of xs:anyURI`

`</d: SupportedMatchingRules>`

All faults arising in an extension or from the application should be generated according to SOAP 1.2 fault message protocols. After transmission of a SOAP fault message to the Sender, the fault should be notified to the application that a fault has been generated.

7.4 Remote discovery extensions

This subclause describes discovery extensions needed to cover more complex network scenarios. These extensions are not required by an ONVIF-compliant endpoints. A device that supports remote service discovery shall support the discovery extensions defined in this subclause.

The remote discovery extensions defined in this subclause can be used *together* with the ordinary multicast base WS-Discovery scheme as defined in this standard. For example, the remote discovery extensions can work in parallel with the ordinary “local” discovery.

7.4.1 Network scenarios

If the client and the device *do not* reside in the same administrative domain, it is not possible for the client to find *and* connect to the device using multicast probe. For example, if the device or the client resides in a network behind a firewall or NAT (Gateway GW) it could not connect to a multicast probe. Other methods, then, are needed and the specification uses four different scenarios:

- 1) the device resides in one administrative domain (private) and the client resides in a public network, see Figure 8;
- 2) the device resides in a public network and the client resides in one administrative domain (private), see Figure 9;
- 3) the device resides in one administrative domain (private) and the client resides in *another* administrative domain (private), see Figure 10;
- 4) both the device and the client reside in a public network, see Figure 11.

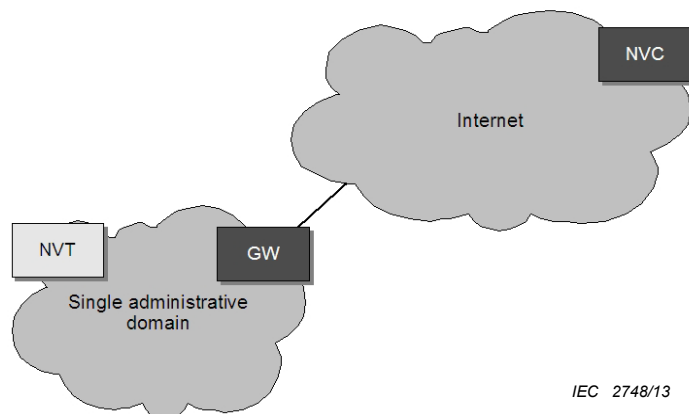


Figure 8 – A device, for example a NVT, in an administrative domain (private) and the client (NVC) in a public network

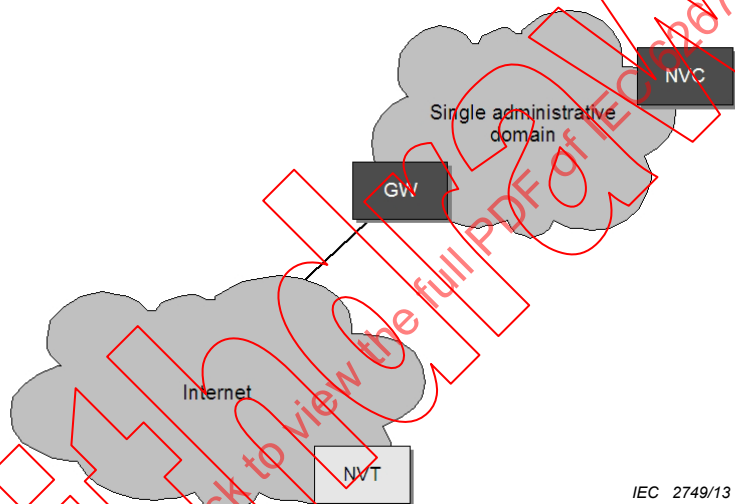


Figure 9 – A device, for example a NVT, in public network and the client (NVC) in an administrative domain (private)

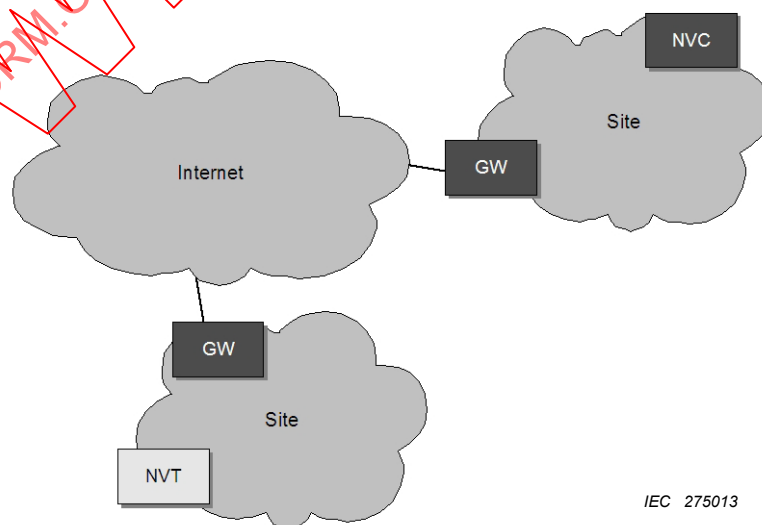


Figure 10 – A device, for example a NVT, in an administrative domain (private) and the client (NVC) in another administrative domain (private)

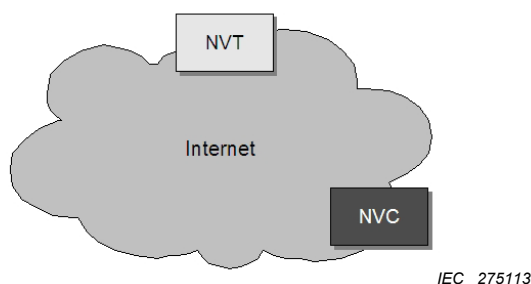


Figure 11 – Both a device, for example a NVT, and the client (NVC) in a public network

The [WS-Discovery] specification introduces a *Discovery Proxy (DP)* to solve some of these scenarios. However the [WS-Discovery] specification does not have support for all the network scenarios introduced in this specification. This specification defines a DP that enables “plug and play” also for the more complex network scenarios we have listed above. This DP is *not* compliant with [WS-Discovery] specification.

7.4.2 Discover proxy (DP)

A network administrator configuring a network for an NVT wide area network spanning several administrative domains, needs to introduce a DP endpoint into the system. The DP performs the following tasks:

- 1) listen for device hello messages and responds to these as defined in 7.4.3;
- 2) responds to probe queries on behalf of registered devices from clients.

The DP may reside in the same administrative domain as the device. In order to support network scenarios where the client and device reside in different domains without multicast connectivity, place the DP in a publicly available network so that device and client endpoints can access it. It shall be possible for the device to find the network address of its “home DP” in order to allow the announcement of its presence with a Hello message *directly* sent to its home DP. According to this standard, the home DP network address can be obtained in the following ways:

- 1) direct address configuration;
- 2) DP discovery using DNS Service record (SRV) lookup.

The device tries to connect to a home DP once it gets network connectivity or when the home DP network address is changed using either of these methods.

It shall be possible to enable/disable the device remote discovery registration. A device supporting remote discovery shall implement the remote Hello disable/enable operation as defined in 0.

A device that is not configured with a home DP address or a device with remote Hello disabled shall NOT send a remote Hello as defined in 7.4.3.

7.4.2.1 Direct DP address configuration

This standard introduces a device management command for home DP address configuration over the network interface, see 0 and 0.

A device that supports remote discovery MAY also offer local configuration of the home DP address. Such configurations are done through a device local interface of choice such as a serial port or USB interface. Such local configuration is *outside* the scope of this standard.

7.4.2.2 DNS service record lookup

If a device has remote discovery enabled but *lacks* remote DP address configuration, it shall try to make a DNS SRV lookup for the home DP. The following record name and protocol definition [RFC 2782] shall be used:

_onvifdiscover._tcp

In order to avoid a DNS SRV lookup by the device, a DP address shall be configured using direct address configuration before enabling remote discovery.

In order for devices to make a successful DP lookup for other devices, an administrator shall enter the DP address, port and priority into the DNS using SRVs. One or several enrolment servers need to be present. The exact number will depend on the load of the system and is *outside the scope* of this standard.

7.4.3 Remote Hello and Probe behaviour

The local discovery pattern as defined in [WS-Discovery] does not work for the remote discovery scenarios. If the device resides behind a NAT/Firewall, like the scenarios shown in Figure 8 or Figure 10, a unicast Probe from the DP will not automatically reach the device if the device does not return a public network address. Furthermore, if the device resides behind a firewall, the device following Probe Match unicast might not reach back to the DP. The specification defines a slightly different communication pattern for the remote discovery to solve this problem.

A device configured for remote Hello sends, in addition to the multicast Hello when it joins a network or its metadata changes, a remote Hello message to its home DP. This message is sent as a Web Services request operation from the device to the DP using the HTTP binding as defined in [C.12]. The remote Hello shall include its scope list in the Hello message.

Once the home DP receives a Hello message from any device, it responds with a Hello response message confirming the device registration through the hello message. Similarly, when a device prepares for leaving a network it should send a Bye request to the remote DP. The DP acknowledges the Bye request through a Bye response message. The DP Hello, Hello response, Bye and Bye response are provided as a DP service, see [C.12] for the WSDL definitions. Using these extensions, the discovery messages can reach the desired endpoints as shown in Figure 12.

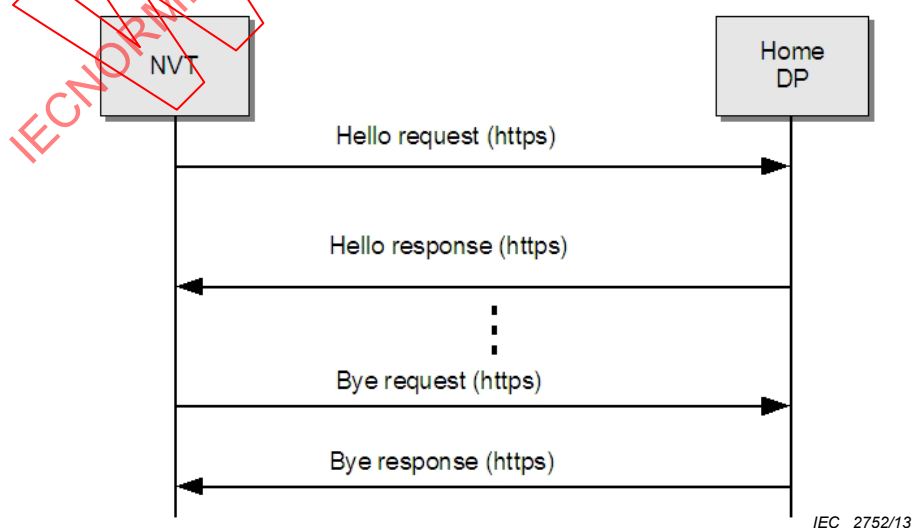


Figure 12 – Remote discovery message exchange pattern between a device (for example a NVT) and a Home DP

7.4.4 Client behaviour

7.4.4.1 General

For the remote discovery scenarios, the client needs to send probe messages to the home DP. The client then needs to be configured such that it can directly connect to the home DP.

7.4.4.2 NVC home DP configuration

The client can be configured to directly probe for new devices through the home DP. In this case the home DP discovery service address shall be pre-configured into the client. The exact means of this configuration is outside the scope of this standard.

An client configured for remote discovery sends probe requests directly to its home DP. The probe message is sent as a Web Services request operation from the client to the DP using the http binding (see C.12).

Once the home DP receives a Probe message from any client, it responds with corresponding Probe Match message according to the normal WS-Discovery message exchange pattern, see the sequence chart in Figure 13.

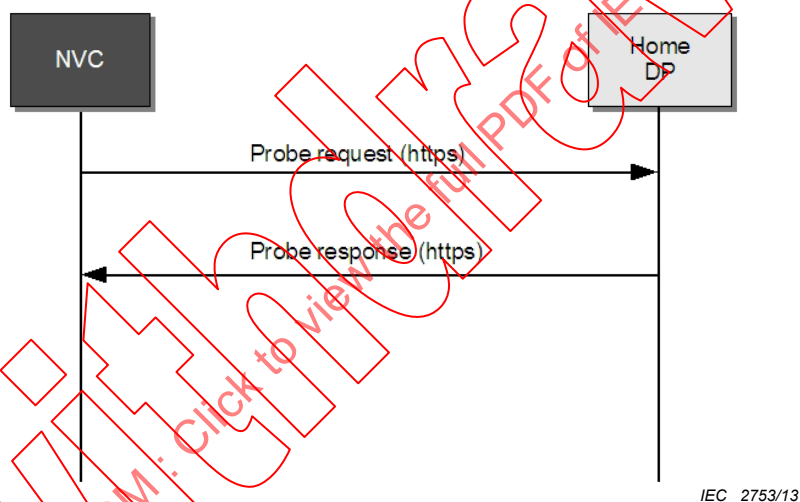


Figure 13 – Message sequence for clients (NVC) pre-configured with home DP address

7.4.5 Security

7.4.5.1 Local discovery

Security and discovery can be viewed as contradictory goals. While the main idea behind a discovery protocol is to announce the presence of a service, it is hard to *exclude* other endpoints from access to the service announcements. WS-Discovery does not provide any extra access to services (if the other security mechanism specified in this specification are used), even on the same LAN; it merely announces their existence. Furthermore, local discovery works only within multicast reach range. Thus, the main security impact of WS-Discovery is the risk of denial of service attacks on devices or privacy issues if it is important to hide the presence of devices in the network. The risk of the latter two problems will very much depend on the device deployment environment. In order to reduce these threats, this standard has introduced the two different discovery modes, see 7.2. This always gives the possibility for the client to switch off the device discovery function in the device. In non-discoverable mode, a device will never announce its presence with Hello messages or respond to any Probe or Resolve requests.

7.4.5.2 Remote discovery

In the remote network scenario, the DP resides on the Internet and is vulnerable. Extra security measurements, then, shall be taken to protect the DP from attacks. The remote Hello and Probe and Probe Match messages, as defined in 7.4.3, shall be sent over HTTPS. This transport will *not* prevent denial of service attacks, but it can protect it from illegal device registrations if client authentication is used. If protection of denial of service is a major concern, other measurements need to be taken, which is outside the scope of the current standard.

Before registering a device in the device data base the DP should authenticate the device to make sure that it is a “legal” device that announces its presence, for example by using client certificates. Client certificate provisioning is outside the scope of the current standard.

The client to DP remote Probe and Probe Match messages shall be sent over HTTPS. The DP shall authenticate the NVC before responding to a Probe request. This can be done using TLS client certificates or any other suitable client authentication mechanism.

8 Device management

The Device Service is divided into five different categories: capabilities, network, system, I/O and security commands. This set of commands can be used to get information about the device capabilities and configurations or to set device configurations. A device shall support the device management service as specified in C.4. A basic set of operations are required for the device management service, other operations are recommended or optional to support. The detailed requirements are listed under the command descriptions.

8.1 Capabilities

8.1.1 Get WSDL URL

It is possible for an endpoint to request a URL that can be used to retrieve the *complete* schema and WSDL definitions of a device. The command gives in return a URL entry point where all the necessary product specific WSDL and schema definitions can be retrieved. The device shall provide a URL for WSDL and schema download through the GetWsdUrl command (see Table 9).

Table 9 – Get WSDL URL command

GetWsdUrl		Request-Response
Message name	Description	
GetWsdUrlRequest	<i>This is an empty message.</i>	
GetWsdUrlResponse	<i>The requested URL.</i> xs:anyURI WsdUrl [1][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.1.2 Capability exchange

Any endpoint can ask for the capabilities of a device using the capability exchange request response operation. The device shall indicate all its ONVIF compliant capabilities through the GetCapabilities command (see Table 10).

The capability list includes references to the addresses (XAddr) of the service implementing the interface operations in the category.

Table 11 describes how to interpret the indicated capability. Apart from the addresses, the capabilities only reflect optional functions in this standard.

Table 10 – Get Capabilities command

GetCapabilities		Request-Response
Message name	Description	
GetCapabilitiesRequest	<i>This message contains a request for device capabilities. The client can either ask for all capabilities or just the capabilities for a particular service category. If no Category is specified the device shall return all capabilities.</i> tt:CapabilityCategory Category [0][unbounded]	
GetCapabilitiesResponse	<i>The capability response message contains the requested device capabilities using a hierarchical XML capability structure.</i> tt:Capabilities Capabilities [1][1]	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:NoSuchService	<i>The requested WSDL service category is not supported by the device.</i>	

Table 11 – The capabilities in the GetCapabilities command

Category	Capability	Description
Analytics	XAddr	The address to the analytics service. If this field is empty the device supports analytics but not the rules or module interfaces.
	RuleSupport	Indication if the device supports rules interface and rules syntax as specified in 17.2.
	AnalyticsModuleSupport	Indication if the device supports the scene analytics module interface as specified in 0.
Device	XAddr	The address to the device service.
Device – Network	IPFilter	Indication if the device supports IP filtering control using the commands in 0, 0, 0 and 0.
	ZeroConfiguration	Indication if the device supports zero configuration according to the commands in 0 and 0.
	IPVersion6	Indication if the device supports IP version 6.
	DynDNS	Indication if the device supports Dynamic DNS configuration according to 0 and 0.
	Dot11Configuration	Indication if the device supports IEEE 802.11 configuration as specified in 0.
Device – System	DiscoveryResolve	Indication if the device responses to resolve requests as described in 7.3.4.
	DiscoveryBye	Indication if the device sends bye messages as described in 7.3.5.
	RemoteDiscovery	Indication if the device supports remote discovery support as specified in 7.4.
	SupportedVersions	List of the device supported ONVIF specification versions.
	SystemBackup	Indication if the device supports system backup and restore as specified in 0 and 0.
	FirmwareUpgrade	Indication if the device supports firmware upgrade as specified in 0.
	SystemLogging	Indication if the device supports system log retrieval as specified in 0.
	HttpSystemBackup	Indication if the device supports system backup and restore using HTTP GET and POST.
	HttpFirmwareUpgrade	Indication if the device supports firmware upgrade using HTTP POST.
	HTTPSystemLogging	Indication if the device supports retrieval of system log using HTTP Get, see 8.3.2.
	HTTPSupportInformation	Indication if the device supports retrieval of support information using HTTP Get, see 8.3.2.
Device – IO	InputConnectors	The number of input connectors.
	RelayOutputs	The number of relay outputs.
	Auxiliary	Indication of support for auxiliary service along with list of supported auxiliary commands
Device – Security	TLS1.0	Support of TLS 1.0.
	TLS1.1	Support of TLS 1.1.
	TLS1.2	Support of TLS 1.2.
	OnboardKeyGeneration	Indication if the device supports onboard key generation and creation of self-signed certificates as specified in 8.4.8.
	AccessPolicyConfig	Indication if the device supports retrieving and loading device access control policy according to 8.4.1 and 0.

Category	Capability	Description
	X.509Token	Indication if the device supports the WS-Security X.509 token [WS-X.509Token].
	SAMLTToken	Indication if the device supports the WS-Security SAML token [WS-SAMLTToken].
	KerberosToken	Indication if the device supports the WS-Security Kerberos token [WS-KerberosToken].
	RELTToken	Indication if the device supports the WS-Security REL token [WS-RELTToken].
	Dot1X	Indication if the device supports IEEE 802.1X port-based network authentication
	SupportedEAPMethod	List of supported EAP Method types. The numbers correspond to the IANA [EAP-Registry].
	RemoteUserHandling	Indication if device supports remote user handling and the corresponding methods defined in 0 and 8.4.22.
Event	XAddr	The address to the event service.
	WSSubscriptionPolicySupport	Indication if the device supports the WS Subscription policy according to 15.1.2
	WSPullPointSupport	Indication if the device supports the WS Pull Point according to 15.1.2
	WSPausableSubscription-ManagerInterfaceSupport	Indication if the device supports the WS Pausable Subscription Manager Interface according to 15.1.2
Imaging	XAddr	The address to the imaging service
Media	XAddr	The address to the media service.
Media – streaming	RTPMulticast	Indication of support of UDP multicasting as described in 12.1.1.1.
	RTP_TCP	Indication if the device supports RTP over TCP, see 12.1.1.2.
	RTP_RTSP_TCP	Indication if the device supports RTP/RTSP/TCP transport, see 12.1.1.3.
Media - profile	MaximumNumberOfProfiles	The maximum Number of MediaProfiles the device supports.
PTZ	XAddr	The address to the PTZ service.
Receiver	XAddr	The address to the receiver service.
	RTP_Multicast	Indication if the device supports receiving of RTP Multicast.
	RTP_TCP	Indication if the device supports receiving of RTP over TCP.
	RTP_RTSP_TCP	Indication if the device supports receiving of RTP over RTSP over TCP
	SupportedReceivers	The maximum number of receivers the device supports.
	MaximumRTSPURILength	The maximum length allowed for RTSP URIs.
Recording	XAddr	The address to the recording control service.
	DynamicRecordings	Indication if the device supports dynamic creation and deletion of recordings, see 19.4 and 19.5.
	DynamicTracks	Indication if the device supports dynamic creation and deletion of tracks, see clause 19.9 and 19.10.
	DeleteData	Indication if the device supports explicit deletion of data, see 19.5.
Search	XAddr	The address to the recording search service.
	MetadataSearch	Indication if the device supports generic search of recorded metadata, see 20.13 and 20.14.

Category	Capability	Description
Replay	XAddr	The address to the replay service.
Analytics Device	XAddr	The address to the analytics device service of the device.
Display	XAddr	The address to the display service.
Display - layout	FixedLayout	Indication if the device has a certain set of predefined layouts.
Device IO	XAddr	The address to the device IO service.
	VideoSources	The number of video inputs
	VideoOutputs	The number of video outputs
	AudioSources	The number of audio inputs
	AudioOutputs	The number of audio outputs
	RelayOutputs	The number of relay outputs

8.2 Network

8.2.1 Get hostname

This operation is used by an endpoint to get the hostname from a device. The device shall return its hostname configurations through the GetHostname command (see Table 12).

Table 12 – GetHostname command

GetHostname		Request-Response
Message name	Description	
GetHostnameRequest	<i>This is an empty message.</i>	
GetHostnameResponse	<i>This message contains:</i> <i>“FromDHCP”: True if the hostname is obtained via DHCP</i> <i>“Name”: The host name. In case of DHCP the host name has been obtained from the DHCP server.</i> xs:boolean FromDHCP [1][1] xs:token Name [0][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.2 Set hostname

This operation sets the hostname on a device. It shall be possible to set the device hostname configurations through the SetHostname command (see Table 13). Attention: a call to SetDNS may result in overriding a previously set hostname.

Table 13 – SetHostname command

SetHostname		Request-Response
Message name	Description	
SetHostnameRequest	<i>This message contains:</i> <i>“Name”: The host name.</i> xs:token Name [1][1]	
SetHostnameResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidHostname	<i>The requested hostname cannot be accepted by the device.</i>	

8.2.3 Get DNS settings

This operation gets the DNS settings from a device. The device shall return its DNS configurations through the GetDNS command (see Table 14).

Table 14 – GetDNS command

GetDNS		Request-Response
Message name	Description	
GetDNSRequest	<i>This is an empty message.</i>	
GetDNSResponse	<i>This message contains:</i> <i>“FromDHCP”: True if the DNS servers are obtained via DHCP.</i> <i>“SearchDomain”: The domain(s) to search if the hostname is not fully qualified.</i> <i>“DNSFromDHCP”: A list of DNS servers obtained via DHCP in case FromDHCP is equal to true. This means that the resolved addresses in the field DNSFromDHCP are coming from DHCP and describes the configuration status.</i> <i>“DNSManual”: A list of manually given DNS servers</i> xs:boolean FromDHCP [1][1] xs:token SearchDomain [0][unbounded] tt:IPAddress DNSFromDHCP [0][unbounded] tt:IPAddress DNSManual [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.4 Set DNS settings

This operation sets the DNS settings on a device. It shall be possible to set the device DNS configurations through the SetDNS command (see Table 15).

Table 15 – Set DNS command

SetDNS		Request-Response
Message name	Description	
SetDNSRequest	<i>This message contains:</i> <i>“FromDHCP”: True if the DNS servers are obtained via DHCP</i> <i>“SearchDomain”: The domain(s) to search if the hostname is not fully qualified.</i> <i>“DNSManual”: A list of manually given DNS servers</i> xs:boolean FromDHCP [1][1] xs:token SearchDomain [0][unbounded] tt:IPAddress DNSManual [0][unbounded]	
SetDNSResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>The suggested IPv6 address is invalid.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>The suggested IPv4 address is invalid.</i>	

8.2.5 Get NTP settings

This operation gets the NTP settings from a device. If the device supports NTP, it shall be possible to get the NTP server settings through the GetNTP command (see Table 16).

Table 16 – GetNTP command

GetNTP		Request-Response
Message name	Description	
GetNTPRequest	<i>This is an empty message.</i>	
GetNTPResponse	<i>This message contains:</i> <i>“FromDHCP”: True if the NTP servers are obtained via DHCP.</i> <i>“NTPFromDHCP”: A list of NTP servers obtained via DHCP in case FromDHCP is equal to true. This means that the NTP server addresses in the field NTPFromDHCP are coming from DHCP and describes the current configuration status.</i> <i>“NTPManual”: A list of manually given NTP servers</i> xs:boolean FromDHCP [1][1] tt:NetworkHost NTPFromDHCP [0][unbounded] tt:NetworkHost NTPManual [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.6 Set NTP settings

This operation sets the NTP settings on a device. If the device supports NTP, it shall be possible to set the NTP server settings through the SetNTP command (see Table 17).

Table 17 – SetNTP command

SetNTP		Request-Response
Message name	Description	
SetNTPRequest	<i>This message contains:</i> <i>“FromDHCP”: True if the NTP servers are obtained via DHCP.</i> <i>“NTPManual”: A list of manually given NTP servers when they not are obtained via DHCP.</i> xs:boolean FromDHCP [1][1] tt:NetworkHost NTPManual [0][unbounded]	
SetNTPResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>The suggested IPv4 address is invalid.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>The suggested IPv6 address is invalid.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDnsName	<i>The suggested NTP server name is invalid.</i>	

8.2.7 Get dynamic DNS settings

This operation gets the dynamic DNS settings from a device. If the device supports dynamic DNS as specified in [RFC 2136] and [RFC 4702], it shall be possible to get the type, name and TTL through the GetDynamicDNS command (see Table 18).

Table 18 – GetDynamicDNS command

GetDynamicDNS		Request-Response
Message name	Description	
GetDynamicDNSRequest	<i>This is an empty message.</i>	
GetDynamicDNSResponse	<i>This message contains:</i> <i>“Type”: The type of update. There are three possible types: the device desires no update (NoUpdate), the device wants the DHCP server to update (ServerUpdates) and the device does the update itself (ClientUpdates).</i> <i>“Name”: The DNS name in case of the device does the update.</i> <i>“TTL”: Time to live.</i> <i>tt:DynamicDNSType Type [1][1]</i> <i>tt:DNSName Name [0][1]</i> <i>xs:duration TTL [0][1]</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.8 Set dynamic DNS settings

This operation sets the dynamic DNS settings on a device. If the device supports dynamic DNS as specified in [RFC 2136] and [RFC 4702], it shall be possible to set the type, name and TTL through the SetDynamicDNS command (see Table 19).

Table 19 – SetDynamicDNS command

SetDynamicDNS		Request-Response
Message name	Description	
SetDynamicDNSRequest	<i>This message contains:</i> <i>“Type”: The type of update. There are three possible types: the device desires no update (NoUpdate), the device wants the DHCP server to update (ServerUpdates) and the device does the update itself (ClientUpdates).</i> <i>“Name”: The DNS name in case of the device does the update.</i> <i>“TTL”: Time to live.</i> <i>tt:DynamicDNSType Type [1][1]</i> <i>tt:DNSName Name [0][1]</i> <i>xs:duration TTL [0][1]</i>	
SetDynamicDNSResponse	<i>This is an empty message.</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.9 Get network interface configuration

This operation gets the network interface configuration from a device. The device shall support return of network interface configuration settings as defined by the NetworkInterface type through the GetNetworkInterfaces command (see Table 20).

Table 20 – GetNetworkInterfaces command

GetNetworkInterfaces		Request-Response
Message name	Description	
GetNetworkInterfacesRequest	<i>This is an empty message.</i>	
GetNetworkInterfacesResponse	<i>This message contains an array of device network interfaces.</i> <i>tt:NetworkInterface NetworkInterfaces [0][unbounded]</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.10 Set network interface configuration

This operation sets the network interface configuration on a device. The device shall support network configuration of supported network interfaces through the SetNetworkInterfaces command (see Table 21).

For interoperability with a client unaware of the IEEE 802.11 extension a device shall retain its IEEE 802.11 configuration if the IEEE 802.11 configuration element is not present in the request.

Table 21 – SetNetworkInterfaces command

SetNetworkInterfaces		Request-Response
Message name	Description	
SetNetworkInterfacesRequest	<i>This message contains:</i> <i>“InterfaceToken”: The token of the network interface to operate on.</i> <i>“NetworkInterface”: The network interface to configure.</i> tt:ReferenceToken InterfaceToken [1][1] tt:NetworkInterfaceSetConfiguration NetworkInterface [1][1]	
SetNetworkInterfacesResponse	<i>This message contains:</i> <i>“RebootNeeded”: An indication if a reboot is needed in case of changes in the network settings.</i> xs:boolean RebootNeeded [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	The supplied network interface token does not exist.	
env:Sender ter:InvalidArgVal ter:InvalidMtuValue	The MTU value is invalid.	
env:Sender ter:InvalidArgVal ter:InvalidInterfaceSpeed	The suggested speed is not supported.	
env:Sender ter:InvalidArgVal ter:InvalidInterfaceType	The suggested network interface type is not supported.	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	The suggested IPv4 address is invalid.	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	The suggested IPv6 address is invalid.	
env:Receiver ter:ActionNotSupported ter:InvalidDot11	IEEE 802.11 Configuration is not supported.	

env:Sender ter:InvalidArgVal ter:InvalidSecurityMode	<i>The selected security mode is not supported.</i>
env:Sender ter:InvalidArgVal ter:InvalidStationMode	<i>The selected station mode is not supported.</i>
env:Sender ter:InvalidArgVal ter:MissingDot11	<i>IEEE 802.11 value is missing in the security configuration.</i>
env:Sender ter:InvalidArgVal ter:MissingPSK	<i>PSK value is missing in security configuration.</i>
env:Sender ter:InvalidArgVal ter:MissingDot1X	<i>IEEE 802.1X value in security configuration is missing or none existing.</i>
env:Sender ter:InvalidArgVal ter:IncompatibleDot1X	<i>IEEE 802.1X value in security configuration is incompatible with the network interface.</i>

8.2.11 Get network protocols

This operation gets defined network protocols from a device. The device shall support the GetNetworkProtocols command returning configured network protocols (see Table 22).

Table 22 – GetNetworkProtocols command

GetNetworkProtocols		Request-Response
Message name	Description	
GetNetworkProtocolsRequest	<i>This is an empty message.</i>	
GetNetworkProtocols-Response	<i>This message returns an array of defined protocols supported by the device. There are three protocols defined, HTTP, HTTPS and RTSP. The following parameters can be retrieved for each protocol:</i> <i>Port</i> <i>Enable/disable</i> tt:NetworkProtocol NetworkProtocols [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.12 Set network protocols

This operation configures defined network protocols on a device. The device shall support configuration of defined network protocols through the SetNetworkProtocols command (see Table 23).

Table 23 – SetNetworkProtocols command

SetNetworkProtocols		Request-Response
Message name	Description	
SetNetworkProtocolsRequest	<i>This message configures one or more defined network protocols supported by the device. There are currently three protocols defined, HTTP, HTTPS and RTSP. The following parameters can be set for each protocol:</i> <i>Port</i> <i>Enable/disable</i> tt:NetworkProtocol NetworkProtocols [1][unbounded]	
SetNetworkProtocols-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:ServiceNotSupported	<i>The supplied network service is not supported.</i>	

8.2.13 Get default gateway

This operation gets the default gateway settings from a device. The device shall support the GetNetworkDefaultGateway command returning configured default gateway *address(es)* (see Table 24).

Table 24 – GetNetworkDefaultGateway command

GetNetworkDefaultGateway		Request-Response
Message name	Description	
GetNetworkDefaultGateway-Request	<i>This is an empty message.</i>	
GetNetworkDefaultGateway-Response	<i>This message contains:</i> <i>“IPv4Address”: The default IPv4 gateway address(es).</i> <i>“IPv6Address”: The default IPv6 gateway address(es).</i> tt:IPv4Address IPv4Address [0][unbounded] tt:IPv6Address IPv6Address [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.14 Set default gateway

This operation sets the default gateway settings on a device. The device shall support configuration of default gateway through the SetNetworkDefaultGateway command (see Table 25).

Table 25 – SetNetworkDefaultGateway command

SetNetworkDefaultGateway		Request-Response
Message name	Description	
SetNetworkDefaultGateway-Request	<i>This message contains:</i> <i>“IPv4Address”: The default IPv4 gateway address(es).</i> <i>“IPv6Address”: The default IPv6 gateway address(es).</i> tt:IPv4Address IPv4Address [0][unbounded] tt:IPv6Address IPv6Address [0][unbounded]	
SetNetworkDefaultGateway-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidGatewayAddress	<i>The supplied gateway address was invalid.</i>	

8.2.15 Get zero configuration

This operation gets the zero-configuration from a device. If the device supports dynamic IP configuration according to [RFC 3927], it shall support the return of IPv4 zero configuration address and status through the GetZeroConfiguration command (see Table 26).

Table 26 – GetZeroConfiguration command

GetZeroConfiguration		Request-Response
Message name	Description	
GetZeroConfigurationRequest	<i>This is an empty message.</i>	
GetZeroConfigurationResponse	<i>This message contains:</i> <i>“InterfaceToken”: The token of the network interface</i> <i>“Enabled”: If zero configuration is enabled or not.</i> <i>“Addresses”: The IPv4 zero configuration address(es).</i> tt:ReferenceToken InterfaceToken [1][1] xs:boolean Enabled [1][1] tt:IPv4Addresses Address [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.16 Set zero configuration

This operation sets the zero-configuration on the device. If the device supports dynamic IP configuration according to [RFC 3927], it shall support the configuration of IPv4 zero configuration address and status through the SetZeroConfiguration command (see Table 27).

Table 27 – SetZeroConfiguration command

SetZeroConfiguration		Request-Response
Message name	Description	
SetZeroConfigurationRequest	<i>This message contains:</i> <i>“InterfaceToken”: The token of the network interface to operate on.</i> <i>“Enabled”: If zero configuration is enabled or not.</i> tt:ReferenceToken InterfaceToken [1][1] xs:boolean Enabled [1][1]	
SetZeroConfigurationResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	<i>The supplied network interface token does not exists</i>	

8.2.17 Get IP address filter

This operation gets the IP address filter settings from a device. If the device supports device access control based on IP filtering rules (denied or accepted ranges of IP addresses), the device shall support the GetIPAddressFilter command (see Table 28).

Table 28 – GetIPAddressFilter command

GetIPAddressFilter		Request-Response
Message name	Description	
GetIPAddressFilterRequest	<i>This is an empty message.</i>	
GetIPAddressFilterResponse	<i>This message contains:</i> <i>“Type”: Sets if the filter should deny or allow access.</i> <i>“IPv4Address”: The IPv4 filter address(es)</i> <i>“IPv6Address”: The IPv6 filter address(es)</i> tt:IPAddressFilterType Type [1][1] tt:PrefixedIPv4Address IPv4Address [0][unbounded] tt:PrefixedIPv6Address IPv6Address [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.2.18 Set IP address filter

This operation sets the IP address filter settings on a device. If the device supports device access control based on IP filtering rules (denied or accepted ranges of IP addresses), the device shall support configuration of IP filtering rules through the SetIPAddressFilter command (see Table 29).

Table 29 – SetIPAddressFilter command

SetIPAddressFilter		Request-Response
Message name	Description	
SetIPAddressFilterRequest	<i>This message contains:</i> <i>“Type”:</i> Sets if the filter should deny or allow access. <i>“IPv4Address”:</i> The IPv4 filter address(es) <i>“IPv6Address”:</i> The IPv6 filter address(es) tt:IPAddressFilterType Type [1][1] tt:PrefixedIPv4Address IPv4Address [0][unbounded] tt:PrefixedIPv6Address IPv6Address [0][unbounded]	
SetIPAddressFilterResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>The suggested IPv6 address is invalid.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>The suggested IPv4 address is invalid.</i>	

8.2.19 Add an IP filter address

This operation adds an IP filter address to a device. If the device supports device access control based on IP filtering rules (denied or accepted ranges of IP addresses), the device shall support adding of IP filtering addresses through the AddIPAddressFilter command (see Table 30).

Table 30 – AddIPAddressFilter command

AddIPAddressFilter		Request-Response
Message name	Description	
AddIPAddressFilterRequest	<i>This message contains:</i> <i>“IPv4Address”: The IPv4 filter address(es)</i> <i>“IPv6Address”: The IPv6 filter address(es)</i> tt:PrefixedIPv4Address IPv4Address [0][unbounded] tt:PrefixedIPv6Address IPv6Address [0][unbounded]	
AddIPAddressFilterResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:IPFilterListIsFull	<i>It is not possible to add more IP filters since the IP filter list is full.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>The suggested IPv6 address is invalid.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>The suggested IPv4 address is invalid.</i>	

8.2.20 Remove an IP filter address

This operation deletes an IP filter address from a device. If the device supports device access control based on IP filtering rules (denied or accepted ranges of IP addresses), the device shall support deletion of IP filtering addresses through the RemoveIPAddressFilter command (see Table 31).

Table 31 – RemoveIPAddressFilter command

RemoveIPAddressFilter		Request-Response
Message name	Description	
RemoveIPAddressFilter-Request	<i>This message contains:</i> “IPv4Address”: The IPv4 filter address(es) “IPv6Address”: The IPv6 filter address(es) tt:PrefixedIPv4Address IPv4Address [0][unbounded] tt:PrefixedIPv6Address IPv6Address [0][unbounded]	
RemoveIPAddressFilter-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>The suggested IPv6 address is invalid.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>The suggested IPv4 address is invalid.</i>	
env:Sender ter:InvalidArgVal ter:NoIPv6Address	<i>The IPv6 address to be removed does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoIPv4Address	<i>The IPv4 address to be removed does not exist.</i>	

8.2.21 IEEE 802.11 configuration

Requirements in this subclause and its subclauses are only valid for a device with IEEE 802.11 support. In this section and subsections the term “the device” is used to indicate a device with IEEE 802.11 support.

The device shall support IEEE 802.11 configuration and shall as a response to the GetNetworkInterfaces method return ieee80211 (71) as the IANA-IfTypes for the 802.11 interface(s).

A device shall not return any link element in the GetNetworkInterfaces reply and it shall ignore any Link element in the SetNetworkInterfaces request.

The device should support that each IEEE 802.11 network interface can have more than one alternative IEEE 802.11 configurations attached to it.

IEEE 802.11 configuration is supported through an optional IEEE 802.11 configuration element in the get and set network configuration element. The following information is handled:

- SSID;

- station mode;
- multiple wireless network configuration;
- security configuration.

The following operations are used to help manage the wireless configuration:

- get IEEE 802.11 capabilities;
- get IEEE 802.11 status;
- scan available IEEE 802.11 networks.

8.2.21.1 SSID

The device shall support configuration of the SSID.

8.2.21.2 Station mode

The device shall support the infrastructure station mode.

The device MAY support the ad-hoc network station mode. The actual configuration needed for ad-hoc network station mode, including manual configuration of the channel number, is outside the scope of this standard. But to allow for devices that support ad-hoc network station modes, the specification allows for selecting (and reporting) this mode.

8.2.21.3 Multiple wireless network configuration

Each IEEE 802.11 configuration shall be identified with an alias (identifier). The alias shall be unique within a network interface configuration. The client shall supply the alias in the SetNetworkInterfaces request. If the client wants to update an existing wireless configuration the same alias shall be used. A wireless configuration, including the alias, shall only exist while it is part of a network interface configuration.

For the device to be able to prioritize between multiple alternative IEEE 802.11 configurations an optional priority value can be used, a higher value means a higher priority. If the priority value is missing from the configuration the lowest priority shall be assumed. If several wireless configurations have the same priority value the order between those configurations is undefined.

The actual algorithm used by the device to enable an IEEE 802.11 network from the prioritized list of IEEE 802.11 configurations is outside the scope of this standard.

8.2.21.4 Security configuration

The security configuration contains the chosen security mode and the configuration needed for that mode. The following security modes are supported:

- None
- PSK (Pre Shared Key) (WPA- and WPA2-Personal)
- IEEE 802.1X-2004 (WPA- and WPA2-Enterprise)

Configuration of WEP security mode is outside the scope of this standard but to allow for devices that support WEP security mode this standard allows for selecting (and reporting) this mode.

For data confidentiality and integrity the device shall, in accordance with the [IEEE 802.11-2007] specification, support the CCMP algorithm and the device MAY support the TKIP algorithm.

The algorithm can either be manually (CCMP, TKIP) or automatically (Any) selected. In manual selected mode the same algorithm shall be used for both the pairwise and group cipher. To be able to support other algorithms an “Extended” value is available.

The device shall support both the manually and the automatically selected mode.

8.2.21.4.1 None mode

The device shall support the “None” security mode.

8.2.21.4.2 PSK mode

The device shall support the PSK security mode.

To minimise the risk for compromising the PSK the device should NOT transmit any PSK to a client, furthermore it shall NOT return the PSK in a response to a GetNetworkInterfaces operation call.

For adding a wireless configuration with the PSK security mode the following rules applies:

- a client shall include a PSK value in the SetNetworkInterfaces request;
- the device shall check so that a PSK value was supplied, if not the device shall return an error.

For updating wireless configuration with the PSK security mode the following rules applies:

- if the client wants to retain the PSK value it should NOT include the PSK value in the SetNetworkInterfaces request;
- the device receiving a SetNetworkInterfaces request without a PSK value shall retain its PSK value.

The [IEEE 802.11-2007] standard states that the PSK should be distributed to the STA with some out-of-band method. In ONVIF the security policy shall make sure that the PSK is sufficiently protected.

8.2.21.4.3 IEEE 802.1X-2004 Mode

The device should support the IEEE 802.1X security mode. For more detailed requirements about the IEEE 802.1X-2004 security mode see [IEEE 802.1X configuration].

8.2.21.5 GetDot11 capabilities

This operation returns the IEEE 802.11 capabilities, see Table 32 and Table 33. The device shall support this operation.

Table 32 – GetDot11Capabilities

GetIEEE802.11Capabilities		Request-Response
Message name	Description	
GetDot11Capabilities-Request	<i>This is an empty message</i>	
GetDot11Capabilities-Response	<i>tt:Dot11Capabilities Capabilities [1][1]</i>	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:InvalidDot11	<i>IEEE 802.11 configuration is not supported.</i>	

Table 33 – IEEE 802.11 capabilities

Capability	Description
TKIP	Indication if the device supports the TKIP algorithm.
ScanAvailableNetworks	Indication if the device supports the Scan Available IEEE 802.11 Networks operation.
MultipleConfiguration	Indication if the device supports multiple alternative IEEE 802.11 configurations.
AdHocStationMode	Indication if the device supports the Ad-Hoc station mode.
WEP	Indication if the device supports the WEP security mode.

8.2.21.6 Get IEEE 802.11 Status

This operation returns the status of a wireless network interface. The device shall support this command (see Table 34). The following status can be returned:

- SSID (shall);
- BSSID (should);
- pair cipher (should);
- group cipher (should);
- signal strength (should);
- alias of active wireless configuration (shall).

Table 34 – GetDot11Status

GetDot11Status		Request-Response
Message name	Description	
GetDot11StatusRequest	<i>tt:ReferenceToken InterfaceToken [1][1]</i>	
GetDot11StatusResponse	<i>tt:Dot11Status Status [1][1]</i>	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:InvalidDot11	<i>IEEE 802.11 configuration is not supported.</i>	
env:Sender ter:InvalidArgVal ter:NotDot11	<i>The interface is not an IEEE 802.11 interface.</i>	
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	<i>The supplied network interface token does not exist.</i>	
env:Receiver ter:Action ter:NotConnectedDot11	<i>IEEE 802.11 network is not connected.</i>	

8.2.21.7 Scan Available IEEE 802.11 Networks

This operation returns a lists of the wireless networks in range of the device. A device should support this operation (see Table 35). The following status can be returned for each network:

- SSID (shall);
- BSSID (should);
- authentication and key management suite(s) (should);
- pair cipher(s) (should);
- group cipher(s) (should);
- signal strength (should).

Table 35 – ScanAvailable802.11Networks

ScanAvailable802.11Networks		Request-Response
Message name	Description	
ScanAvailableDot11-NetworksRequest	<i>tt:ReferenceToken InterfaceToken [1][1]</i>	
ScanAvailableDot11-NetworksResponse	<i>tt:Dot11AvailableNetworks Networks [0][unbounded]</i>	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:InvalidDot11	<i>IEEE 802.11 configuration is not supported.</i>	
env:Sender ter:InvalidArgVal ter:NotDot11	<i>The interface is not an IEEE 802.11 interface</i>	
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	<i>The supplied network interface token does not exist.</i>	
env;Receiver ter:ActionNotSupported ter:NotScanAvailable	<i>ScanAvailableDot11Networks is not supported.</i>	

8.3 System

8.3.1 Device Information

This operation gets device information, such as manufacturer, model and firmware version from a device. The device shall support the return of device information through the GetDeviceInformation command (see Table 36).

Table 36 – GetDeviceInformation command

GetDeviceInformation		Request-Response
Message name	Description	
GetDeviceInformationRequest	<i>This is an empty message.</i>	
GetDeviceInformationResponse	<i>The get device information response message returns following device information:</i> xs:string Manufacturer [1][1] xs:string Model [1][1] xs:string FirmwareVersion [1][1] xs:string SerialNumber [1][1] xs:string HardwareId [1][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.2 Get System URIs

This operation is used to retrieve URIs from which system information may be downloaded using HTTP. URIs may be returned for the following system information:

- System Logs. Multiple system logs may be returned, of different types. The exact format of the system logs is outside the scope of this standard;
- Support Information. This consists of arbitrary device diagnostics information from a device. The exact format of the diagnostic information is outside the scope of this standard;
- System Backup. The received file is a backup file that can be used to restore the current device configuration at a later date. The exact format of the backup configuration file is outside the scope of this standard.

If the device allows retrieval of system logs, support information or system backup data, it should make them available via HTTP GET. If it does, it shall support the GetSystemUris command (see Table 37).

Table 37 – GetSystemUri command

GetSystemUri		Request-Response
Message name	Description	
GetSystemUriRequest	<i>This is an empty message.</i>	
GetSystemUriResponse	<i>This message contains the URIs from which the various system information components may be downloaded.</i> tt:SystemLogUriList SystemLogUri [0][1] xs:anyURI SupportInfoUri [0][1] xs:anyURI SystemBackupUri [0][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.3 Backup

This operation is retrieves system backup configuration file(s) from a device. The device should support return of back up configuration file(s) through the GetSystemBackup command (see Table 38). The backup is returned with reference to a name and mime-type together with binary data. The exact format of the backup configuration files is *outside the scope* of this standard.

The backup configuration file(s) are transmitted through MTOM [MTOM].

Table 38 – GetSystemBackup command

GetSystemBackup		Request-Response
Message name	Description	
GetSystemBackupRequest	<i>This is an empty message.</i>	
GetSystemBackupResponse	<i>The get system backup response message contains the system backup configuration files(s).</i> tt:BackupFile BackupFiles [1][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.4 Restore

This operation restores the system backup configuration files(s) previously retrieved from a device. The device should support restore of backup configuration file(s) through the RestoreSystem command (see Table 39). The exact format of the backup configuration file(s) is *outside the scope* of this standard. If the command is supported, it shall accept backup files returned by the GetSystemBackup command.

The back up configuration file(s) are transmitted through MTOM [MTOM].

Table 39 – RestoreSystem command

RestoreSystem		Request-Response
Message name	Description	
RestoreSystemRequest	<i>This message contains the system backup file(s).</i> tt:BackupFile BackupFiles [1][unbounded]	
RestoreSystemResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidBackupFile	<i>The backup file(s) are invalid.</i>	

8.3.5 Start system restore

This operation initiates a system restore from backed up configuration data using the HTTP POST mechanism. The response to the command includes an HTTP URL to which the backup file may be uploaded. The actual restore takes place as soon as the HTTP POST operation has completed. Devices should support system restore through the StartSystemRestore command (see Table 40). The exact format of the backup configuration data is outside the scope of this standard.

System restore over HTTP may be achieved using the following steps:

- 1) client calls StartSystemRestore;
- 2) server responds with upload URI;
- 3) client transmits the configuration data to the upload URI using HTTP POST;
- 4) server applies the uploaded configuration, then reboots if necessary.

If the system restore fails because the uploaded file was invalid, the HTTP POST response shall be “415 Unsupported Media Type”. If the system restore fails due to an error at the device, the HTTP POST response shall be “500 Internal Server Error”.

The value of the Content-Type header in the HTTP POST request shall be “application/octet-stream”.

Table 40 – StartSystemRestore command

StartSystemRestore		Request-Response
Message name	Description	
StartSystemRestoreRequest	<i>This is an empty message</i>	
StartSystemRestoreResponse	<i>This message contains</i> <i>A URL to which the system configuration file may be uploaded.</i> <i>An optional duration that indicates how long the device expects to be unavailable after the upload is complete.</i> xs:anyURI UploadUri [1][1] xs:duration ExpectedDowntime [0][1]	
Fault codes	Description	
	<i>No command-specific faults.</i>	

8.3.6 Get system date and time

This operation gets the device system date and time. The device shall support the return of the daylight saving setting and of the manual system date and time (if applicable) or indication of NTP time (if applicable) through the GetSystemDateAndTime command (see Table 41).

A device shall provide the UTCDateTime information although the item is marked as optional to ensure backward compatibility.

Table 41 – GetSystemDateAndTime command

GetSystemDateAndTime		Request-Response
Message name	Description	
GetSystemDateAndTime-Request	<i>This is an empty message.</i>	
GetSystemDateAndTime-Response	<i>This message contains the date and time information of the device.</i> <i>“DateTimeType”: If the system time and date are set manually or by NTP</i> <i>“DaylightSavings”: Daylight savings on or off</i> <i>“TimeZone”: The time zone as it is defined in POSIX 1003.1, 8.3</i> <i>“UTCDateTime”: The time and date in UTC.</i> <i>“LocalDateTime”: The local time and date of the device</i> tt:SetDateTimeType DateTimeType [1][1] xs:boolean DayLightSavings [1][1] tt:TimeZone TimeZone [0][1] tt:DateTime UTCDateTime [0][1] tt:DateTime LocalDateTime [0][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.7 Set system date and time

This operation sets the device system date and time. The device shall support the configuration of the daylight saving setting and of the manual system date and time (if applicable) or indication of NTP time (if applicable) through the SetSystemDateAndTime command (see Table 42).

If system time and date are set manually, the client shall include UTCDateTime or LocalDateTime in the request.

Table 42 – SetSystemDateAndTime command

SetSystemDateAndTime		Request-Response
Message name	Description	
SetSystemDateAndTime-Request	<i>This message contains the date and time information of the device.</i> <i>“DateTimeType”: If the system time and date are set manually or by NTP</i> <i>“DaylightSavings”: Daylight savings on or off</i> <i>“TimeZone”: The time zone is defined in POSIX 1003.1, 8.3</i> <i>“UTCDateTime”: The time and date in UTC. If DateTimeType is NTP, UTCDateTime has no meaning.</i> tt:SetDateTimeType DateTimeType [1][1] xs:boolean DayLightSavings [1][1] tt:TimeZone TimeZone [0][1] tt:DateTime UTCDateTime [0][1]	
SetSystemDateAndTime-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidTimeZone	<i>An invalid time zone was specified.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDateTime	<i>An invalid date or time was specified.</i>	

8.3.8 Factory default

This operation reloads parameters of a device to their factory default values. The device shall support hard and soft factory default through the SetSystemFactoryDefault command (see Table 43). The meaning of *soft factory default* is device product-specific and vendor-specific. The effect of a *soft factory default* operation is not fully defined. However, it shall be guaranteed that after a soft reset the device is reachable on the same IP address as used before the reset. This means that basic network settings like IP address, subnet and gateway or DHCP settings are kept unchanged by the soft reset.

Table 43 – SetSystemFactoryDefault command

SetSystemFactoryDefault		Request-Response
Message name	Description	
SetSystemFactoryDefault-Request	<i>This message contains the types of factory default to perform.</i> <i>“Hard”: All parameters are set to their factory default value</i> <i>“Soft”: All parameters except device vendor specific parameters are set to their factory default values</i> tt:FactoryDefaultType FactoryDefault [1][1]	
SetSystemFactoryDefault-Response	<i>This is an empty message.</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.9 Firmware upgrade

This operation upgrades a device firmware version. After a successful upgrade the response message is sent before the device reboots. The device should support firmware upgrade through the UpgradeSystemFirmware command (see Table 44). The exact format of the firmware data is *outside the scope* of this standard.

The firmware is transmitted through MTOM [MTOM].

Table 44 – UpgradeSystemFirmware command

UpgradeSystemFirmware		Request-Response
Message name	Description	
UpgradeSystemFirmware-Request	This message contains the firmware used for the upgrade. The firmware upgrade is “soft” meaning that all parameters keep their current value. tt:AttachmentData Firmware [1][1]	
UpgradeSystemFirmware-Response	This message contains a “Message” string allowing the device to report back a message to the client as for an example “Upgrade successful, rebooting in x seconds.” xs:string Message [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:InvalidFirmware	<i>The firmware was invalid, i.e., not supported by this device.</i>	
env:Receiver ter:Action ter:FirmwareUpgrade- Failed	<i>The firmware upgrade failed.</i>	

8.3.10 Start firmware upgrade

This operation initiates a firmware upgrade using the HTTP POST mechanism. The response to the command includes an HTTP URL to which the upgrade file may be uploaded. The actual upgrade takes place as soon as the HTTP POST operation has completed. The device should support firmware upgrade through the StartFirmwareUpgrade command (see Table 45). The exact format of the firmware data is outside the scope of this standard.

Firmware upgrade over HTTP may be achieved using the following steps:

- 1) client calls StartFirmwareUpgrade;
- 2) server responds with upload URI and optional delay value;
- 3) client waits for delay duration if specified by server;
- 4) client transmits the firmware image to the upload URI using HTTP POST;
- 5) server reprograms itself using the uploaded image, then reboots.

If the firmware upgrade fails because the upgrade file was invalid, the HTTP POST response shall be “415 Unsupported Media Type”. If the firmware upgrade fails due to an error at the device, the HTTP POST response shall be “500 Internal Server Error”.

The value of the Content-Type header in the HTTP POST request shall be “application/octet-stream”.

Table 45 – StartFirmwareUpgrade command

StartFirmwareUpgrade		Request-Response
Message name	Description	
StartFirmwareUpgrade-Request	<i>This is an empty message</i>	
StartFirmwareUpgrade-Response	<i>This message contains:</i> <i>A URL to which the firmware file may be uploaded.</i> <i>An optional delay; the client shall wait for this amount of time before initiating the firmware upload.</i> <i>A duration that indicates how long the device expects to be unavailable after the firmware upload is complete.</i> xs:anyURI UploadUri [1][1] xs:duration UploadDelay [0][1] xs:duration ExpectedDowntime [0][1]	
Fault codes	Description	
	<i>No command-specific faults.</i>	

8.3.11 Get system logs

This operation gets a system log from a device. The device should support system log information retrieval through the GetSystemLog command (see Table 46). The exact format of the system logs is *outside the scope* of this standard.

The system log information is transmitted through MTOM [MTOM] or as a string.

Table 46 – GetSystemLog command

GetSystemLog		Request-Response
Message name	Description	
GetSystemLogRequest	<i>This message contains the type of system log to retrieve. The types of supported log information is defined in two different types:</i> <i>“System”: The system log</i> <i>“Access”: The client access log</i> tt:SystemLogType LogType [1][1]	
GetSystemLogResponse	This message contains the requested system log information. The device can choose if it wants to return the system log information as binary data in an attachment or as a common string. tt:AttachmentData Binary [0][1] xs:string String [0][1]	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:AccesslogUnavailable	There is no access log information available	
env:Sender ter:InvalidArgs ter:SystemlogUnavailable	There is no system log information available	

8.3.12 Get support information

This operation gets arbitrary device diagnostics information from a device. The device MAY support retrieval of diagnostics information through the GetSystemSupportInformation command (see Table 47). The exact format of the diagnostic information is *outside the scope* of this standard.

The diagnostics information is transmitted as an attachment through MTOM [MTOM] or as string.

Table 47 – GetSystemSupportInformation command

GetSystemSupportInformation		Request-Response
Message name	Description	
GetSystemSupport-InformationRequest	<i>This is an empty message.</i>	
GetSystemSupport-Information Response	<i>The message contains the support information. The device can choose if it wants to return the support information as binary data or as a common string.</i> tt:AttachmentData BinaryFormat [0][1] xs:string StringFormat [0][1]	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:SupportInformation-Unavailable	<i>There is no support information available.</i>	

8.3.13 Reboot

This operation reboots a device. Before the device reboots the response message shall be sent. The device shall support reboot through the SystemReboot command (see Table 48).

Table 48 – SystemReboot command

SystemReboot		Request-Response
Message name	Description	
SystemReboot	<i>This is an empty message.</i>	
SystemRebootResponse	This message contains a “Message” string allowing the device to report back a message to the client as for an example “Rebooting in x seconds.” xs:string Message [1][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.14 Get scope parameters

This operation *requests* the scope parameters of a device. The scope parameters are used in the device discovery to match a probe message, see Clause 7. The Scope parameters are of two different types:

- fixed;
- configurable.

Fixed scope parameters *cannot* be altered through the device management interface but are permanent device characteristics part of the device firmware configurations. The scope type is indicated in the scope list returned in the get scope parameters response. Configurable scope parameters can be set through the set and add scope parameters operations, see 0 and 0. The device shall support retrieval of discovery scope parameters through the GetScopes command (see Table 49). As some scope parameters are mandatory, the client always expects a scope list in the response.

Table 49 – GetScopes command

GetScopes		Request-Response
Message name	Description	
GetScopesRequest	This is an <i>empty</i> message.	
GetScopesResponse	<i>The scope response message contains a list of URIs defining the device scopes. See also Clause 7 for the ONVIF scope definitions.</i> tt:Scope: Scopes [1][unbounded]	
Fault codes	Description	
env:Receiver ter:Action ter:EmptyScope	Scope list is empty.	

8.3.15 Set scope parameters

This operation sets the scope parameters of a device. The scope parameters are used in the device discovery to match a probe message, see Clause 7.

This operation *replaces* all existing configurable scope parameters (not fixed parameters). If this shall be avoided, one should use the scope add command instead. The device shall support configuration of discovery scope parameters through the SetScopes command (voir Tableau 50).

Table 50 – SetScopes command

SetScopes		Request-Response
Message name	Description	
SetScopesRequest	<i>The set scope contains a list of URIs defining the device scope. See also Clause 7.</i> xs:anyURI: Scopes [1][unbounded]	
SetScopesResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:OperationProhibited ter:ScopeOverwrite	<i>Scope parameter overwrites fixed device scope setting, command rejected.</i>	
env:Receiver ter:Action ter:TooManyScopes	<i>The requested scope list exceeds the supported number of scopes.</i>	

8.3.16 Add scope parameters

This operation *adds* new configurable scope parameters to a device. The scope parameters are used in the device discovery to match a probe message, see Clause 7. The device shall support addition of discovery scope parameters through the AddScopes command (see Table 51).

Table 51 – AddScopes command

AddScopes		Request-Response
Message name	Description	
AddScopesRequest	<i>The add scope contains a list of URIs to be added to the existing configurable scope list. See also Clause 7.</i> xs:anyURI:ScopeItem [1][unbounded]	
AddScopesResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Receiver ter:Action ter:TooManyScopes	<i>The requested scope list exceeds the supported number of scopes.</i>	

8.3.17 Remove scope parameters

This operation *deletes* scope-configurable scope parameters from a device. The scope parameters are used in the device discovery to match a probe message, see Clause 7. The device shall support deletion of discovery scope parameters through the RemoveScopes command (see Table 52).

Table 52 – RemoveScopes command

RemoveScopes		Request-Response
Message name	Description	
RemoveScopesRequest	<i>The remove scope contains a list of URIs that should be removed from the device scope.</i> xs:anyURI: ScopeItem [1][unbounded]	
RemoveScopesResponse	<i>The scope response message contains a list of URIs that has been Removed from the device scope.</i> xs:anyURI: ScopeItem [0][unbounded]	
Fault codes	Description	
env:Sender ter:OperationProhibited ter:FixedScope	<i>Trying to Remove fixed scope parameter, command rejected.</i>	
env:Sender ter:InvalidArgVal ter:NoScope	<i>Trying to Remove scope which does not exist.</i>	

8.3.18 Get discovery mode

This operation gets the discovery mode of a device. See 7.2 for the definition of the different device discovery modes. The device shall support retrieval of the discovery mode setting through the GetDiscoveryMode command (see Table 53).

Table 53 – GetDiscoveryMode command

GetDiscoveryMode		Request-Response
Message name	Description	
GetDiscoveryModeRequest	<i>This is an empty message.</i>	
GetDiscoveryModeResponse	<i>This message contains the current discovery mode setting, i.e. discoverable or non-discoverable.</i> tt:DiscoveryMode: DiscoveryMode [1][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.19 Set discovery mode

This operation sets the discovery mode operation of a device. See 7.2 for the definition of the different device discovery modes. The device shall support configuration of the discovery mode setting through the SetDiscoveryMode command (see Table 54).

Table 54 – SetDiscoveryMode command

SetDiscoveryMode		Request-Response
Message name	Description	
SetDiscoveryModeRequest	<i>This message contains the requested discovery mode setting, i.e. discoverable or non-discoverable.</i> tt:DiscoveryMode: DiscoveryMode [1][1]	
SetDiscoveryModeResponse	<i>This is an empty message.</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.20 Get remote discovery mode

This operation gets the remote discovery mode of a device. See 7.4 for the definition of remote discovery extensions. A device that supports remote discovery shall support retrieval of the remote discovery mode setting through the GetRemoteDiscoveryMode command (see Table 55).

Table 55 – GetRemoteDiscoveryMode command

GetRemoteDiscoveryMode		Request-Response
Message name	Description	
GetRemoteDiscoveryMode-Request	<i>This is an empty message.</i>	
GetRemoteDiscoveryMode-Response	<i>This message contains the current remote discovery mode setting, i.e. discoverable or non-discoverable.</i> tt:DiscoveryMode: RemoteDiscoveryMode [1][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.21 Set remote discovery mode

This operation sets the remote discovery mode of operation of a device. See 7.4 for the definition of remote discovery remote extensions. A device that supports remote discovery shall support configuration of the discovery mode setting through the SetRemoteDiscoveryMode command (see Table 56).

Table 56 – SetRemoteDiscoveryMode command

SetRemoteDiscoveryMode		Request-Response
Message name	Description	
SetRemoteDiscoveryMode-Request	<i>This message contains the requested remote discovery mode setting, i.e. discoverable or non-discoverable.</i> tt:DiscoveryMode: RemoteDiscoveryMode [1][1]	
SetRemoteDiscoveryMode-Response	<i>This is an empty message.</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.22 Get remote DP addresses

This operation gets the remote DP address or addresses from a device. If the device supports remote discovery, as specified in 7.4, the device shall support retrieval of the remote DP address(es) through the GetDPAddresses command (see Table 57).

Table 57 – GetDPAddresses command

GetDPAddresses		Request-Response
Message name	Description	
GetDPAddressesRequest	<i>This is an empty message.</i>	
GetDPAddressesResponse	<i>This message contains the device configured remote DP address or addresses. If no remote DP address is configured, an empty list is returned.</i> tt:NetworkHost: DPAAddress [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.3.23 Set remote DP addresses

This operation sets the remote DP address or addresses on a device. If the device supports remote discovery, as specified in 7.4, the device shall support configuration of the remote DP address(es) through the SetDPAddresses command (see Table 58).

Table 58 – SetDPAddresses command

SetDPAddresses		Request-Response
Message name	Description	
SetDPAddressesRequest	<i>This message contains the device configured remote DP address or addresses.</i> tt:NetworkHost: DPAAddress [0][unbounded]	
SetDPAddressesResponse	<i>This is an empty message.</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.4 Security

This subclause contains a set of security management operations. Such operations are sensitive to network attacks and shall be protected using appropriate authorization levels in order not to compromise the device.

8.4.1 Get access policy

Access to different services and sub-sets of services should be subject to access control. The WS-Security framework gives the prerequisite for end-point authentication. Authorization decisions can then be taken using an *access security policy*. This standard does not mandate any particular policy description format or security policy but this is up to the device manufacturer or system provider to choose policy and policy description format of choice. However, an access policy (in arbitrary format) can be requested using this command (see

Table 59). If the device supports access policy settings based on WS-Security authentication, then the device shall support this command.

Table 59 – GetAccessPolicy command

GetAccessPolicy		Request-Response
Message name	Description	
GetAccessPolicyRequest	<i>This is an empty message.</i>	
GetAccessPolicyResponse	This message contains the requested policy file. tt:BinaryData PolicyFile [1][1]	
Fault codes	Description	
env:Receiver ter:Action ter:EmptyPolicy	<i>The device policy file does not exist or it is empty.</i>	

8.4.2 Set access policy

This command sets the device access security policy (for more details on the access security policy see the Get command, 8.4.1). If the device supports access policy settings based on WS-Security authentication, then the device shall support this command (see Table 60).

Table 60 – SetAccessPolicy command

SetAccessPolicy		Request-Response
Message name	Description	
SetAccessPolicyRequest	This message contains the policy file to set. tt:BinaryData PolicyFile [1][1]	
SetAccessPolicyResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:PolicyFormat	<i>The requested policy cannot be set due to unknown policy format.</i>	

8.4.3 Get users

This operation lists the registered users and corresponding credentials on a device. The device shall support retrieval of registered device users and their credentials for the user token through the GetUsers command (see Table 61).

Table 61 – GetUsers command

GetUsers		Request-Response
Message name	Description	
GetUsersRequest	<i>This is an empty message.</i>	
GetUsersResponse	<i>This message contains list of users and corresponding credentials. Each entry includes:</i> <i>Username</i> <i>User level,</i> <i>i.e, the username password is not included into the response.</i> tt:User: User [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults.</i>	

8.4.4 Create users

This operation creates new device users and corresponding credentials on a device for the user token profile, see 5.12 for user token definitions. The device shall support creation of device users and their credentials for the user token through the CreateUsers command (see Table 62). Either all users are created successfully or a fault message shall be returned without creating any user.

ONVIF compliant devices are recommended to support password length of at least 28 bytes, as clients may follow the password derivation mechanism which results in 'password equivalent' of length 28 bytes, as described in 3.1.2 of [ONVIF Security].

Table 62 – CreateUsers command

CreateUsers		Request-Response
Message name	Description	
CreateUsersRequest	<i>This message contains a user parameters element for a new user. Each user entry includes:</i> <i>Username</i> <i>Password</i> <i>UserLevel</i> tt:User: User [1][unbounded]	
CreateUsersResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:OperationProhibited ter:UsernameClash	<i>Username already exists.</i>	
env:Sender ter:OperationProhibited ter:PasswordTooLong	<i>The password is too long</i>	
env:Sender ter:OperationProhibited ter:UsernameTooLong	<i>The username is too long</i>	
env:Sender ter:OperationProhibited ter:Password	<i>Too weak password.</i>	
env:Receiver ter:Action ter:TooManyUsers	<i>Maximum number of supported users exceeded.</i>	
env:Sender ter:OperationProhibited ter:AnonymousNotAllowed	<i>User level anonymous is not allowed.</i>	
env:Sender ter:OperationProhibited ter:UsernameTooShort	<i>The username is too short</i>	

8.4.5 Delete users

This operation deletes users on a device for the user token profile, see 5.12 for user token definitions. The device shall support deletion of device users and their credentials for the user token through the DeleteUsers command (see Table 63). A device may have one or more

fixed users that cannot be deleted to ensure access to the unit. Either all users are deleted successfully or a fault message shall be returned and no users be deleted.

Table 63 – DeleteUsers command

DeleteUsers		Request-Response
Message name	Description	
DeleteUsersRequest	<i>This message contains the name of the user or users to be deleted.</i> xs:string: Username [1][unbounded]	
DeleteUsersResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:UsernameMissing	<i>Username NOT recognized.</i>	
env:Sender ter:InvalidArgVal ter:FixedUser	<i>Username may not be deleted</i>	

8.4.6 Set users settings

This operation updates the settings for one or several users on a device for the user token profile. The device shall support update of device users and their credentials for the user token through the SetUser command (see Table 64). Either all change requests are processed successfully or a fault message shall be returned and no change requests be processed.

In case the optional password value is omitted the password of the user will be cleared.

Table 64 – SetUser command

SetUser		Request-Response
Message name	Description	
SetUserRequest	<i>This message contains a list of users and corresponding parameters to be updated.</i> <i>Username</i> <i>Password</i> <i>UserLevel</i> tt:User: User [1][unbounded]	
SetUserResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:UsernameMissing	<i>Username NOT recognized.</i>	
env:Sender ter:OperationProhibited ter>PasswordTooLong	<i>The password is too long</i>	
env:Sender ter:OperationProhibited ter>PasswordTooWeak	<i>Too weak password.</i>	
env:Sender ter:OperationProhibited ter:AnonymousNotAllowed	<i>User level anonymous is not allowed.</i>	

8.4.7 IEEE 802.1X configuration

This standard defines the following parameters as a set of IEEE 802.1X configuration parameters.

- Configuration Token

This parameter indicates a reference token of IEEE 802.1X configuration parameters and is defined as 'Dot1XConfigurationToken' in [ONVIF Schema]. This naming convention of 'Dot1X', which actually represents 'IEEE 802.1X' is used for better readability of schema element in the generated source code.

- EAP Identity

This parameter indicates the user name of supplicant which connects to IEEE 802.1X managed network. This is defined as 'Identity' in [ONVIF Schema].

- EAP method

This parameter indicates authentication method used. This is defined as 'EAPMethod' in [ONVIF Schema].

- CA Certificate ID

This parameter indicates the ID of CA certificate used for authentication server verification. This is defined as 'CACertificateID' in [ONVIF Schema].

- Respective configuration parameters for selected EAP method

Depending on selected EAP method, some specific parameters are needed as follows

- **[EAP-MD5], [EAP-PEAP/MSCHAP-V2], [EAP-TTLS types]**: Identity password so that Authentication server can verify the user (the device) by using specified password. [EAP-MD5] method is not applicable for the purpose of 802.11 (WPA-Enterprise) usage.
- **[EAP-TLS]**: Client certificate ID so that the RADIUS server can verify the user (the device) by using specified certificate.

This IEEE 802.1X parameters will be referred by security configuration as a part of a certain network interface configuration. For the details, please refer to 8.2.10.

This standard assumes that IEEE 802.1X configuration on device will be done outside the IEEE 802.1X managed network. In case of reconfiguring the IEEE 802.1X settings, it is also assumed that it will be done outside the 802.1X managed network.

Note that in ONVIF 2.0 support for IEEE 802.1X is limited to IEEE 802.11 interfaces.

8.4.7.1 Create IEEE 802.1X configuration

This operation newly creates IEEE 802.1X configuration parameter set of the device. The device shall support this command (see Table 65) if it supports IEEE 802.1X. If the device receives this request with already existing configuration token (Dot1XConfigurationToken) specification, the device should respond with 'for:ReferenceToken' error to indicate there is some configuration conflict.

Table 65 – CreateDot1XConfiguration command

CreateDot1XConfiguration		Request-Response
Message name	Description	
CreateDot1XConfigurationRequest	<i>This message contains:</i> tt:Dot1XConfiguration Dot1XConfiguration[1][1]	
CreateDot1XConfigurationResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:EAPMethodNotSupported	<i>The suggested EAP method is not supported.</i>	
env:Receiver ter:Action ter:MaxDot1X	<i>Maximum number of IEEE 802.1X configurations reached.</i>	
env:Sender ter:OperationProhibited ter:CertificateID	<i>Invalid Certificate ID error.</i>	
env:Sender ter:InvalidArgVal ter:ReferenceToken	<i>Dot1XConfigurationToken already exists.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDot1X	<i>Invalid IEEE 802.1X configuration.</i>	

8.4.7.2 Set IEEE 802.1X configuration

While the CreateDot1XConfiguration command is trying to create a new configuration parameter set, this operation modifies existing IEEE 802.1X configuration parameter set of the device. A device that support IEEE 802.1X shall support this command (see Table 66).

Table 66 – SetDot1XConfigurationRequest command

SetDot1XConfiguration		Request-Response
Message name	Description	
SetDot1XConfigurationRequest	<i>This message contains:</i> tt:Dot1XConfiguration Dot1XConfiguration[1][1]	
SetDot1XConfigurationResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:EAPMethodNotSupported	<i>The suggested EAP method is not supported.</i>	
env:Sender ter:OperationProhibited ter:CertificateID	<i>Invalid Certificate ID error.</i>	
env:Sender ter:OperationProhibited ter:ReferenceToken	<i>Invalid Dot1XConfigurationToken error</i>	
env:Sender ter:InvalidArgVal ter:InvalidDot1X	<i>Invalid IEEE 802.1X configuration.</i>	

8.4.7.3 Get IEEE 802.1X configuration

This operation gets one IEEE 802.1X configuration parameter set from the device by specifying the configuration token (Dot1XConfigurationToken) (see Table 67).

A device that supports IEEE 802.1X shall support this command.

Regardless of whether the 802.1X method in the retrieved configuration has a password or not, the device shall not include the Password element in the response.

Table 67 – GetDot1XConfiguration command

GetDot1XConfiguration		Request-Response
Message name	Description	
GetDot1XConfigurationRequest	<i>This is message contains:</i> tt:ReferenceToken Dot1XConfigurationToken[1][1]	
GetDot1XConfigurationResponse	<i>This message contains:</i> tt:Dot1XConfiguration Dot1XConfiguration[1][1]	
Fault codes	Description	
env:Sender	<i>Invalid Dot1XConfigurationToken error</i>	
ter:OperationProhibited		
ter:ReferenceToken		

8.4.7.4 Get IEEE 802.1X configurations

This operation gets all the existing IEEE 802.1X configuration parameter sets from the device. The device shall respond with all the IEEE 802.1X configurations so that the client can get to know how many IEEE 802.1X configurations are existing and how they are configured (see Table 68).

A device that support IEEE 802.1X shall support this command.

Regardless of whether the 802.1X method in the retrieved configuration has a password or not, the device shall not include the Password element in the response.

Table 68 – GetDot1XConfigurations command

GetDot1XConfigurations		Request-Response
Message name	Description	
GetDot1XConfigurationsRequest	<i>This is an empty message.</i>	
GetDot1XConfigurationsResponse	<i>This message contains:</i> tt: Dot1XConfiguration Dot1XConfiguration[0][unbounded]	
Fault codes	Description	
<i>No command specific faults!</i>		

8.4.7.5 Delete IEEE 802.1X configuration

This operation deletes an IEEE 802.1X configuration parameter set from the device. Which configuration should be deleted is specified by the 'Dot1XConfigurationToken' in the request. A device that support IEEE 802.1X shall support this command (see Table 69).

Table 69 – DeleteDot1XConfigurations command

DeleteDot1XConfigurations		Request-Response
Message name	Description	
DeleteDot1XConfigurationRequest	<i>This message contains:</i> tt:ReferenceToken Dot1XConfigurationToken[1][1]	
DeleteDot1XConfigurationResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:OperationProhibited ter:ReferenceToken	<i>Invalid Dot1XConfigurationToken error</i>	
env:Receiver ter:OperationProhibited ter:ReferenceToken	<i>Cannot delete specified IEEE 802.1X configuration.</i>	

8.4.8 Create self-signed certificate

This operation generates a private/public key pair and also can create a self-signed device certificate as a result of key pair generation. The certificate is created using a suitable *onboard* key pair generation mechanism.

If a device supports *onboard* key pair generation, the device that supports TLS shall support this certificate creation command. And also, if a device supports *onboard* key pair generation, the device that support IEEE 802.1X shall support this command for the purpose of key pair generation. Certificates and key pairs are identified using certificate IDs. These IDs are either chosen by the certificate generation requester or by the device (in case that no ID value is given) (see Table 70).

Table 70 – CreateCertificate command

CreateCertificate		Request-Response
Message name	Description	
CreateCertificateRequest	<i>This message contains (if applicable) requested Certificate ID and additional other requested parameters: subject, valid not before and valid not after.</i> xs:token CertificateID [0][1] xs:string Subject [0][1] xs:dateTime ValidNotBefore [0][1] xs:dateTime ValidNotAfter [0][1]	
CreateCertificateResponse	<i>This message contains the generated self-signed certificate.</i> tt:Certificate NvtCertificate [1][1]	
Fault codes	Description	
env:Receiver ter:Action ter:KeyGeneration	<i>The private/public key generation failed.</i>	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>CertificateID already exists.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDateTime	<i>Specified ValidNotBefore or ValidNotAfter parameter is not valid.</i>	

8.4.9 Get certificates

This operation gets all device server certificates (including self-signed) for the purpose of TLS authentication and all device client certificates for the purpose of IEEE 802.1X authentication (see Table 71). This command lists only the TLS server certificates and IEEE 802.1X client certificates for the device (neither trusted CA certificates nor trusted root certificates). The certificates are returned as binary data. A device that supports TLS shall support this command and the certificates shall be encoded using ASN.1 [X.681], [X.682], [X.683] DER [X.690] encoding rules. See Table 71.

Table 71 – GetCertificates command

GetCertificates		Request-Response
Message name	Description	
GetCertificatesRequest	<i>This is an empty message.</i>	
GetCertificatesResponse	This message contains a list of the device certificates. tt:Certificate NvtCertificate [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.4.10 Get CA certificates

CA certificates will be loaded into a device and be used for the sake of following two cases. The one is for the purpose of TLS client authentication in TLS server function. The other one is for the purpose of Authentication Server authentication in IEEE 802.1X function. This operation gets all CA certificates loaded into a device (see Table 72). A device that supports either TLS client authentication or IEEE 802.1X shall support this command and the returned certificates shall be encoded using ASN.1 [X.681], [X.682], [X.683] DER [X.690] encoding rules.

Table 72 – GetCACertificates command

GetCACertificates		Request-Response
Message name	Description	
GetCACertificatesRequest	<i>This is an empty message.</i>	
GetCACertificatesResponse	This message contains a list of the CA certificates. tt:Certificate CACertificate [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.4.11 Get certificate status

This operation is specific to TLS functionality. This operation gets the status (enabled/disabled) of the device TLS server certificates. A device that supports TLS shall support this command (see Table 73).

Table 73 – GetCertificatesStatus command

GetCertificatesStatus		Request-Response
Message name	Description	
GetCertificatesStatusRequest	<i>This is an empty message.</i>	
GetCertificatesStatus-Response	This message contains a list of the device server certificates referenced by ID and their status. The status is defined as a Boolean value (true = enabled, false = disabled). tt:CertificateStatus CertificateStatus [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.4.12 Set certificate status

This operation is specific to TLS functionality. This operation sets the status (enable/disable) of the device TLS server certificates. A device that supports TLS shall support this command (see Table 74). Typically *only* one device server certificate is allowed to be enabled at a time.

Table 74 – SetCertificatesStatus command

SetCertificatesStatus		Request-Response
Message name	Description	
SetCertificatesStatusRequest	This message contains a list of device server certificates referenced by ID and the requested certificate status, i.e., enabled or disabled. tt:CertificateStatus CertificateStatus [0][unbounded]	
SetCertificatesStatus-Response	<i>This is an empty message</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>Unknown certificate reference.</i>	

8.4.13 Get certificate request

This operation requests a PKCS #10 certificate signature request from the device. The returned information field shall be either formatted exactly as specified in [PKCS#10] or PEM encoded [PKCS#10] format (see Table 75). In order for this command to work, the device shall already have a private/public key pair. This key pair should be referred by *CertificateID* as specified in the input parameter description. This CertificateID refers to the key pair generated using CreateCertificate command defined in 8.4.8.

A device that support *onboard* key pair generation that supports either TLS or IEEE 802.1X using client certificate shall support this command.

Table 75 – GetPkcs10Request command

GetPkcs10Request		Request-Response
Message name	Description	
GetPkcs10RequestRequest	<i>This message contains a reference to the certificate (key pair) and optional certificate parameters for the certificate request. These attributes needs to be encoded as DER ASN.1 objects.</i> xs:token CertificateID [1][1] xs:string Subject [0][1] xs:BinaryData Attributes [0][1]	
GetPkcs10RequestResponse	This message contains the PKCS#10 request data structure. tt:BinaryData Pkcs10Request [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>Invalid CertificateID</i>	
env:Receiver ter:Action ter:Signature	<i>PKCS#10 signature creation failed.</i>	

8.4.14 Get client certificate status

This operation is specific to TLS functionality. This operation gets the status (enabled/disabled) of the device TLS client authentication. A device that supports TLS shall support this command (see Table 76).

Table 76 – GetClientCertificateMode command

GetClientCertificateMode		Request-Response
Message name	Description	
GetClientCertificateMode-Request	<i>This is an empty message.</i>	
GetClientCertificateMode-Response	This message contains the device client authentication status, i.e., enabled or disabled. xs:boolean Enabled [1][1]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.4.15 Set client certificate status

This operation is specific to TLS functionality. This operation sets the status (enabled/disabled) of the device TLS client authentication. A device that supports TLS shall support this command (see Table 77).

Table 77 – SetClientCertificateMode command

SetClientCertificateMode		Request-Response
Message name	Description	
SetClientCertificateMode-Request	This message contains the requested device client authentication status, i.e., enabled or disabled. xs:boolean Enabled [1][1]	
SetClientCertificateMode-Response	<i>This is an empty message</i>	
Fault codes	Description	
env:Receiver ter:InvalidArgVal ter:ClientAuth	<i>Trying to enable client authentication, but client authentication is not supported or not configured.</i>	

8.4.16 Load device certificate

TLS server certificate(s) or IEEE 802.1X client certificate(s) created using the PKCS#10 certificate request command can be loaded into the device using this command (see 8.4.13). The certificate ID in the request shall be present. The device may sort the received certificate(s) based on the public key and subject information in the certificate(s).

The certificate ID in the request will be the ID value the client wish to have. The device is supposed to scan the generated key pairs present in the device to identify which is the

correspondent key pair with the loaded certificate and then make the link between the certificate and the key pair.

A device that supports *onboard* key pair generation that support either TLS or IEEE 802.1X shall support this command (see Table 78).

The certificates shall be encoded using ASN.1 [X.681], [X.682], [X.683] DER [X.690] encoding rules.

This command is applicable to any device type, although the parameter name is called for historical reasons NVTCertificate.

Table 78 – LoadCertificates command

LoadCertificates		Request-Response
Message name	Description	
LoadCertificatesRequest	This message contains a list of the device certificates to upload. tt:Certificate NVTCertificate [1][unbounded]	
LoadCertificatesResponse	<i>This is an empty message</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:CertificateFormat	Bad certificate format or the format is not supported by the device.	
env:Sender ter:InvalidArgVal ter:CertificateID	Certificate ID already exists.	
env:Sender ter:InvalidArgVal ter:InvalidCertificate	Invalid Certificate.	

8.4.17 Load device certificates in conjunction with its private key

There might be some cases that a Certificate Authority or some other equivalent creates a certificate without having PKCS#10 certificate signing request. In such cases, the certificate will be bundled in conjunction with its private key (see Table 79). This command will be used for such use case scenarios. The certificate ID in the request is optionally set to the ID value the client wish to have. If the certificate ID is not specified in the request, device can choose the ID accordingly.

This operation imports a private/public key pair into the device.

The certificates shall be encoded using ASN.1 [X.681], [X.682], [X.683] DER [X.690] encoding rules.

A device that does not support onboard key pair generation and support either TLS or IEEE 802.1X using client certificate shall support this command. A device that support onboard key pair generation MAY support this command. The security policy of a device that supports this operation should make sure that the private key is sufficiently protected.

Table 79 – LoadCertificateWithPrivateKey command

LoadCertificateWithPrivateKey		Request-Response
Message name	Description	
LoadCertificateWithPrivateKeyRequest	This message contains a private/public key pair to import. tt:CertificateWithPrivateKey CertificateWithPrivateKey[1][unbounded]	
LoadCertificateWithPrivateKeyResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env: Sender ter:InvalidArgVal ter:CertificateFormat	<i>Bad certificate format or the format is not supported by the device.</i>	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>CertificateID already exists.</i>	
env: Sender ter:InvalidArgVal ter: KeysNotMatching	The public and private key are not matching.	

8.4.18 Get certificate information request

This operation requests the information of a certificate specified by certificate ID. The device should respond with its “Issuer DN”, “Subject DN”, “Key usage”, “Extended key usage”, “Key Length”, “Version”, “Serial Number”, “Signature Algorithm” and “Validity” data as the information of the certificate, as long as the device can retrieve such information from the specified certificate. The IssuerDN and SubjectDN shall be encoded using the rules in [RFC 4514].

A device that supports either TLS or IEEE 802.1X should support this command (see Table 80).

Table 80 – GetCertificateInformation command

GetCertificateInformation	Request-Response
Message name	Description
GetCertificateInformationRequest	<i>This message contains:</i> <i>CertificateID: The token of the certificate.</i> <i>xs: token CertificateID [1][1]</i>
GetCertificateInformationResponse	<i>This message contains:</i> <i>tt:CertificateInformation CertificateInformation[1][1]</i>
Fault codes	Description
env:Sender ter:InvalidArgVal ter:CertificateID	<i>Invalid Certificate ID</i>

8.4.19 Load CA certificates

This command is used when it is necessary to load trusted CA certificates or trusted root certificates for the purpose of verification for its counterpart i.e. client certificate verification in TLS function or server certificate verification in IEEE 802.1X function (see Table 81).

A device that support either TLS or IEEE 802.1X shall support this command. The device shall support the DER format; other formats may be supported by the device. The device may sort the received certificate(s) based on the public key and subject information in the certificate(s). Either all CA certificates are loaded successfully or a fault message shall be returned without loading any CA certificate.

Table 81 – LoadCACertificates command

LoadCACertificates		Request-Response
Message name	Description	
LoadCACertificatesRequest	This message contains a list of the device CA certificates to upload. tt:Certificate CACertificate [1][unbounded]	
LoadCACertificatesResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:CertificateFormat	<i>Bad certificate format or the format is not supported by the device.</i>	
env:Sender ter:InvalidArgVal ter:CACertificateID	<i>CA Certificate ID already exists.</i>	
env:Receiver ter:OperationProhibited ter:MaxCertificates	<i>Maximum number of Certificates already loaded.</i>	

8.4.20 Delete certificate

This operation deletes a certificate or multiple certificates. The device MAY also delete a private/public key pair which is coupled with the certificate to be deleted. The device that support either TLS or IEEE 802.1X shall support the deletion of a certificate or multiple certificates through this command (see Table 82). Either all certificates are deleted successfully or a fault message shall be returned without deleting any certificate.

Table 82 – DeleteCertificates command

DeleteCertificates		Request-Response
Message name	Description	
DeleteCertificatesRequest	This message deletes certificates identified with the CertificateID parameter. xs:token CertificateID[1][unbounded]	
DeleteCertificatesResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>Unknown certificate reference.</i>	
env:Receiver ter:OperationProhibited ter:CertificateID	<i>Cannot delete specified Certificates.</i>	

8.4.21 Get remote user

This operation returns the configured remote user (if any). A device supporting remote user handling shall support this operation (see Table 83). The user is only valid for the WS-UserToken profile or as a HTTP / RTSP user.

The algorithm to use for deriving the password is described in 5.12.2.2. *A device shall never return the Password field in RemoteUser.*

Table 83 – GetRemoteUser command

GetRemoteUser		Request-Response
Message name	Description	
GetRemoteUserRequest	<i>This is an empty message.</i>	
GetRemoteUserResponse	<i>This message contains the configured remote user (if any). The value returned are:</i> • <i>xs:string Username [1][1]</i> • <i>xs:boolean UseDerivedPassword [1][1]</i> <i>tt:RemoteUser: RemoteUser [0][1]</i>	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:NotRemoteUser	<i>Remote User handling is not supported</i>	

8.4.22 Set remote user

This operation sets the remote user. A device supporting remote user handling shall support this operation (see Table 84). The user is only valid for the WS-UserToken profile or as a HTTP / RTSP user.

The password that is set shall always be the original (not derived) password.

If **UseDerivedPassword** is set password derivation shall be done by the device when connecting to a remote device. The algorithm to use for deriving the password is described in 5.12.2.2.

To remove the remote user **SetRemoteUser** should be called without the **RemoteUser** parameter.

Table 84 – SetRemoteUser command

SetRemoteUser		Request-Response
Message name	Description	
SetRemoteUserRequest	<i>This message contains the remote user. The value that can set are:</i> <i>xs:string Username [1][1]</i> <i>xs:string Password [0][1]</i> <i>xs:boolean UseDerivedPassword [1][1]</i> <i>tt:RemoteUser: RemoteUser [0][1]</i>	
SetRemoteUserResponse	<i>This is an empty message</i>	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:NotRemoteUser	<i>Remote User handling not supported</i>	

8.4.23 Get endpoint reference

A client can ask for the device service endpoint reference address property that can be used to derive the password equivalent for remote user operation. The device shall support the GetEndpointReference command returning the address property of the device service endpoint reference (see Table 85).

Table 85 – GetEndpointReference command

GetEndpointReference		Request-Response
Message name	Description	
GetEndpointReferenceRequest	<i>This is an empty message.</i>	
GetEndpointReferenceResponse	<i>The requested URL.</i> <i>xs:string GUID [1][1]</i>	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.5 Input/Output (I/O)

The commands in this subclause are deprecated. Refer to Clause 9.

The Input/Output (I/O) commands are used to control the state or observe the status of the I/O ports. If the device has I/O ports, then it shall support the I/O commands.

8.5.1 Get relay outputs

This operation gets a list of all available relay outputs and their settings (see Table 86).

Table 86 – GetRelayOutputs command

GetRelayOutputs		Request-Response
Message name	Description	
GetRelayOutputsRequest	<i>This is an empty message.</i>	
GetRelayOutputsResponse	<i>This message contains an array of relay outputs.</i> tt:RelayOutput RelayOutputs [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

8.5.2 Set relay output settings

This operation sets the settings of a relay output (see Table 87).

The relay can work in two relay modes:

- Bistable – after setting the state, the relay remains in this state;
- Monostable – after setting the state, the relay returns to its idle state after the specified time.

The physical idle state of a relay output can be configured by setting the IdleState to 'open' or 'closed' (inversion of the relay behaviour).

Idle State 'open' means that the relay is open when the relay state is set to 'inactive' through the trigger command (see 0) and closed when the state is set to 'active' through the same command.

Idle State 'closed' means that the relay is closed when the relay state is set to 'inactive' through the trigger command (see 0) and open when the state is set to 'active' through the same command.

Table 87 – SetRelayOutputSettings command.

SetRelayOutputSettings		Request-Response
Message name	Description	
SetRelayOutputSettingsRequest	<i>This message contains:</i> <i>“RelayToken”: Token reference to the requested relay output.</i> <i>“RelayOutputSettings”: The settings of the relay</i> . tt:ReferenceToken RelayOutputToken [1][1] tt:RelayOutputSettings RelayOutputSettings [1][1]	
SetRelayOutputSettingsResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Unknown relay token reference.</i>	
env:Sender ter:InvalidArgVal ter:ModeError	<i>Monostable delay time not valid</i>	

8.5.3 Trigger relay output

This operation triggers a relay output¹ (see Table 88).

¹ There is no GetRelayState command; the current logical state of the relay output is transmitted via notification and their properties.

Table 88 – SetRelayOutputState command

SetRelayOutputState		Request-Response
Message name	Description	
SetRelayOutputStateRequest	<i>This message contains:</i> <i>"RelayToken": Token reference to the requested relay output.</i> <i>"LogicalState": Trigger request, i.e., active or inactive.</i> tt:ReferenceToken RelayOutputToken [1][1] tt:RelayLogicalState LogicalState [1][1]	
SetRelayOutputStateResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Unknown relay token reference.</i>	

8.5.4 Auxiliary operation

This subclause describes operations to manage auxiliary commands supported by a device, such as controlling an Infrared (IR) lamp, a heater or a wiper or a thermometer that is connected to the device.

The supported commands can be retrieved by the AuxiliaryData parameter which derives from GetCapabilities command response. The command transmitted by using this command should match one of the supported commands listed in the AuxiliaryData response. If the capability command response lists only *irlampon* command, then the SendAuxiliaryCommand argument will be *irlampon*, which may indicate turning the connected IR lamp on.

A device that indicates auxiliary service capability shall support this command (see Table 89).

Table 89 – Send auxiliary command

SendAuxiliaryCommand		Request-Response
Message name	Description	
SendAuxiliaryCommandRequest	<i>This message contains the auxiliary command.</i> tt:AuxiliaryData AuxiliaryCommand[1][1]	
SendAuxiliaryCommandResponse	<i>The response contains the auxiliary response.</i> tt:AuxiliaryData AuxiliaryCommandResponse[0][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:AuxiliaryDataNotSupported	<i>The requested AuxiliaryCommand is not supported</i>	

8.6 Service specific fault codes

Table 90 lists the device service-specific fault codes. In addition, each command can also generate a generic fault, see Table 6.

The specific faults are defined as sub code of a generic fault, see 5.11.2.1. The parent generic subcode is the *subcode* at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 90 – Device service specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Receiver	ter:Action	The policy is empty	The device policy file does not exist or it is empty.
	ter:EmptyPolicy		
env:Receiver	ter:Action	The scope list is empty	Scope list is empty.
	ter:EmptyScope		
env:Receiver	ter:Action	Upgrade failed	The firmware upgrade failed.
	ter:FirmwareUpgradeFailed		
env:Receiver	ter:Action	Generating a key failed	The private/public key generation failed.
	ter:KeyGeneration		
env:Receiver	ter:Action	Creating a signature failed	PKCS#10 signature creation failed.
	ter:Signature		
env:Receiver	ter:InvalidArgVal	Client authentication not supported	Trying to enable client authentication, but client authentication is not supported or not configured
	ter:ClientAuth		
env:Receiver	ter:Action	Too many users	Maximum number of supported users exceeded.
	ter:TooManyUsers		
env:Receiver	ter:Action	Too large scope list	The scope list exceeds the supported number of scopes.
	ter:TooManyScopes		
env:Receiver	ter:ActionNotSupported	The service is not supported	The requested WSDL service category is not supported by the device.
	ter:NoSuchService		
env:Sender	ter:InvalidArgs	No access log available	There is no access log information available.
	ter:AccesslogUnavailable		
env:Sender	ter:InvalidArgVal	Invalid format	Bad certificate format or the format is not supported by the device.
	ter:CertificateFormat		
env:Sender	ter:InvalidArgVal	Invalid certificate ID	Unknown certificate reference or the certificate ID already exists.
	ter:CertificateID		
env:Sender	ter:InvalidArgVal	Invalid CA certificate ID	Unknown CA certificate reference or the CA certificate ID already exists.
	ter:CACertificateID		
env:Sender	ter:InvalidArgVal	Invalid file	The backup file(s) are invalid.
	ter:InvalidBackupFile		
env:Sender	ter:InvalidArgVal	Invalid date and time.	An invalid date or time was specified.
	ter:InvalidDateTime		

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Sender	ter:InvalidArgVal	Invalid name	The suggested NTP server name is invalid.
	ter:InvalidDnsName		
env:Sender	ter:InvalidArgs	Invalid firmware	The firmware was invalid i.e. not supported by this device.
	ter:InvalidFirmware		
env:Sender	ter:InvalidArgVal	Invalid address	The supplied gateway address was invalid.
	ter:InvalidGatewayAddress		
env:Sender	ter:InvalidArgVal	Invalid name	The requested hostname cannot be accepted by the device.
	ter:InvalidHostname		
env:Sender	ter:InvalidArgVal	Invalid speed	The suggested speed is not supported.
	ter:InvalidInterfaceSpeed		
env:Sender	ter:InvalidArgVal	Invalid type	The suggested network interface type is not supported.
	ter:InvalidInterfaceType		
env:Sender	ter:InvalidArgVal	Invalid address	The suggested IPv4 address is invalid.
	ter:InvalidIPv4Address		
env:Sender	ter:InvalidArgVal	Address does not exist	The IPv4 address to be removed does not exist.
	ter:NoIPv4Address		
env:Sender	ter:InvalidArgVal	Invalid address	The suggested IPv6 address is invalid.
	ter:InvalidIPv6Address		
env:Sender	ter:InvalidArgVal	Address does not exist	The IPv6 address to be removed does not exist.
	ter:NoIPv6Address		
env:Sender	ter:InvalidArgVal	Invalid data	The MTU value is invalid.
	ter:InvalidMtuValue		
env:Sender	ter:InvalidArgVal	Invalid token	The supplied network interface token does not exist
	ter:InvalidNetworkInterface		
env:Sender	ter:InvalidArgVal	Invalid data	An invalid time zone was specified.
	ter:InvalidTimeZone		
env:Sender	ter:InvalidArgVal	The list is full	It is not possible to add more IP filters since the IP filter list is full.
	ter:IPFilterListIsFull		
env:Sender	ter:InvalidArgVal	Invalid data	Monostable delay time not valid.
	ter:ModeError		
env:Sender	ter:InvalidArgs	Invalid format	The requested policy cannot be set due to unknown policy format.
	ter:PolicyFormat		
env:Sender	ter:InvalidArgVal	Unknown relay token.	The token reference is unknown.
	ter:RelayToken		
env:Sender	ter:InvalidArgVal	The service is not	The supplied network

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
	ter:ServiceNotSupported	supported	service is not supported.
env:Sender	ter:InvalidArgVal	No support information available	There is no support information available.
	ter:SupportInformationUnavailable		
env:Sender	ter:InvalidArgs	No system log available	There is no system log information available.
	ter:SystemlogUnavailable		
env:Sender	ter:InvalidArgVal	Username not recognized	Username NOT recognized.
	ter:UsernameMissing		
env:Sender	ter:OperationProhibited	Trying to delete fixed scope parameter	Trying to delete fixed scope parameter, command rejected.
	ter:FixedScope		
env:Sender	ter:InvalidArgVal	Scope does not exist	Trying to Remove scope which does not exist.
	ter:NoScope		
env:Sender	ter:OperationProhibited	Too weak password	Too weak password.
	ter:Password		
env:Sender	ter:OperationProhibited	Too long password	The password is too long.
	ter:PasswordTooLong		
env:Sender	ter:OperationProhibited	Too long password	The password is too short.
	ter:UsernameTooShort		
env:Sender	ter:OperationProhibited	Trying overwriting permanent device scope setting	Scope parameter overwrites permanent device scope setting, command rejected.
	ter:ScopeOverwrite		
env:Sender	ter:OperationProhibited	Username already exists	Username already exists.
	ter:UsernameClash		
env:Sender	ter:OperationProhibited	Too long username	The username is too long.
	ter:UsernameTooLong		
env:Receiver	ter:ActionNotSupported	Not supported	IEEE 802.11 Configuration is not supported.
	ter:InvalidDot11		
env:Sender	ter:InvalidArgVal	Not Supported	The selected security mode is not supported.
	ter:InvalidSecurityMode		
env:Sender	ter:InvalidArgVal	Not Supported	The selected station mode is not supported.
	ter:InvalidStationMode		
env:Sender	ter:InvalidArgVal	IEEE 802.11 value missing	<i>IEEE 802.11 value is missing in the security configuration.</i>
	ter:MissingDot11		
env:Sender	ter:InvalidArgVal	PSK value missing	PSK value is missing in security configuration.
	ter:MissingPSK		
env:Sender	ter:InvalidArgVal	IEEE 802.1X value is missing	IEEE 802.1X value in security configuration is missing or none existing.
	ter:MissingDot1X		

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Sender	ter:InvalidArgVal	IEEE 802.1X value is incompatible	IEEE 802.1X value in security configuration is incompatible with the network interface.
	ter:IncompatibleDot1X		
env:Sender	ter:InvalidArgVal	Not IEEE 802.11	The interface is not an IEEE 802.11 interface.
	ter:NotDot11		
env:Sender	ter:InvalidArgVal	Invalid IEEE 802.1X configuration	Specified IEEE 802.1X configuration is not valid.
	ter:InvalidDot1X		
env:Receiver	ter:Action	IEEE 802.11 not connected	IEEE 802.11 network is not connected.
	ter:NotConnectedDot11		
env:Receiver	ter:ActionNotSupported	ScanAvailableIEEE802.11Networks is not supported.	ScanAvailableIEEE802.11Networks is not supported.
	ter:NotScanAvailable		
env:Receiver	ter:ActionNotSupported	Remote User handling is not supported.	Remote User handling is not supported.
	ter:NotRemoteUser		
env:Receiver	ter:ActionNotSupported	The suggested EAP method is not supported.	The suggested EAP method is not supported.
	ter:EAPMethodNotSupported		
env:Receiver	ter:Action	Maximum number of IEEE 802.1X configurations reached.	Device reached maximum number of IEEE 802.1X configurations.
	ter:MaxDot1X		
env:Receiver	ter:OperationProhibited	Cannot delete specified IEEE 802.1X configuration.	It is not possible to delete specified IEEE 802.1X configuration.
	ter:ReferenceToken		
env:Receiver	ter:OperationProhibited	Cannot delete specified Certificate(s).	It is not possible to delete specified Certificate(s).
	ter:CertificateID		
env:Sender	ter:OperationProhibited	Invalid Dot1XConfigurationToken error.	Specified IEEE 802.1X configuration token is invalid.
	ter:ReferenceToken		
env:Sender	ter:OperationProhibited	Invalid Certificate ID error.	Specified Certificate ID is invalid.
	ter:CertificateID		

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Sender	ter:InvalidArgVal	Dot1XConfigurationToken already exists.	Specified Dot1XConfigurationToken already exists in the device.
	ter:ReferenceToken		
env:Sender	ter:InvalidArgVal	Invalid certificate.	Specified certificate is invalid.
	ter:InvalidCertificate		
env:Receiver	ter:OperationProhibited	Maximum number of Certificates already loaded.	Device reached maximum number of loaded Certificates.
	ter:MaxCertificates		
env:Sender	ter:OperationProhibited	Too weak password	Too weak password
	ter:PasswordTooWeak		
env:Sender	ter:InvalidArgVal	The requested AuxiliaryCommand is not supported.	The requested AuxiliaryCommand is not supported.
	ter:AuxiliaryDataNotSupported		
env:Sender	ter:InvalidArgVal	Invalid Timeout value specified.	Specified TimeOut value is invalid.
	ter:InvalidTimeOutValue		
env:Sender	ter:OperationProhibited	Number of available bytes exceeded.	Number of available bytes exceeded.
	ter:DataLengthOver		
env:Sender	ter:OperationProhibited	Sequence of character (delimiter) is not supported.	Sequence of character (delimiter) is not supported.
	ter:DelimiterNotSupport		
env:Receiver	ter:OperationProhibited	Device is not ready to operate in command mode.	Device is not ready to operate in command mode.
	ter:InvalidMode		
env:Sender	ter:InvalidArgVal	Removing fixed user	Client trying to remove fixed user.
	ter:FixedUser		
env:Sender	ter:OperationProhibited	User level anonymous is not allowed.	User level anonymous is not allowed.
	ter:AnonymousNotAllowed		

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
Env:Sender	ter:InvalidArgVal	Keys not matching	The public and private key is not matching.
	ter:KeysNotMatching		

9 Device IO Service

This service offers commands to retrieve and configure the physical Inputs and Outputs of a device.

Commands to request the available video and audio in- and outputs are defined as well as commands to request the available relays. This service also offers functions to request and change the configuration of these entities.

A device that has physical sources and outputs shall support this service as described in Annex C.3.

Some functionality of this service overlaps with functionality that is defined in the Media Service. If a device (e.g. a NVT) needs to implement both services it should use the commands that are defined in this service to configure its audio in- and outputs or its video sources.

9.1 VideoOutputs

9.1.1 General

The VideoOutput type represents the physical Video Outputs of a device that can be connected to a monitor to display the video signal. The structure contains the Layout Settings that can be configured using the Display Service (see Clause 14).

9.1.2 GetVideoOutputs

This command lists all available video outputs of a device (see Table 91). A device that has one or more physical video outputs shall support listing of available video outputs through the GetVideoOutputs command.

Table 91 – GetVideoOutputs command

GetVideoOutputs		Request-Response
Message name	Description	
GetVideoOutputsRequest	<i>This is an empty message.</i>	
GetVideoOutputsResponse	<i>Contains a list of structures describing all available video outputs of the device. If a device has no VideoOutputs an empty list is returned.</i> tt:VideoOutput VideoOutputs [0][unbounded]	
Fault codes	Description	
<i>No specific fault codes.</i>		

9.2 VideoOutputConfiguration

9.2.1 GetVideoOutputConfiguration

This operation requests the configuration of a Video Output. A device that has one or more Video Outputs shall support the retrieval of the VideoOutputConfiguration through this command (see Table 92).

Table 92 – GetVideoOutputConfiguration command

GetVideoOutputConfiguration		Request-Response
Message name	Description	
GetVideoOutputConfigurationRequest	<i>This message contains the token of the VideoOutput.</i> tt:ReferenceToken VideoOutputToken [1][1]	
GetVideoOutputConfigurationResponse	<i>This message contains the requested VideoOutputConfiguration with the matching token.</i> tt:VideoOutputConfiguration VideoOutputConfiguration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested VideoOutput indicated with VideoOutputToken does not exist.</i>	

9.2.2 SetVideoOutputConfiguration

This operation modifies a video output configuration (see Table 93). A device that has one or more video outputs shall support the setting of its video output configuration through this command.

Table 93 – SetVideoOutputConfiguration command

SetVideoOutputConfiguration		Request-Response
Message name	Description	
SetVideoOutputConfiguration-Request	<i>The Configuration element contains the modified VideoOutput configuration.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:VideoOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoOutputConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	

9.2.3 GetVideoOutputConfigurationOptions

This operation requests the VideoOutputConfigurationOptions of a VideoOutput. A device that has one or more Video outputs shall support the retrieval of VideoOutputConfigurationOptions through this command (see Table 94).

Table 94 – GetVideoOutputConfigurationOptions command

GetVideoOutputConfigurationOptions		Request-Response
Message name	Description	
GetVideoOutputConfiguration-OptionsRequest	<i>The VideoOutputToken element specifies the VideoOutput whose options are requested. The VideoOutput shall exist in the device</i>	
	tt:ReferenceToken VideoOutputToken[1][1]	
GetVideoOutputConfiguration-OptionsResponse	<i>The response contains the VideoOutputOptions of the device.</i>	
	tt:VideoOutputConfigurationOptions VideoOutputOptions[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	

9.3 VideoSources

9.3.1 General

A VideoSource represents physical video input. The structure contains the pixel resolution of the video, framerate and imaging settings. The imaging settings can be manipulated through the ImagingService if supported and contains parameters for focus, exposure and brightness, for example.

9.3.2 GetVideoSources

This operation lists all available video sources for the device (see Table 95). The device that has one or more video inputs shall support the listing of available video sources through the GetVideoSources command.

Table 95 – GetVideoSources command

GetVideoSources		Request-Response
Message name	Description	
GetVideoSourcesRequest	<i>This is an empty message.</i>	
GetVideoSourcesResponse	<i>Contains a list of structures describing all available video sources of the device. If a device has no Video Source an empty list is returned</i> tt:VideoSource VideoSource [0][unbounded]	
Fault codes	Description	
No specific fault codes.		

9.4 VideoSourceConfiguration

A VideoSourceConfiguration contains a reference to a VideoSource and a Bounds structure containing either the whole VideoSource pixel area or a sub-portion of it. The Bounds and VideoSource define the image that is streamed to a client.

9.4.1 GetVideoSourceConfiguration

This operation lists the video source configurations of a VideoSource. A device with one or more video sources shall support the GetVideoSourceConfigurations command (see Table 96).

Table 96 – GetVideoSourceConfiguration command

GetVideoSourceConfiguration		Request-Response
Message name	Description	
GetVideoSourceConfigurationRequest	<i>This message contains the token of the video input.</i> tt:ReferenceToken VideoSourceToken [1][1]	
GetVideoSourceConfigurationResponse	<i>This message contains the requested VideoSourceConfiguration with the matching token.</i> tt:VideoSourceConfiguration VideoSourceConfiguration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoSource	<i>The requested VideoSource indicated with VideoSourceToken does not exist.</i>	

9.4.2 SetVideoSourceConfiguration

This operation modifies a video input configuration. A device that has one or more video sources shall support the setting of the VideoSourceConfiguration through this command (see Table 97).

Table 97 – SetVideoSourceConfiguration command

SetVideoSourceConfiguration		Request-Response
Message name	Description	
SetVideoSourceConfiguration-Request	<i>The Configuration element contains the modified VideoSource configuration. The Configuration contains an element that specifies the VideoSource whose configuration is to be modified. The VideoSource shall exist in the device</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:VideoSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoSourceConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoSource	<i>The requested VideoSource does not exist</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	

9.4.3 GetVideoSourceConfigurationOptions

This operation requests the VideoSourceConfigurationOptions of a VideoSource. A device with one or more video sources shall support this command (see Table 98).

Table 98 – GetVideoSourceConfiguartionOptions command

GetVideoSourceConfiguartionOptions		Request-Response
Message name	Description	
GetVideoSourceConfiguration-OptionsRequest	<i>The VideoSourceToken element specifies the Video Input whose options are requested. The Video Input shall exist in the device</i> tt:ReferenceToken VideoSourceToken[1][1]	
GetVideoSourceConfiguration-OptionsResponse	<i>The VideoSourceOptions return the valid Bounds as well as a element that delivers the VideoSourceToken available. This field shall be set to the Source whose options are requested.</i> tt:VideoSourceConfigurationOptions VideoSourceOptions[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoSource	<i>The requested Video Input does not exist</i>	

9.5 AudioOutputs

9.5.1 General

The Audio Output represents the physical audio outputs that can be connected to a loudspeaker.

9.5.2 GetAudioOutputs

This command lists all available audio outputs of a device. A device that has one or more physical audio outputs shall support listing of available audio outputs through the GetAudioOutputs command (see Table 99).

Table 99 – GetAudioOutputs command

GetAudioOutputs		Request-Response
Message name	Description	
GetAudioOutputsRequest	<i>This is an empty message.</i>	
GetAudioOutputsResponse	<i>Contains a list of structures describing all available audio outputs of the device. If a device has no AudioOutputs an empty list is returned.</i> tt:AudioOutput AudioOutputs [0][unbounded]	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio or Audio Outputs are not supported by the Device</i>	

9.6 AudioOutputConfiguration

An AudioOutputConfiguration contains a reference to an existing AudioOutput. The AudioOutput configuration contains a parameter to control the output level.

9.6.1 GetAudioOutputConfiguration

This operation requests the AudioOutputConfiguration of an AudioOutput. A device that has one or more AudioOutputs shall support the retrieval of the AudioOutputConfiguration through this command (see Table 100).

Table 100 – GetAudioOutputConfiguration command

GetAudioOutputConfiguration		Request-Response
Message name	Description	
GetAudioOutputConfigurationRequest	<i>This message contains the token of the AudioOutput.</i> tt:ReferenceToken AudioOutputToken [1][1]	
GetAudioOutputConfigurationResponse	<i>This message contains the requested AudioOutputConfiguration with the matching token.</i> tt:AudioOutputConfiguration AudioOutputConfiguration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAudioOutput	<i>The requested AudioOutput indicated with AudioOutputToken does not exist.</i>	

9.6.2 SetAudioOutputConfiguration

This operation modifies an audio output configuration. A device that has one ore more audio outputs shall support the setting of the AudioOutputConfiguration through this command (see Table 101).

Table 101 – SetAudioOutputConfiguration command

SetAudioOutputConfiguration		Request-Response
Message name	Description	
SetAudioOutputConfiguration-Request	<i>The Configuration element contains the modified AudioOutput configuration. The Configuration contains an element that specifies the Audio Output whose configuration is to be modified. The Audio Output shall exist in the device.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AudioOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioOutputConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAudioOutput	<i>The requested Audio Output does not exist</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	

9.6.3 GetAudioOutputConfigurationOptions

This operation requests the AudioOutputConfigurationOptions of an AudioOutput. A device that has one or more AudioOutputs shall support this command (see Table 102).

Table 102 – GetAudioOutputConfigurationOptions command

GetAudioOutputConfigurationOptions		Request-Response
Message name	Description	
GetAudioOutputConfiguration-OptionsRequest	<i>The AudioOutputToken element specifies the Audio Output whose options are requested. The Audio Output shall exist in the device</i> tt:ReferenceToken AudioOutputToken[1][1]	
GetAudioOutputConfiguration-OptionsResponse	<i>The AudioOutputsOptions return the valid value ranges for SendPrimacy and OutputLevel as well as the AudioOutputToken available. This field shall be set to the Output whose options are requested.</i> tt:AudioOutputConfigurationOptions AudioOutputOptions[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAudioOutput	<i>The requested Audio Output does not exist</i>	

9.7 AudioSources

9.7.1 General

An AudioSource represents unencoded audio input and states the number of input channels

9.7.2 GetAudioSources

This operation lists all available audio sources for the device. The device that has one or more audio sources shall support the listing of available audio inputs through the GetAudioSources command (see Table 103).

Table 103 – GetAudioSources command

GetAudioSources		Request-Response
Message name	Description	
GetAudioSourcesRequest	<i>This is an empty message.</i>	
GetAudioSourcesResponse	<i>Contains a list of structures describing all available audio sources of the device. If a device has no Audio Input an empty list is returned</i> tt:AudioSource AudioSource [0][unbounded]	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>NVT does not support audio.</i>	

9.8 AudioSourceConfiguration

An AudioSourceConfiguration contains a reference to an Audio Source.

9.8.1 GetAudioSourceConfiguration

This operation lists the configuration of an Audio Input. A device with one or more audio inputs shall support the GetAudioSourceConfiguration command (see Table 104).

Table 104 – GetAudioSourceConfiguration command

GetAudioSourceConfiguration		Request-Response
Message name	Description	
GetAudioSourceConfigurationRequest	<i>This message contains the token of the AudioSource.</i> tt:ReferenceToken AudioSourceToken [1][1]	
GetAudioSourceConfigurationResponse	<i>This message contains the requested AudioSourceConfiguration with the matching token.</i> tt:AudioSourceConfiguration AudioSourceConfiguration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAudioSource	<i>The requested AudioSource indicated with AudioSourceToken does not exist.</i>	

9.8.2 SetAudioSourceConfiguration

This operation modifies an audio source configuration. A device that has a one or more audio sources shall support the setting of the AudioSourceConfiguration through this command (see Table 105).

Table 105 – SetAudioSourceConfiguration command

SetAudioSourceConfiguration		Request-Response
Message name	Description	
SetAudioSourceConfiguration-Request	<i>The Configuration element contains the modified AudioSource configuration. The Configuration contains an element that specifies the AudioSource whose configuration is to be modified. The Audio Input shall exist in the device.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AudioSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioSourceConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAudioSource	<i>The requested AudioSource does not exist</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	

9.8.3 GetAudioSourceConfigurationOptions

This operation requests the AudioSourceConfigurationOptions of an AudioSource. A device with one or more AudioSources shall support this command (see Table 106).

Table 106 – GetAudioSourceConfigurationOptions command

GetAudioSourceConfigurationOptions		Request-Response
Message name	Description	
GetAudioSourceConfigurationOptions-Request	<i>The AudioSourceToken element specifies the Audio Input whose options are requested. The AudioSource shall exist in the device</i> tt:ReferenceToken AudioSourceToken[1][1]	
GetAudioSourceConfiguration-Response	<i>The AudioSourcesOptions return the AudioSourceToken available. This field shall be set to the source whose options are requested.</i> tt:AudioSourceConfigurationOptions AudioSourceOptions[1][1]	
Fault codes	Description	
env:Sender	<i>The requested Audio Input does not exist</i>	
ter:InvalidArgVal		
ter:NoAudioSource		

9.9 Relay outputs

The Input/Output (I/O) commands are used to control the state or observe the status of the I/O ports. If the device has I/O ports, then it shall support the I/O commands.

Relay outputs is also defined in DeviceManagement (see Input/Output(I/O)). Relay outputs can access both DeviceManagement service and DeviceIO.

9.9.1 Get relay outputs

This operation gets a list of all available relay outputs and their settings (see Table 107).

Table 107 – GetRelayOutputs command

GetRelayOutputs		Request-Response
Message name	Description	
GetRelayOutputsRequest	<i>This is an empty message.</i>	
GetRelayOutputsResponse	<i>This message contains an array of relay outputs.</i> tt:RelayOutput RelayOutputs [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

9.9.2 Set relay output settings

This operation sets the settings of a relay output.

The relay can work in two relay modes:

- Bistable – after setting the state, the relay remains in this state;
- Monostable – after setting the state, the relay returns to its idle state after the specified time.

The physical idle state of a relay output can be configured by setting the IdleState to 'open' or 'closed' (inversion of the relay behaviour).

Idle State 'open' means that the relay is open when the relay state is set to 'inactive' through the trigger command (see 0) and closed when the state is set to 'active' through the same command.

Idle State 'closed' means, that the relay is closed when the relay state is set to 'inactive' through the trigger command (see 0) and open when the state is set to 'active' through the same command (see Table 108).

Table 108 – SetRelayOutputSettings command

SetRelayOutputSettings		Request-Response
Message name	Description	
SetRelayOutputSettingsRequest	<i>This message contains:</i> <i>“RelayOutputToken”:</i> Token reference to the requested relay output. <i>“RelayOutputSettings”:</i> The settings of the relay tt:ReferenceToken RelayOutputToken [1][1] tt:RelayOutputSettings RelayOutputSettings [1][1]	
SetRelayOutputSettingsResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Unknown relay token reference.</i>	
env:Sender ter:InvalidArgVal ter:ModeError	<i>Monostable delay time not valid</i>	

9.9.3 Trigger relay output

This operation triggers a relay output² (see Table 109).

² There is no GetRelayState command; the current logical state of the relay output is transmitted via notification and their properties.

Table 109 – SetRelayOutputState command

SetRelayOutputState		Request-Response
Message name	Description	
SetRelayOutputStateRequest	<i>This message contains:</i> <i>"RelayOutputToken": Token reference to the requested relay output.</i> <i>"LogicalState": Trigger request, i.e., active or inactive.</i> tt:ReferenceToken RelayOutputToken [1][1] tt:RelayLogicalState LogicalState [1][1]	
SetRelayOutputStateResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Unknown relay token reference.</i>	

9.10 Service specific fault codes

Table 110 lists the DeviceIO service specific fault codes. Additionally, each command can also generate a generic fault, see Table 6.

Table 110 – DeviceIO service specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Sender	ter:InvalidArgVal	Invalid configuration parameters	The configuration parameters are not possible to set.
	ter:ConfigModify		
env:Sender	ter:InvalidArgVal	Video output token does not exist.	The requested VideoOutput indicated with VideoOutputToken does not exist.
	ter:NoVideoOutput		
env:Sender	ter:InvalidArgVal	Video source token does not exist.	The requested VideoSource indicated with VideoSourceToken does not exist.
	ter:NoVideoSource		
env:Sender	ter:InvalidArgVal	Audio output token does not exist.	The requested AudioOutput indicated with AudioOutputToken does not exist.
	ter:NoAudioOutput		
env:Sender	ter:InvalidArgVal	Audio source token does not exist.	The requested AudioSource indicated with AudioSourceToken does not exist.
	ter:NoAudioSource		
env:Sender	ter:InvalidArgVal	Unknown relay token reference	The requested RelayOutput indicated RelayOutputToken does not exist.
	ter:RelayToken		
env:Sender	ter:InvalidArgVal	Monostable delay time not valid	
	ter:ModeError		

10 Imaging configuration

The imaging service provides operations used to control and configure imaging properties on a device. A device that has one or more video sources should support the imaging service as defined in C.7. The imaging settings are part of the VideoSource entity. This means that imaging parameters directly affect a specific video source.

10.1 Imaging settings

The imaging service provides operations to get or set imaging parameters and the valid ranges for those parameters. Some parameters have no effect if a specific mode is not set. Some of the parameters included in the settings require a specific imaging capability that can be requested through the GetOptions command. The following settings are available through the imaging service operations:

BacklightCompensation: Enables/disables BLC mode (on/off)

- On
 - Optional level parameter (unspecified unit).
- Off

Brightness: Adjusts the image brightness (unspecified unit).

ColorSaturation: Adjusts the color saturation in the image (unspecified unit).

Sharpness: Adjusts the sharpness in the image (unspecified unit).

Contrast: Adjusts the image contrast (unspecified unit).

Exposure:

- Auto – Enables the exposure algorithm on the device:
 - Priority – Sets the exposure priority mode (low noise/framerate).
 - Window – Rectangular exposure mask.
 - Min/MaxExposureTime – Exposure time range allowed to be used by the algorithm.
 - Min/MaxGain – The sensor gain range that is allowed to be used by the algorithm.
 - Min/MaxIris – The iris range allowed to be used by the algorithm.
- Manual – Disables the exposure algorithm on the device:
 - ExposureTime – The fixed exposure time used by the image sensor (μ s).
 - Gain – The fixed gain used by the image sensor (dB).
 - Iris – The fixed attenuation of input light affected by the iris (dB). 0 dB maps to a fully opened iris.

Focus:

- Auto (parameters that apply to automatic mode only):
 - Near/FarLimit – Limits for focus lens (m).
- Manual (parameters that apply to manual mode only):
 - Default speed – The default speed for focus move operation (when the speed parameter not is present). Manual control is done through the move command, see 0.

Ir cut filter: Toggles the Ir cut filter state between on, off and auto. The auto state lets the exposure algorithm handle when the Ir cut filter should be turned on or off.

Whitebalance:

- Auto whitebalancing mode (auto/manual).
- Manual (parameters that apply to manual mode only):
 - Rgain (unitless).
 - Bgain (unitless).

WideDynamicRange: Wide dynamic range (on/off):

- On
 - Optional level parameter (unitless).
- Off

The available imaging settings can be retrieved through the GetVideoSources command part of the media service, as specified in 11.3.2. The imaging settings are part of the video source.

10.1.1 Get imaging settings

This operation requests the imaging setting for a video source on the device. If the Video Source supports any of the imaging settings as defined by the ImagingSettings type in the [ONVIF Schema], then it should be possible to retrieve the imaging settings from the device through the GetImagingSettings command (see Table 111).

The imaging settings parameters are described in 10.1.

Table 111 – GetImagingSettings command

GetImagingSettings		Request-Response
Message name	Description	
GetImagingSettingsRequest	<i>This message contains a reference to the VideoSource for which the ImagingSettings should be requested.</i> tt:ReferenceToken VideoSourceToken[1][1]	
GetImagingSettingsResponse	<i>This message contains the ImagingSettings for the VideoSource that was requested</i> tt:ImagingSettings20ImagingSettings[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoSource	<i>The requested VideoSource does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	<i>The requested VideoSource does not support imaging settings.</i>	

10.1.2 Set imaging settings

This operation sets the imaging settings for a video source on a device. If the device supports any of the imaging settings as defined by the ImagingSettings type in [ONVIF Schema], then the it should be possible to configure these parameters in the device through the SetImagingSettings command (see Table 112).

The possible configurable imaging settings parameters are described in 10.1. Settings options are obtained through the command defined in 0

Table 112 – SetImagingSettings command

SetImagingSettings		Request-Response
Message name	Description	
SetImagingSettingsRequest	<i>This message contains a reference to the VideoSource and ImagingSettings that should be set.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:ReferenceToken VideoSourceToken[1][1] tt:ImagingSettings20ImagingSettings[1][1] xs:boolean ForcePersistence [0][1]	
SetImagingSettingsResponse	<i>This message contains no response.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoSource	<i>The requested VideoSource does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	<i>The requested VideoSource does not support imaging settings.</i>	
env:Sender ter:InvalidArgVal ter:SettingsInvalid	<i>The requested settings are incorrect.</i>	

10.1.3 Get options

This operation gets the valid ranges for the imaging parameters that have device specific ranges. If the device supports the SetImagingSettings command to set imaging parameter on the device, then it shall get the configuration options from the device through the GetOptions command (see Table 113).

Table 113 – GetOptions command

GetOptions		Request-Response
Message name	Description	
GetOptionsRequest	<i>Reference to the VideoSource for which the imaging parameter options are requested.</i> tt:ReferenceToken VideoSourceToken[1][1]	
GetOptionsResponse	<i>This message contains the valid ranges for the imaging parameters that are categorized as device specific.</i> tt:ImagingOptions20 ImagingOptions[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoSource	<i>The requested VideoSource does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	<i>The requested VideoSource does not support imaging settings.</i>	

10.1.4 Move

The Move command moves the focus lens in an absolute, a relative or in a continuous manner from its current position (see Table 114). The speed argument is optional for absolute and relative control, but required for continuous. If no speed argument is used, the default speed is used. Focus adjustments through this operation will turn off the autofocus. A device with support for remote focus control should support absolute, relative or continuous control through the Move operation.

Imaging capabilities specifies which specific focus operations are supported by this operation. At least one focus control capability is required for this operation to be functional.

The move operation contains the following commands:

Absolute – Requires position parameter and optionally takes a speed argument. A unitless type is used by default for focus positioning and speed. Optionally, if supported, the position may be requested in m^{-1} units.

Relative – Requires distance parameter and optionally takes a speed argument. Negative distance means negative direction.

Continuous – Requires a speed argument. Negative speed argument means negative direction.

Table 114 – Move (focus) command

Move		Request-Response
Message name	Description	
MoveRequest	<i>Reference to the VideoSource for the requested move (focus) operation.</i> tt:ReferenceToken VideoSourceToken[1][1] tt:FocusMove Focus[1][1]	
MoveResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoSource	<i>The requested VideoSource does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	<i>The requested VideoSource does not support imaging settings.</i>	

10.1.5 Get move options

The GetMoveOptions command retrieves the focus lens move options to be used in the move command as defined in 0. A device that supports the lens move operation shall also support the GetMoveOptions command (see Table 115).

Table 115 – GetMoveOptions (focus) command

GetMoveOptions		Request-Response
Message name	Description	
GetMoveOptionsRequest	<i>Reference to the VideoSource for the requested move options.</i> tt:ReferenceToken VideoSourceToken [1][1]	
GetMoveOptionsResponse	<i>This message contains the valid ranges for the focus lens move options.</i> tt:MoveOptions20 MoveOptions [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoSource	<i>The requested VideoSource does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	<i>The requested VideoSource does not support imaging settings.</i>	

10.1.6 Stop

The Stop command stops all ongoing focus movements of the lense (see Table 116). If the device supports focus, it should be possible to stop focus through the stop operation. The operation will not affect ongoing autofocus operation.

Table 116 – Stop (focus) command

Stop		Request-Response
Message name	Description	
StopRequest	<i>Reference to the VideoSource where the focus movement should be stopped.</i> tt:ReferenceToken VideoSourceToken[1][1]	
StopResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoSource	<i>The requested VideoSource does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	<i>The requested VideoSource does not support imaging settings.</i>	

10.1.7 Get imaging status

The GetStatus command requested the current imaging status from the device (see Table 117). If the device supports focus move control, then it should be possible to get the available imaging status through the GetStatus command.

The imaging status contains:

- Focus position, move status and error information.
 - The focus position is represented in a unitless type.
 - Move status may be in a MOVING, IDLE or UNKNOWN state.
 - Error information provided as a string, for example a positioning error indicated by the hardware.

Table 117 – GetStatus (focus) command

GetStatus		Request-Response
Message name	Description	
GetStatusRequest	<i>This message contains a reference to the VideoSource where the imaging status should be requested.</i> tt:VideoSourceToken VideoSourceToken[1][1]	
GetStatusResponse	This message contains the requested imaging status. tt:ImagingStatus20 ImagingStatus[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoSource	<i>The requested VideoSource does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource	<i>The requested VideoSource does not support imaging settings.</i>	

10.2 Service specific fault codes

Table 118 lists the imaging service specific fault codes. In addition each command can also generate a generic fault, see Table 6.

The specific faults are defined as subcode of a generic fault, see 5.11.2.1. The parent generic subcode is the subcode at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 118 – Imaging specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Receiver	ter:ActionNotSupported	VideoSource does not support imaging settings	The requested VideoSource does not support imaging settings.
	ter:NoImagingForSource		
env:Sender	ter:InvalidArgVal	Invalid configuration	The requested settings are incorrect.
	ter:SettingsInvalid		
env:Sender	ter:InvalidArgVal	Video source does not exist	The requested VideoSource does not exist.
	ter:NoSource		

11 Media configuration

The media service is used to configure the NVT media streaming properties. The NVT shall support the media service as specified in C.8.

The media service allows a client to configure media and other real time streaming configurations. Media configurations are handled through media profiles. An overview of the ONVIF media configuration model is given in 4.8.

The media service commands are divided into two major categories:

- Media configuration:
 - Media profile commands
 - Video source commands
 - Video encoder commands
 - Audio source commands
 - Audio encoder commands
 - Video analytics commands
 - Metadata commands
 - Audio output commands
 - Audio decoder commands
- Media streaming:
 - Request stream URI
 - Get snapshot URI
 - Multicast control commands
 - Media synchronization point

A basic set of operations are required for the media service; other operations are recommended to support. The detailed requirements are listed under the command descriptions.

11.1 Audio and video codecs

The NVT streams audio and video data using suitable encoding algorithms. The NVT may also be able to decode audio. The NVT supports any audio and video codecs, bitrates and resolution according to the manufacturer's choice. In order to ensure interoperability between the client and NVT, this standard mandates the following codec profiles:

- The NVT shall support JPEG QVGA.
- The NVT shall support G.711 μ Law (Simplex-Camera Microphone Only, 1ch) [ITU-T G.711] if the NVT supports audio.

11.2 Media profile

A media profile consists of a set of media configurations. Media profiles are used by a client to configure properties of a media stream from an NVT.

An NVT shall provide at least one media profile at boot. An NVT should provide “ready to use” profiles for the most common media configurations that the device offers.

A profile consists of a set of interconnected *configuration entities*. Configurations are provided by the NVT and can be either static or created dynamically by the NVT. For example, the dynamic configurations can be created by the NVT depending on current available encoding resources. A configuration entity is one of the following:

- Video source configuration
- Audio source configuration
- Video encoder configuration
- Audio encoder configuration
- PTZ configuration
- Video analytics configuration
- Metadata configuration
- Audio output configuration
- Audio decoder configuration

A profile consists of all or a subset of these configuration entities. Depending on the capabilities of the NVT, a particular configuration entity can be part of a profile or not. For example, a profile with an audio source and an audio encoder configuration can exist only in a device with audio support.

11.2.1 Create media profile

This operation creates a new empty media profile (see Table 119). The media profile shall be created in the NVT and shall be persistent (remain after reboot). The NVT shall support the creation of media profiles as defined in this standard through the CreateProfile command. The maximum number of profiles a device supports is returned in the media capabilities.

A created profile shall be deletable and an NVT shall set the “fixed” attribute to false in the returned Profile.

Table 119 – CreateProfile command

CreateProfile		Request-Response
Message name	Description	
CreateProfileRequest	<i>Contains the friendly Name of the Profile to create as well as an optional Token parameter, specifying the unique identifier of the new media profile</i> tt:Name Name [1][1] tt:ReferenceToken Token [0][1]	
CreateProfileResponse	<i>Returns an empty Profile structure with no configuration entities.</i> tt:Profile Profile [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:ProfileExists	<i>A profile with the token ProfileToken already exists.</i>	
env:Receiver ter:Action ter:MaxNVTProfiles	<i>The maximum number of supported profiles has been reached.</i>	

11.2.2 Get media profiles

Any endpoint can ask for the *existing* media profiles of an NVT using the GetProfiles command. Pre-configured or dynamically configured profiles can be retrieved using this command (see Table 120). This command lists *all* configured profiles in a device. The client does not need to know the media profile in order to use the command. The NVT shall support the retrieval of media profiles through the GetProfiles command.

A NVT shall include the “fixed” attribute in all the returned Profile elements.

Table 120 – GetProfiles command

GetProfiles		Request-Response
Message name	Description	
GetProfilesRequest	<i>This is an empty message.</i>	
GetProfilesResponse	The response contains a list of profiles. Each profile contains a set of configuration entities defining a specific configuration that can be used for media streaming, analytics, metadata streaming etc. tt:Profile Profiles [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

11.2.3 Get media profile

If the profile token is already known, a profile can be fetched through the GetProfile command (see Table 121). The NVT shall support the retrieval of a specific media profile through the GetProfile command.

A NVT shall include the “fixed” attribute in the returned Profile element.

Table 121 – GetProfile command

GetProfile		Request-Response
Message name	Description	
GetProfileRequest	This message contains the token to the requested profile. tt:ReferenceToken ProfileToken [1][1]	
GetProfileResponse	The response contains the Profile indicated by the Token parameter. A Profile contains a set of configuration entities defining a specific configuration that can be used for media streaming, analytics, metadata streaming etc. tt:Profile Profile [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	

11.2.4 Add video source configuration to a profile

This operation adds a VideoSourceConfiguration to an existing media profile (see Table 122). If such a configuration exists in the media profile, it will be replaced. The change shall be persistent. The NVT shall support addition of a video source configuration to a profile through the AddVideoSourceConfiguration command.

Table 122 – AddVideoSourceConfiguration command

AddVideoSourceConfiguration		Request-Response
Message name	Description	
AddVideoSourceConfiguration Request	Contains a reference to the VideoSourceConfiguration to add and the Profile where it shall be added. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddVideoSourceConfiguration Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The VideoSourceConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	

11.2.5 Add video encoder configuration to a profile

This operation adds a VideoEncoderConfiguration to an existing media profile (see Table 123). If a configuration exists in the media profile, it will be replaced. The change shall be persistent. An NVT shall support addition of a video encoder configuration to a profile through the AddVideoEncoderConfiguration command.

Adding a VideoEncoderConfiguration to a Profile means that a stream using that Profile will contain video data. Video encoder configurations should be added after adding a video source configuration.

Table 123 – AddVideoEncoderConfiguration command

AddVideoEncoderConfiguration		Request-Response
Message name	Description	
AddVideoEncoderConfiguration Request	Contains a reference to the VideoEncoderConfiguration to add and the Profile where it shall be added. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddVideoEncoderConfiguration Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgs ter:NoConfig	<i>The VideoEncoderConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	

11.2.6 Add audio source configuration to a profile

This operation adds an AudioSourceConfiguration to an existing media profile (see Table 124). If a configuration exists in the media profile, it will be replaced. The change shall be persistent. An NVT that supports audio streaming from NVT to client shall support addition of audio source configuration to a profile through the AddAudioSourceConfiguration command.

Table 124 – AddAudioSourceConfiguration command

AddAudioSourceConfiguration		Request-Response
Message name	Description	
AddAudioSourceConfiguration Request	Contains a reference to the AudioSourceConfiguration to add and the Profile where it shall be added. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioSourceConfiguration Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The AudioSourceConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Audio is not supported.</i>	

11.2.7 Add audio encoder configuration to a profile

This operation adds an AudioEncoderConfiguration to an existing media profile (see Table 125). If a configuration exists in the media profile, it will be replaced. The change shall be persistent. An NVT that supports audio streaming from NVT to client shall support addition of audio encoder configurations to a profile through the AddAudioEncoderConfiguration command.

Adding an AudioEncoderConfiguration to a media profile means that streams using that media profile will contain audio data. Audio encoder configurations should be added after adding an audio source configuration.

Table 125 – AddAudioEncoderConfiguration command

AddAudioEncoderConfiguration		Request-Response
Message name	Description	
AddAudioEncoderConfiguration Request	Contains a reference to the AudioEncoderConfiguration to add and the Profile where it shall be added. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioEncoderConfiguration Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The AudioEncoderConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Audio is not supported.</i>	

11.2.8 Add PTZ configuration to a profile

This operation adds a PTZConfiguration to an existing media profile (see Table 126). If a configuration exists in the media profile, it will be replaced. The change shall be persistent. An NVT that supports PTZ control shall support addition of PTZ configurations to a profile through the AddPTZConfiguration command.

Adding a PTZConfiguration to a media profile means that streams using that media profile can contain PTZ status (in the metadata), and that the media profile can be used for controlling PTZ movement, see Clause 16.

Table 126 – AddPTZConfiguration command

AddPTZConfiguration		Request-Response
Message name	Description	
AddPTZConfigurationRequest	Contains a reference to the PTZConfiguration to add and the Profile where it shall be added. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddPTZConfigurationResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The PTZConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

11.2.9 Add video analytics configuration to a profile

This operation adds a VideoAnalytics configuration to an existing media profile (see Table 127). If a configuration exists in the media profile, it will be replaced. The change shall be persistent. An NVT that supports video analytics shall support addition of video analytics configurations to a profile through the AddVideoAnalyticsConfiguration command.

Adding a VideoAnalyticsConfiguration to a media profile means that streams using that media profile can contain video analytics data (in the metadata) as defined by the submitted configuration reference. Video analytics data is specified in 17.1 and analytics configurations are managed through the commands defined in 0.

A profile containing only a video analytics configuration but no video source configuration is incomplete. Therefore, a client should first add a video source configuration to a profile before adding a video analytics configuration. The NVT can deny adding of a video analytics configuration before a video source configuration. In this case, it should respond with a ConfigurationConflict Fault.

Table 127 – AddVideoAnalytics command

AddVideoAnalytics		Request-Response
Message name	Description	
AddVideoAnalyticsRequest	Contains a reference to the VideoAnalytics to add and the Profile where it shall be added. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddVideoAnalyticsResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The VideoAnalytics indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:VideoAnalyticsNotSupported	<i>VideoAnalytics is not supported.</i>	

11.2.10 Add metadata configuration to a profile

This operation adds a Metadata configuration to an existing media profile (see Table 128). If a configuration exists in the media profile, it will be replaced. The change shall be persistent. An NVT shall support the addition of a metadata configuration to a profile through the AddMetadataConfiguration command.

Adding a MetadataConfiguration to a Profile means that streams using that profile contain metadata. Metadata can consist of events, PTZ status, and/or video analytics data. Metadata configurations are handled through the commands defined in 11.10 and 11.9.4.

Table 128 – AddMetadataConfiguration command

AddMetadataConfiguration		Request-Response
Message name	Description	
AddMetadataConfiguration Request	Contains a reference to the MetadataConfiguration to add and the Profile where it shall be added. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddMetadataConfiguration Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The MetadataConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	

11.2.11 Add audio output configuration

This operation adds an AudioOutputConfiguration to an existing media profile (see Table 129). If a configuration exists in the media profile, it will be replaced. The change shall be persistent. An NVT that has an Audio Output shall support addition of an audio output configuration to a profile through the AddAudioOutputConfiguration command.

Table 129 – AddAudioOutputConfiguration

AddAudioOutputConfiguration		Request-Response
Message name	Description	
AddAudioOutputConfiguration Request	<i>Contains a reference to the AudioOutputConfiguration to add and the Profile where it shall be added.</i> tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioOutputConfiguration Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgs ter:NoConfig	<i>The AudioOutputConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio or Audio Output is not supported</i>	

11.2.12 Add audio decoder configuration

This operation adds an AudioDecoderConfiguration to an existing media profile (see Table 130). If a configuration exists in the media profile, it shall be replaced. The change shall be persistent. An NVT that has audio decoding capabilities shall support addition of an audio decoder configuration to a profile through the AddAudioDecoderConfiguration command.

Table 130 – AddAudioDecoderConfiguration

AddAudioDecoderConfiguration		Request-Response
Message name	Description	
AddAudioDecoderConfiguration Request	<i>Contains a reference to the AudioConfiguration to add and the Profile where it shall be added.</i> tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioDecoderConfiguration Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgs ter:NoConfig	<i>The AudioDecoderConfiguration indicated by the ConfigurationToken does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile conflicts with the one to add and adding it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	<i>Audio or Audio Decoding is not supported</i>	

11.2.13 Remove video source configuration from a profile

This operation removes a VideoSourceConfiguration from an existing media profile (see Table 131). If the media profile does not contain a VideoSourceConfiguration, the operation has no effect. The removal shall be persistent. The NVT shall support removal of a video source configuration from a profile through the RemoveVideoSourceConfiguration command.

Video source configurations should only be removed after removing a VideoEncoderConfiguration from the media profile.

Table 131 – RemoveVideoSourceConfiguration command

RemoveVideoSourceConfiguration		Request-Response
Message name	Description	
RemoveVideoSourceConfiguration-Request	Contains a reference to the media profile from which the VideoSourceConfiguration shall be removed. tt:ReferenceToken ProfileToken [1][1]	
RemoveVideoSourceConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no video source configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the VideoSourceConfiguration and removing it would cause a conflicting media profile.</i>	

11.2.14 Remove video encoder configuration from a profile

This operation removes a VideoEncoderConfiguration from an existing media profile (see Table 132). If the media profile does not contain a VideoEncoderConfiguration, the operation has no effect. The removal shall be persistent. The NVT shall support removal of a video encoder configuration from a profile through the RemoveVideoEncoderConfiguration command.

Table 132 – RemoveVideoEncoderConfiguration command

RemoveVideoEncoderConfiguration		Request-Response
Message name	Description	
RemoveVideoEncoderConfiguration-Request	Contains a reference to the media profile from which the VideoEncoderConfiguration shall be removed. tt:ReferenceToken ProfileToken [1][1]	
RemoveVideoEncoderConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no video encoder configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the VideoEncoderConfiguration and removing it would cause a conflicting media profile.</i>	

11.2.15 Remove audio source configuration from a profile

This operation removes an AudioSourceConfiguration from an existing media profile (see Table 133). If the media profile does not contain an AudioSourceConfiguration, the operation has no effect. The removal shall be persistent. An NVT that supports audio streaming from NVT to client shall support removal of an audio source configuration from a profile through the RemoveAudioSourceConfiguration command.

Audio source configurations should only be removed after removing an AudioEncoderConfiguration from the media profile.

Table 133 – RemoveAudioSourceConfiguration command

RemoveAudioSourceConfiguration		Request-Response
Message name	Description	
RemoveAudioSourceConfiguration-Request	Contains a reference to the media profile from which the AudioSourceConfiguration shall be removed. tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioSourceConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no audio source configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the AudioSourceConfiguration and removing it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Audio is not supported.</i>	

11.2.16 Remove audio encoder configuration from a profile

This operation removes an AudioEncoderConfiguration from an existing media profile (see Table 134). If the media profile does not contain an AudioEncoderConfiguration, the operation has no effect. The removal shall be persistent. An NVT that supports audio streaming from NVT to client shall support removal of audio encoder configurations from a profile through the RemoveAudioEncoderConfiguration command.

Table 134 – RemoveAudioEncoderConfiguration command

RemoveAudioEncoderConfiguration		Request-Response
Message name	Description	
RemoveAudioEncoderConfiguration-Request	Contains a reference to the media profile from which the AudioEncoderConfiguration shall be removed. tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioEncoderConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no audio encoder configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the AudioEncoderConfiguration and removing it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Audio is not supported.</i>	

11.2.17 Remove PTZ configuration from a profile

This operation removes a PTZConfiguration from an existing media profile (see Table 135). If the media profile does not contain a PTZConfiguration, the operation has no effect. The removal shall be persistent. An NVT that supports PTZ control shall support removal of PTZ configurations from a profile through the RemovePTZConfiguration command.

Table 135 – RemovePTZConfiguration command

RemovePTZConfiguration		Request-Response
Message name	Description	
RemovePTZConfiguration-Request	Contains a reference to the media profile from which the PTZConfiguration shall be removed. tt:ReferenceToken ProfileToken [1][1]	
RemovePTZConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no PTZ configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the PTZConfiguration and removing it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

11.2.18 Remove video analytics configuration from a profile

This operation removes a VideoAnalyticsConfiguration from an existing media profile (see Table 136). If the media profile does not contain a VideoAnalyticsConfiguration, the operation has no effect. The removal shall be persistent. An NVT that supports video analytics shall support removal of a video analytics configuration from a profile through the RemoveVideoAnalyticsConfiguration command.

Table 136 – RemoveVideoAnalyticsConfiguration command

RemoveVideoAnalyticsConfiguration		Request-Response
Message name	Description	
RemoveVideoAnalyticsConfiguration-Request	Contains a reference to the media profile from which the VideoAnalyticsConfiguration shall be removed. tt:ReferenceToken ProfileToken [1][1]	
RemoveVideoAnalyticsConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no video analytics configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the VideoAnalyticsConfiguration and removing it would cause a conflicting media profile.</i>	
env:Receiver ter:ActionNotSupported ter:VideoAnalyticsNotSupported	<i>VideoAnalytics is not supported.</i>	

11.2.19 Remove metadata configuration from a profile

This operation removes a MetadataConfiguration from an existing media profile (see Table 137). If the media profile does not contain a MetadataConfiguration, the operation has no effect. The removal shall be persistent. An NVT shall support the removal of a metadata configuration from a profile through the RemoveMetadataConfiguration command.

Table 137 – RemoveMetadataConfiguration command

RemoveMetadataConfiguration		Request-Response
Message name	Description	
RemoveMetadataConfiguration-Request	Contains a reference to the media profile from which the MetadataConfiguration shall be removed. tt:ReferenceToken ProfileToken [1][1]	
RemoveMetadataConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no metadata configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the MetadataConfiguration and removing it would cause a conflicting media profile.</i>	

11.2.20 Remove audio output configuration

This operation removes an AudioOutputConfiguration from an existing media profile (see Table 138). If the media profile does not contain an AudioOutputConfiguration, the operation has no effect. The removal shall be persistent. An NVT that has at least one audio output shall support removal of an audio output configuration from a profile through the RemoveAudioOutputConfiguration command.

Table 138 – RemoveAudioOutputConfiguration

RemoveAudioOutputConfiguration		Request-Response
Message name	Description	
RemoveAudioOutputConfiguration-Request	Contains a reference to the media profile from which the AudioOutputConfiguration shall be removed.	
	tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioOutputConfiguration-Response	This is an empty message.	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	There exists no audio output configuration in the media profile.	
env:Receiver ter:Action ter:ConfigurationConflict	Other configurations of the media profile are dependant on the AudioOutputConfiguration and removing it would cause a conflicting media profile.	
env: Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	Audio or Audio output is not supported	

11.2.21 Remove audio decoder configuration

This operation removes an AudioDecoderConfiguration from an existing media profile (see Table 139). If the media profile does not contain an AudioDecoderConfiguration, the operation has no effect. The removal shall be persistent. An NVT that supports audio decoding shall support removal of an audio decoder configuration from a profile through the RemoveAudioDecoderConfiguration command.

Table 139 – RemoveAudioDecoderConfiguration

RemoveAudioDecoderConfiguration		Request-Response
Message name	Description	
RemoveAudioDecoderConfiguration-Request	<i>Contains a reference to the media profile from which the AudioDecoderConfiguration shall be removed.</i> tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioDecoderConfiguration-Response	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>There exists no audio decoder configuration in the media profile.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Other configurations of the media profile are dependant on the AudioDecoder Configuration and removing it would cause a conflicting media profile.</i>	
env: Receiver ter:ActionNotSupported ter::AudioDecodingNotSupported	<i>Audio or AudioDecoding is not supported</i>	

11.2.22 Delete media profile

This operation deletes a profile (see Table 140). This change shall always be persistent. The NVT shall support the deletion of a media profile through the DeleteProfile command.

Table 140 – DeleteProfile command

DeleteProfile		Request-Response
Message name	Description	
DeleteProfileRequest	<i>Contains a ProfileToken that indicates what media profile to delete.</i> tt:ReferenceToken ProfileToken [1][1]	
DeleteProfileResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:Action ter:DeletionOfFixedProfile	<i>The fixed Profile cannot be deleted.</i>	

11.3 Video source

11.3.1 General

A VideoSource represents unencoded video input. The structure contains the pixel resolution of the video, framerate and imaging settings. The imaging settings can be manipulated through the ImagingService if supported and contains parameters for focus, exposure and brightness, for example. See Clause 0 for more information.

11.3.2 GetVideoSources

This operation lists all available video sources for the device (see Table 141). The NVT shall support the listing of available video sources through the GetVideoSources command.

Table 141 – GetVideoSources command

GetVideoSources		Request-Response
Message name	Description	
GetVideoSourcesRequest	<i>This is an empty message.</i>	
GetVideoSourcesResponse	<i>Contains a list of structures describing all available video sources of the device.</i> tt:VideoSource VideoSources [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

11.4 Video source configuration

A VideoSourceConfiguration contains a reference to a VideoSource and a Bounds structure containing either the whole VideoSource pixel area or a sub-portion of it. The Bounds and VideoSource define the image that is streamed to a client. If a VideoSourceConfiguration is used inside a profile its UseCount parameter is increased to indicate that changing this configuration could affect other users.

11.4.1 Get video source configurations

This operation lists all *existing* video source configurations for an NVT. This command lists *all* video source configurations in a device. The client need not know anything about the video source configurations in order to use the command. The NVT shall support the listing of available video source configurations through the GetVideoSourceConfigurations command (see Table 142).

Table 142 – GetVideoSourceConfigurations command

GetVideoSourceConfigurations		Request-Response
Message name	Description	
GetVideoSourceConfigurations-Request	<i>This is an empty message.</i>	
GetVideoSourceConfigurations-Response	<i>This message contains a list of all existing video source configurations in the NVT. A video source configuration does always point at a real video source with the SourceToken element.</i> tt:VideoSourceConfiguration Configurations [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

11.4.2 Get video source configuration

If the video source configuration token is already known, the video source configuration can be fetched through the GetVideoSourceConfiguration command. The NVT shall support retrieval of specific video source configurations through the GetVideoSourceConfiguration command (see Table 143).

Table 143 – GetVideoSourceConfiguration command

GetVideoSourceConfiguration		Request-Response
Message name	Description	
GetVideoSourceConfiguration-Request	<i>This message contains the token of the requested video source configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoSourceConfiguration-Response	<i>This message contains the requested VideoSourceConfiguration with the matching token. A video source configuration does always point at a real video source with the SourceToken element.</i> tt:VideoSourceConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

11.4.3 Get compatible video source configurations

This operation requests all the video source configurations of the NVT that are compatible with a certain media profile. Each of the returned configurations shall be a valid input parameter for the AddVideoSourceConfiguration command on the media profile. The result will vary depending on the capabilities, configurations and settings in the device. The NVT shall support the listing of compatible (with a specific profile) video source configurations through the GetCompatibleVideoSourceConfigurations command (see Table 144).

Table 144 – GetCompatibleVideoSourceConfigurations command

GetCompatibleVideoSourceConfigurations		Request-Response
Message name	Description	
GetCompatibleVideoSource-ConfigurationsRequest	<i>Contains the token of an existing media profile.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleVideoSource-ConfigurationsResponse	<i>Contains a list of video source configurations that are compatible with the media profile.</i> tt:VideoSourceConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	

11.4.4 Get video source configuration options

This operation returns the available options when the video source parameters are reconfigured. If a video source configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. The NVT shall support the listing of available video source parameter options (for a given profile and configuration) through the GetVideoSourceConfigurationOptions command (see Table 145).

Table 145 – GetVideoSourceConfigurationOptions command

GetVideoSourceConfigurationOptions		Request-Response
Message name	Description	
GetVideoSourceConfiguration-OptionsRequest	This message contains optional tokens of a video source configuration and a media profile. ConfigurationToken specifies an existing configuration that the options are intended for. <i>ProfileToken specifies an existing media profile that the options shall be compatible with.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetVideoSourceConfiguration-OptionsResponse	<i>This message contains the video configuration options. If a video source configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. If no tokens are specified, the options shall be considered generic for the device.</i> tt:VideoSourceConfigurationOptions Options [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration does not exist.	

11.4.5 Modify a video source configuration

This operation modifies a video source configuration. The ForcePersistence flag indicates if the changes shall remain after reboot of the NVT. Running streams using this configuration may be immediately updated according to the new settings. The changes are not guaranteed to take effect unless the client requests a new stream URI and restarts any affected stream. NVC methods for changing a running stream are out of scope for this standard. The NVT shall support the modification of video source parameters through the SetVideoSourceConfiguration command (see Table 146).

Table 146 – SetVideoSourceConfiguration command

SetVideoSourceConfiguration		Request-Response
Message name	Description	
SetVideoSourceConfiguration-Request	<i>The Configuration element contains the modified video source configuration. The configuration shall exist in the NVT.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:VideoSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoSourceConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	

11.5 Video encoder configuration

A VideoEncoderConfiguration contains the following parameters for configuring the encoding of video data:

- Encoder – the encoding used for the video data;
- Resolution – the pixel resolution of the encoded video data;
- Quality – determines the quality of the video. A high value within supported quality range means higher quality;
- RateControl – defines parameters to configure the bitrate [kbps] as well as an EncodingInterval parameter (Interval at which images are encoded and transmitted) and a FrameRateLimit [fps] parameter to configure the output framerate;
- MPEG4/H264 specifics – defines the encoding profile and GOV length [frame].

The VideoEncoderConfiguration structure also contains multicast parameters and a session timeout to define video streaming behaviour. If a VideoEncoderConfiguration is used inside a profile its UseCount parameter is increased to indicate that changing this configuration could affect other users.

11.5.1 Get video encoder configurations

This operation lists all *existing* video encoder configurations of an NVT. This command lists *all* configured video encoder configurations in a device. The client need not know anything apriori about the video encoder configurations in order to use the command. The NVT shall support the listing of available video encoder configurations through the GetVideoEncoderConfigurations command (see Table 147).

Table 147 – GetVideoEncoderConfigurations command

GetVideoEncoderConfigurations		Request-Response
Message name	Description	
GetVideoEncoderConfigurations-Request	<i>This is an empty message.</i>	
GetVideoEncoderConfigurations-Response	<i>This message contains a list of all existing video encoder configurations in the NVT.</i> tt:VideoEncoderConfiguration Configurations [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

11.5.2 Get video encoder configuration

If the video encoder configuration token is already known, the encoder configuration can be fetched through the GetVideoEncoderConfiguration command. The NVT shall support the retrieval of a specific video encoder configuration through the GetVideoEncoderConfiguration command (see Table 148).

Table 148 – GetVideoEncoderConfiguration command

GetVideoEncoderConfiguration		Request-Response
Message name	Description	
GetVideoEncoderConfiguration-Request	<i>This message contains the token of the requested video encoder configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoEncoderConfiguration-Response	<i>This message contains the requested VideoEncoderConfiguration with the matching token.</i> tt:VideoEncoderConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

11.5.3 Get compatible video encoder configurations

This operation lists all the video encoder configurations of the NVT that are compatible with a certain media profile. Each of the returned configurations shall be a valid input parameter for the AddVideoEncoderConfiguration command on the media profile. The result will vary depending on the capabilities, configurations and settings in the device. The NVT shall support the listing of compatible (with a specific profile) video encoder configurations through the GetCompatibleVideoEncoderConfigurations command (see Table 149).

Table 149 – GetCompatibleVideoEncoderConfigurations command

GetCompatibleVideoEncoderConfigurations		Request-Response
Message name	Description	
GetCompatibleVideoEncoderConfigurationsRequest	<i>Contains the token of an existing media profile.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleVideoEncoderConfigurationsResponse	<i>Contains a list of video encoder configurations that are compatible with the given media profile.</i> tt:VideoEncoderConfiguration Configurations [0]unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	

11.5.4 Get video encoder configuration options

This operation returns the available options when the video encoder parameters are reconfigured. The NVT shall support the listing of available video parameter options (for a given profile and configuration) through the GetVideoEncoderConfigurationOptions command (see Table 150).

Table 150 – GetVideoEncoderConfigurationOptions command

GetVideoEncoderConfigurationOptions		Request-Response
Message name	Description	
GetVideoEncoderConfiguration-OptionsRequest	This message contains optional tokens of a video encoder configuration and a media profile. ConfigurationToken specifies an existing configuration that the options are intended for. <i>ProfileToken specifies an existing media profile that the options shall be compatible with.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetVideoEncoderConfiguration-OptionsResponse	<i>This message contains the video configuration options. If a video encoder configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. If no tokens are specified, the options shall be considered generic for the device.</i> tt:VideoEncoderConfigurationOptions Options [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration does not exist.	

11.5.5 Modify a video encoder configuration

This operation modifies a video encoder configuration. The ForcePersistence flag indicates if the changes shall remain after reboot of the NVT. Changes in the Multicast settings shall always be persistent. Running streams using this configuration may be immediately updated according to the new settings, but the changes are not guaranteed to take effect unless the client requests a new stream URI and restarts any affected stream. If the new settings invalidate any parameters already negotiated using RTSP, for example by changing codec type, the NVT shall not apply these settings to existing streams. Instead it shall either continue to stream using the old settings or stop sending data on the affected streams.

NVC methods for changing a running stream are out of scope for this standard. The NVT shall support the modification of video encoder parameters through the SetVideoEncoderConfiguration command (see Table 151).

Table 151 – SetVideoEncoderConfiguration command

SetVideoEncoderConfiguration		Request-Response
Message name	Description	
SetVideoEncoderConfiguration-Request	<i>The Configuration element contains the modified video encoder configuration. The configuration shall exist in the NVT.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:VideoEncoderConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoEncoderConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	

11.5.6 Get guaranteed number of video encoder instances

The GetGuaranteedNumberOfVideoEncoderInstances command can be used to request the minimum number of guaranteed video encoder instances (applications) per Video Source Configuration. An NVT shall support this command. This command was added in ONVIF 1.02. See Table 152.

Table 152 – GetGuaranteedNumberOfVideoEncoderInstances command

GetGuaranteedNumberOfVideoEncoderInstances		Request-Response
Message name	Description	
GetGuaranteedNumberOfEncoderInstancesRequest	<i>This request contains a token to the video source configuration.</i> tt: ReferenceToken ConfigurationToken [1][1]	
GetGuaranteedNumberOfEncoderInstancesResponse	<i>This message contains the minimum guaranteed TotalNumber of encoder instances (applications) per VideoSourceConfiguration. If a device limits the number of instances for respective Video Codecs the response contains the information how many Jpeg, H264 and Mpeg4 can be set up at the same time. In all other cases the device is able to deliver the TotalNumber of streams independent from the configured VideoCodec at the same time.</i> xs:int TotalNumber [1][1] xs:int JPEG [0][1] xs:int H264 [0][1] xs:int MPEG4 [0][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

11.6 Audio source

11.6.1 General

An AudioSource represents unencoded audio input and states the number of input channels.

11.6.2 Get audio sources

This operation lists all available audio sources of the device. An NVT that supports audio streaming from NVT to client shall support listing of available audio sources through the GetAudioSources command (see Table 153).

Table 153 – GetAudioSources command

GetAudioSources		Request-Response
Message name	Description	
GetAudioSourcesRequest	<i>This message is empty.</i>	
GetAudioSourcesResponse	<i>Contains a list of structures describing all available audio sources of the device.</i> tt:AudioSource AudioSources [0][unbounded]	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio.</i>	

11.7 Audio source configuration

An AudioSourceConfiguration contains a reference to an AudioSource that is to be used for input in a media profile. If an AudioSourceConfiguration is used inside a profile its UseCount parameter is increased to indicate that changing this configuration could affect other users.

11.7.1 Get audio source configurations

This operation lists all *existing* audio source configurations of an NVT. This command lists *all* audio source configurations in a device. The client need not know anything apriori about the audio source configurations in order to use the command. An NVT that supports audio streaming from NVT to client shall support listing of available audio source configurations through the GetAudioSourceConfigurations command (see Table 154).

Table 154 – GetAudioSourceConfigurations command

GetAudioSourceConfigurations		Request-Response
Message name	Description	
GetAudioSourceConfigurations-Request	<i>This is an empty message.</i>	
GetAudioSourceConfigurations-Response	<i>This message contains a list of all existing audio source configurations in the NVT. An audio source configuration does always point at a real audio source with the SourceToken element.</i> tt:AudioSourceConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio.</i>	

11.7.2 Get audio source configuration

The GetAudioSourceConfiguration command fetches the audio source configurations if the audio source configuration token is already known. An NVT that supports audio streaming from NVT to client shall support the retrieval of a specific audio source configuration through the GetAudioSourceConfiguration command (see Table 155).

Table 155 – GetAudioSourceConfiguration command

GetAudioSourceConfiguration		Request-Response
Message name	Description	
GetAudioSourceConfiguration-Request	<i>This message contains the token of the requested audio source configuration. An audio source configuration does always point at a real audio source with the SourceToken element.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioSourceConfiguration-Response	<i>This message contains the requested AudioSourceConfiguration with the matching token.</i> tt:AudioSourceConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio.</i>	

11.7.3 Get compatible audio source configurations

This operation requests all audio source configurations of a device that are compatible with a certain media profile. Each of the returned configurations shall be a valid input parameter for the AddAudioSourceConfiguration command on the media profile. The result varies depending on the capabilities, configurations and settings in the device. An NVT that supports audio streaming from NVT to client shall support listing of compatible (with a specific profile) audio source configurations through the GetCompatibleAudioSourceConfigurations command (see Table 156).

Table 156 – GetCompatibleAudioSourceConfigurations command

GetCompatibleAudioSourceConfigurations		Request-Response
Message name	Description	
GetCompatibleAudioSourceConfigurationsRequest	<i>Contains the token of an existing media profile.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioSourceConfigurationsResponse	<i>Contains a list of audio source configurations that are compatible with the media profile.</i> tt:AudioSourceConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio.</i>	

11.7.4 Get audio source configuration options

This operation returns the available options when the audio source parameters are reconfigured. If an audio source configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. An NVT that supports audio streaming from NVT to client shall support the listing of available audio parameter options (for a given profile and configuration) through the GetAudioSourceConfigurationOptions command (see Table 157).

Table 157 – GetAudioSourceConfigurationOptions command

GetAudioSourceConfigurationOptions		Request-Response
Message name	Description	
GetAudioSourceConfiguration-OptionsRequest	This message contains optional tokens of an audio source configuration and a media profile. ConfigurationToken specifies an existing configuration that the options are intended for. <i>ProfileToken specifies an existing media profile that the options shall be compatible with.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioSourceConfiguration-OptionsResponse	<i>This message contains the audio configuration options. If an audio source configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. If no tokens are specified, the options shall be considered generic for the device.</i> tt:AudioSourceConfigurationOptions Options [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration does not exist.	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT does not support audio.	

11.7.5 Modify an audio source configuration

This operation modifies an audio source configuration. The ForcePersistence flag indicates if the changes shall remain after reboot of the NVT. Running streams using this configuration may be immediately updated according to the new settings, but the changes are not guaranteed to take effect unless the client requests a new stream URI and restarts any affected stream. If the new settings invalidate any parameters already negotiated using RTSP, for example by changing codec type, the NVT shall not apply these settings to existing streams. Instead it shall either continue to stream using the old settings or stop sending data on the affected streams.

NVC methods for changing a running stream are out of scope for this standard. An NVT that supports audio streaming from NVT to client shall support the configuration of audio source parameters through the SetAudioSourceConfiguration command (see Table 158).

Table 158 – SetAudioSourceConfiguration command

SetAudioSourceConfiguration		Request-Response
Message name	Description	
SetAudioSourceConfiguration-Request	<i>The Configuration element contains the modified audio source configuration. The configuration shall exist in the NVT.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AudioSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioSourceConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio.</i>	

11.8 Audio encoder configuration

An AudioEncoderConfiguration contains the following parameters for encoding audio data:

- Encoder – the encoding used for audio data;
- Bitrate – the output bitrate [kbps];
- SampleRate – the output sample rate [kHz].

The AudioEncoderConfiguration structure also contains multicast parameters and a session timeout to define audio streaming behaviour.

If an AudioEncoderConfiguration is used inside a profile its UseCount parameter is increased to indicate that changing this configuration could affect other users.

11.8.1 Get audio encoder configurations

This operation lists all *existing* device audio encoder configurations. The client need not know anything apriori about the audio encoder configurations in order to use the command. An NVT that supports audio streaming from NVT to client shall support the listing of available audio encoder configurations through the GetAudioEncoderConfigurations command (see Table 159).

Table 159 – GetAudioEncoderConfigurations command

GetAudioEncoderConfigurations		Request-Response
Message name	Description	
GetAudioEncoderConfigurations-Request	<i>This is an empty message.</i>	
GetAudioEncoderConfigurations-Response	<i>This message contains a list of all existing audio encoder configurations in the NVT.</i> tt:AudioEncoderConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Receiver	<i>NVT does not support audio.</i>	
ter:ActionNotSupported		
ter:AudioNotSupported		

11.8.2 Get audio encoder configuration

The GetAudioEncoderConfiguration command fetches the encoder configuration if the audio encoder configuration token is known. An NVT that supports audio streaming from NVT to client shall support the listing of a specific audio encoder configuration through the GetAudioEncoderConfiguration command (see Table 160).

Table 160 – GetAudioEncoderConfiguration command

GetAudioEncoderConfiguration		Request-Response
Message name	Description	
GetAudioEncoderConfiguration-Request	<i>This message contains the token of the requested audio encoder configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioEncoderConfiguration-Response	<i>This message contains the requested AudioEncoderConfiguration with the matching token.</i> tt:AudioEncoderConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio.</i>	

11.8.3 Get compatible audio encoder configurations

This operation requests all audio encoder configurations of the NVT that are compatible with a certain media profile. Each of the returned configurations shall be a valid input parameter for the AddAudioEncoderConfiguration command on the media profile. The result varies depending on the capabilities, configurations and settings in the device. An NVT that supports audio streaming from NVT to client shall support listing of compatible (with a specific profile) audio encoder configurations through the GetCompatibleAudioEncoderConfigurations command (see Table 161).

Table 161 – GetCompatibleAudioEncoderConfigurations command

GetCompatibleAudioEncoderConfigurations		Request-Response
Message name	Description	
GetCompatibleAudioEncoderConfigurationsRequest	<i>Contains the token of an existing media profile.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioEncoderConfigurationsResponse	<i>Contains a list of audio encoder configurations that are compatible with the given media profile.</i> tt:AudioEncoderConfiguration Configurations [0]unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio</i>	

11.8.4 Get audio encoder configuration options

This operation returns the available options when the audio encoder parameters are reconfigured. An NVT that supports audio streaming from NVT to client shall support the listing of available audio encoder parameter options (for a given profile and configuration) through the GetAudioEncoderConfigurationOptions command (see Table 162).

Table 162 – GetAudioEncoderConfigurationOptions command

GetAudioEncoderConfigurationOptions		Request-Response
Message name	Description	
GetAudioEncoderConfiguration-OptionsRequest	This message contains optional tokens of an audio encoder configuration and a media profile. ConfigurationToken specifies an existing configuration that the options are intended for. <i>ProfileToken specifies an existing media profile that the options shall be compatible with.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioEncoderConfiguration-OptionsResponse	<i>This message contains the audio configuration options. If a audio encoder configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. If no tokens are specified, the options shall be considered generic for the device.</i> tt:AudioEncoderConfigurationOptions Options [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration does not exist.	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	NVT does not support audio.	

11.8.5 Modify audio encoder configurations

This operation modifies an audio encoder configuration. The ForcePersistence flag indicates if the changes shall remain after reboot of the NVT. Changes in the Multicast settings shall always be persistent. Running streams using this configuration may be immediately updated according to the new settings. The changes are not guaranteed to take effect unless the client requests a new stream URI and restarts any affected streams. NVC methods for changing a running stream are out of scope for this standard. An NVT that supports audio streaming from NVT to client shall support the configuration of audio encoder parameters through the SetAudioEncoderConfiguration command (see Table 163).

Table 163 – SetAudioEncoderConfiguration command

SetAudioEncoderConfiguration		Request-Response
Message name	Description	
SetAudioEncoderConfiguration-Request	<i>The Configuration element contains the modified audio encoder configuration. The configuration shall exist in the NVT.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AudioEncoderConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioEncoderConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>NVT does not support audio.</i>	

11.9 Video analytics configuration

VideoAnalyticsConfiguration contains parameters for an *analytics engine* and a *rule engine* (see 4.12). Thereby, the analytics engine consists of multiple modules which can be managed by the analytics module part of the analytics service. Similarly, the rule engine consists of

multiple rules which can be managed by the rule engine part of the analytics service. The subsequent commands are introduced to handle complete video analytics configuration in an atomic way. For instance, the ModifyVideoAnalyticsConfiguration command changes analytics and rule engine configuration in an atomic operation. When a video analytics configuration is present in a profile, the metadata configuration can activate the streaming of the scene description within the RTP streams (see 11.10).

A device MAY NOT allow referencing the very same VideoAnalyticsConfiguration from multiple media profiles with different VideoSourceConfigurations. If the device allows it, it shall generate individual scene descriptions for each profile, since the coordinate system of a scene description relates to a specific VideoSourceConfiguration. Also masking and geometrical rules relate to the coordinate system of the VideoSourceConfiguration. This MAY require separate processing of the whole video analytics for each VideoSourceConfiguration, even if they refer to the very same VideoSource.

Since the options of a VideoAnalyticsConfiguration are dynamic and often vendor specific, they can only be retrieved via the video analytics service.

11.9.1 Get video analytics configurations

This operation lists all video analytics configurations of a device. This command lists *all* configured video analytics in a device. The client need not know anything apriori about the video analytics in order to use the command. A device that supports video analytics shall support the listing of available video analytics configuration through the GetVideoAnalyticsConfigurations command (see Table 164).

Table 164 – GetVideoAnalyticsConfigurations command

GetVideoAnalyticsConfigurations		Request-Response
Message name	Description	
GetVideoAnalyticsConfigurations-Request	<i>This message is empty.</i>	
GetVideoAnalyticsConfigurations-Response	<i>This message contains a list of all existing video analytics configurations in the device.</i>	
	tt:VideoAnalyticsConfiguration [0][unbounded]	Configurations
Fault codes	Description	
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	<i>Device does not support video analytics.</i>	

11.9.2 Get video analytics configuration

The GetVideoAnalyticsConfiguration command fetches the video analytics configuration if the video analytics token is known. A device that supports video analytics shall support the listing of a specific video analytics configuration through the GetVideoAnalyticsConfiguration command (see Table 165).

Table 165 – GetVideoAnalyticsConfiguration command

GetVideoAnalyticsConfiguration		Request-Response
Message name	Description	
GetVideoAnalyticsConfiguration-Request	<i>This message contains the token of an existing video analytics configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoAnalyticsConfiguration-Response	<i>This message contains the requested video analytics configuration.</i> tt:VideoAnalyticsConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	<i>The device does not support video analytics.</i>	

11.9.3 Get compatible video analytics configurations

This operation requests all video analytic configurations of the device that are compatible with a certain media profile. Each of the returned configurations shall be a valid input parameter for the AddVideoAnalyticsConfiguration command on the media profile. The result varies depending on the capabilities, configurations and settings in the device. A device that supports video analytics shall support the listing of compatible (with a specific profile) video analytics configuration through the GetCompatibleVideoAnalyticsConfigurations command (see Table 166).

Table 166 – GetCompatibleVideoAnalyticsConfigurations command

GetCompatibleVideoAnalyticsConfigurations		Request-Response
Message name	Description	
GetCompatibleVideoAnalytics-ConfigurationsRequest	<i>Contains the token of an existing media profile.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleVideoAnalytics-ConfigurationsResponse	<i>Contains a list of video analytics configurations that are compatible with the given media profile.</i> tt:VideoAnalyticsConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	<i>The device does not support video analytics.</i>	

11.9.4 Modify a video analytics configuration

A video analytics configuration is modified using this command. The ForcePersistence flag indicates if the changes shall remain after reboot of the device or not. Running streams using this configuration shall be immediately updated according to the new settings. Otherwise inconsistencies can occur between the scene description processed by the rule engine and the notifications produced by analytics engine and rule engine which reference the very same video analytics configuration token. A device that supports video analytics shall support the configuration of video analytics parameters through the SetVideoAnalyticsConfiguration command (see Table 167).

Table 167 – SetVideoAnalyticsConfiguration command

SetVideoAnalyticsConfiguration		Request-Response
Message name	Description	
SetVideoAnalyticsConfiguration-Request	<i>The Configuration element contains the modified video analytics configuration. The configuration shall exist in the device.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:VideoAnalyticsConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoAnalyticsConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	<i>The device does not support video analytics.</i>	

11.10 Metadata configuration

A MetadataConfiguration contains parameters for selecting the data to include in the metadata stream. The choices include PTZ status, PTZ position, events as defined by a subscription and analytics data. The event subscription data is described in 15.5. The analytics parameters define which data to include from the analytics engine part of the profile, see 0.

The structure also contains multicast parameters used to configure and control multicast of the metadata stream. A session timeout parameter defines the session timeout (see 0)

If a MetadataConfiguration is used inside a profile its UseCount parameter is increased to indicate that changing this configuration could affect other users.

11.10.1 Get metadata configurations

This operation lists all *existing* metadata configurations. The client need not know anything apriori about the metadata in order to use the command. A NVT or another device that supports metadata streaming shall support the listing of existing metadata configurations through the GetMetadataConfigurations command (see Table 168).

Table 168 – GetMetadataConfigurations command

GetMetadataConfigurations		Request-Response
Message name	Description	
GetMetadataConfigurations-Request	<i>This message is empty.</i>	
GetMetadataConfigurations-Response	<i>This message contains a list of all existing metadata configurations in the device.</i> tt:MetadataConfiguration Configurations [0][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

11.10.2 Get metadata configuration

The GetMetadataConfiguration command fetches the metadata configuration if the metadata token is known. A NVT or another device that supports metadata streaming shall support the listing of a specific metadata configuration through the GetMetadataConfiguration command (see Table 169).

Table 169 – GetMetadataConfiguration command

GetMetadataConfiguration		Request-Response
Message name	Description	
GetMetadataConfiguration-Request	<i>This message contains the token of an existing metadata configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetMetadataConfiguration-Response	<i>This message contains the requested metadata configuration.</i> tt:MetadataConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

11.10.3 Get compatible metadata configurations

This operation requests all the metadata configurations of the device that are compatible with a certain media profile. Each of the returned configurations shall be a valid input parameter for the AddMetadataConfiguration command on the media profile. The result varies depending on the capabilities, configurations and settings in the device. A NVT or other device that supports metadata streaming shall support the listing of compatible (with a specific profile) metadata configuration through the GetCompatibleMetadataConfigurations command (see Table 170).

Table 170 – GetCompatibleMetadataConfigurations command

GetCompatibleMetadataConfigurations		Request-Response
Message name	Description	
GetCompatibleMetadataConfigurationsRequest	<i>Contains the token of an existing media profile.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleMetadataConfigurationsResponse	<i>Contains a list of metadata configurations that are compatible with the given media profile.</i> tt:MetadataConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	

11.10.4 Get metadata configuration options

This operation returns the available options for changing the metadata configuration. A NVT or another device that supports metadata streaming shall support the listing of available metadata parameter options (for a given profile and configuration) through the GetMetadataConfigurationOptions command (see Table 171).

Table 171 – GetMetadataConfigurationOptions command

GetMetadataConfigurationOptions		Request-Response
Message name	Description	
GetMetadataConfiguration-OptionsRequest	This message contains optional tokens of a metadata configuration and a media profile. ConfigurationToken specifies an existing configuration that the options are intended for. ProfileToken specifies an existing media profile that the options shall be compatible with. tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetMetadataConfiguration-OptionsResponse	<i>This message contains the metadata configuration options. If a metadata configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. If no tokens are specified, the options shall be considered generic for the device.</i> tt:MetadataConfigurationOptions Options [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration does not exist.	

11.10.5 Modify a metadata configuration

This operation modifies a metadata configuration. The ForcePersistence flag indicates if the changes shall remain after reboot of the device. Changes in the Multicast settings shall always be persistent. Running streams using this configuration may be updated immediately according to the new settings. The changes are not guaranteed to take effect unless the client requests a new stream URI and restarts any affected streams. NVC methods for changing a running stream are out of scope for this standard. A NVT or another device that supports metadata streaming shall support the configuration of metadata parameters through the SetMetadataConfiguration command (see Table 172).

Table 172 – etMetadataConfiguration command

SetMetadataConfiguration		Request-Response
Message name	Description	
SetMetadataConfiguration-Request	<i>The Configuration element contains multicast settings as well as a set of filters determining what data to include in the metadata stream.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:MetadataConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetMetadataConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	

11.11 Audio outputs

11.11.1 General

The Audio Output represents the physical audio outputs that can be connected to a loudspeaker.

11.11.2 Get audio outputs

This command lists all available audio outputs of a device. An NVT that has one or more physical audio outputs shall support listing of available audio outputs through the GetAudioOutputs command (see Table 173).

Table 173 – GetAudioOutputs

GetAudioOutputs		Request-Response
Message name	Description	
GetAudioOutputsRequest	<i>This is an empty message.</i>	
GetAudioOutputsResponse	<i>Contains a list of structures describing all available audio outputs of the device. If a device has no AudioOutputs an empty list is returned.</i> tt:AudioOutput AudioOutputs [0][unbounded]	
Fault codes	Description	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio or Audio Outputs are not supported by the NVT</i>	

11.12 Audio output configuration

The audio output configuration contains the following parameters:

- SourceToken: a reference to an existing audio output.
- OutputLevel: a parameter to configure the output volume.
- SendPrimacy: a parameter that can be used for NVTs with a half duplex audio in/output to configure the active transmission direction (see 11.14).

If an AudioOutputConfiguration is used inside a profile its UseCount parameter is increased to indicate that changing this configuration could affect other users.

11.12.1 Get audio output configurations

This command lists all existing AudioOutputConfigurations of a device. The NVC need not know anything apriori about the audio configurations to use this command. An NVT that is able to output audio shall support the listing of AudioOutputConfigurations through this command (see Table 174).

Table 174 – GetAudioOutputConfiguration

GetAudioOutputConfigurations		Request-Response
Message name	Description	
GetAudioOutputConfigurationsRequest	<i>This is an empty message.</i>	
GetAudioOutputConfigurationsResponse	<i>Contains a list of AudioOutputConfigurations that are available on the device</i> tt:AudioOutputConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env: Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio or Audio Outputs are not supported by the device</i>	

11.12.2 Get audio output configuration

If the audio output configuration token is already known, the output configuration can be fetched through the GetAudioOutputConfiguration command. An NVT that has one or more audio outputs shall support the retrieval of a specific audio output configuration through the GetAudioOutputConfiguration command (see Table 175).

Table 175 – GetAudioOutputConfiguration

GetAudioOutputConfiguration		Request-Response
Message name	Description	
GetAudioOutputConfigurationRequest	<i>This message contains the token of the requested AudioOutput configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioOutputConfigurationResponse	<i>This message contains the requested AudioOutputConfiguration with the matching token.</i> tt:AudioOutputConfiguration Configuration [1][1]	
Fault codes	Description	
env: Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	
env: Receiver ter:ActionNotSupported ter::AudioOutputNotSupported	<i>Audio or Audio Outputs are not supported by the device</i>	

11.12.3 Get compatible audio output configurations

This command lists all audio output configurations of a device that are compatible with a certain media profile. Each returned configuration shall be a valid input for the AddAudioOutputConfiguration command. An NVT that has one or more audio outputs shall support the listing of compatible (with a specific profile) AudioOutputConfigurations through the GetCompatibleAudioOutputConfigurations command (see Table 176).

Table 176 – GetCompatibleAudioOutputConfiguration

GetCompatibleAudioOutputConfigurations		Request-Response
Message name	Description	
GetCompatibleAudioOutputConfigurations Request	Contains the token of an existing media profile. tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioOutputConfigurations Response	Contains a list of audio output configurations that are compatible with the given media profile. tt:AudioOutputConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	Audio or Audio Outputs are not supported by the device	

11.12.4 Get audio output configuration options

This operation returns the available options for configuring an audio output. An NVT that has one or more audio outputs shall support the listing of available audio output configuration options (for a given profile and configuration) through the GetAudioOutputConfigurationOptions command (see Table 177).

Table 177 – GetAudioOutputConfigurationOptions

GetAudioOutputConfigurationOptions		Request-Response
Message name	Description	
GetAudioOutputConfigurationOptionsRequest	<i>This message contains optional tokens of an audio output configuration and a media profile.</i> <i>ConfigurationToken specifies an existing configuration that the options are intended for.</i> <i>ProfileToken specifies an existing media profile that the options shall be compatible with.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioOutputConfigurationOptionsResponse	<i>This message contains the audio output configuration options. If a audio output configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. If no tokens are specified, the options shall be considered generic for the device.</i> tt:AudioOutputConfigurationOptions Options [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration does not exist.	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	Audio or Audio Outputs are not supported by the device	

11.12.5 Modify audio output configuration

This operation modifies an audio output configuration. The ForcePersistence flag indicates if the changes shall remain after reboot of the device. An NVT that has one or more audio outputs shall support the modification of audio output parameters through the SetAudioOutputConfiguration command (see Table 178).

Table 178 – SetAudioOutputConfiguration

SetAudioOutputConfiguration		Request-Response
Message name	Description	
SetAudioOutputConfiguration-Request	<i>The Configuration element contains the modified Audio Output configuration. The configuration shall exist in the device.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AudioOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioOutputConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	
env: Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio or Audio Outputs are not supported by the device</i>	

11.13 Audio decoder configuration

The Audio Decoder Configuration does not contain any that parameter to configure the decoding. A decoder shall decode every data it receives (according to its capabilities).

If an AudioDecoderConfiguration is used inside a profile its UseCount parameter is increased to indicate that changing this configuration could affect other users.

11.13.1 Get audio decoder configurations

This command lists all existing AudioDecoderConfigurations of a device.

The NVC need not know anything apriori about the audio decoder configurations in order to use this command. An NVT that is able to decode audio shall support the listing of AudioOutputConfigurations through this command (see Table 179).

Table 179 – GetAudioDecoderConfigurations

GetAudioDecoderConfigurations		Request-Response
Message name	Description	
GetAudioDecoderConfigurationsRequest	<i>This is an empty message.</i>	
GetAudioDecoderConfigurationsResponse	<i>Contains a list of AudioDecoderConfigurations that are available on the device</i> tt:AudioDecoderConfiguration [0][unbounded] Configurations	
Fault codes		Description
env:Receiver		<i>Audio or Audio decoding is not supported by the device</i>
ter:ActionNotSupported		
ter:AudioDecodingNotSupported		

11.13.2 Get audio decoder configuration

If the audio decoder configuration token is already known, the decoder configuration can be fetched through the GetAudioDecoderConfiguration command. An NVT that is able to decode audio shall support the retrieval of a specific audio decoder configuration through the GetAudioDecoderConfiguration command (see Table 180).

Table 180 – GetAudioDecoderConfiguration

GetAudioDecoderConfiguration		Request-Response
Message name	Description	
GetAudioDecoderConfigurationRequest	<i>This message contains the token of the requested AudioDecoder configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioDecoderConfigurationResponse	<i>This message contains the requested AudioDecoder Configuration with the matching token.</i> tt:AudioDecoderConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	<i>Audio or Audio decoding is not supported by the device</i>	

11.13.3 Get compatible audio decoder configurations

This operation lists all the audio decoder configurations of the device that are compatible with a certain media profile. Each of the returned configurations shall be a valid input parameter for the AddAudioDecoderConfiguration command on the media profile. An NVT that is able to decode audio shall support the listing of compatible (with a specific profile) audio decoder configurations through the GetCompatibleAudioDecoderConfigurations command (see Table 181).

Table 181 – GetCompatibleAudioDecoderConfigurations

GetCompatibleAudioDecoderConfigurations		Request-Response
Message name	Description	
GetCompatibleAudioDecoderConfigurations Request	Contains the token of an existing media profile. tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioDecoderConfigurations Response	Contains a list of audiodecoder configurations that are compatible with the given media profile. tt:AudioDecoderConfiguration Configurations [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	Audio or Audio decoding is not supported by the device	

11.13.4 Get audio decoder configuration options

This command list the audio decoding capabilities for a given profile and configuration of a device. A device that is able to decode audio shall support the retrieval of AudioDecoderConfigurationOptions through this command (see Table 182).

Table 182 – GetAudioDecoderConfigurationOptions

GetAudioDecoderConfigurationOptions		Request-Response
Message name	Description	
GetAudioDecoderConfiguration-OptionsRequest	This message contains optional tokens of a audio decoder configuration and a media profile. ConfigurationToken specifies an existing configuration that the options are intended for. <i>ProfileToken specifies an existing media profile that the options shall be compatible with.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioDecoderConfiguration-OptionsResponse	<i>This message contains the audio decoder configuration options. If a audio decoder configuration is specified, the options shall concern that particular configuration. If a media profile is specified, the options shall be compatible with that media profile. If no tokens are specified, the options shall be considered generic for the device.</i> tt:AudioDecoderConfigurationOptions Options [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The requested profile token ProfileToken does not exist.	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration does not exist.	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	Audio or Audio decoding is not supported by the device	

11.13.5 Modify audio decoder configuration

This operation modifies an audio decoder configuration. The ForcePersistence flag indicates if the changes shall remain after reboot of the device. The device that is able to decode audio shall support the modification of audio decoder parameters through the SetAudioDecoderConfiguration command (see Table 183).

Table 183 – SetAudioDecoderConfiguration

SetAudioDecoderConfiguration		Request-Response
Message name	Description	
SetAudioDecoderConfiguration-Request	<i>The Configuration element contains the modified AudioDecoder configuration. The configuration shall exist in the device.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AudioDecoderConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioDecoderConfiguration-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflicts with other uses of the configuration.</i>	
env: Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	<i>Audio or Audio decoding is not supported by the device</i>	

11.14 Audio channel modes

An audio channel MAY support different types of audio transmission. While for full duplex operation no special handling is required, in half duplex operation the transmission direction needs to be switched.

An optional Send-Primacy Parameter inside the AudioOutputConfiguration indicates which direction is currently active. An NVC can switch between different modes by setting the AudioOutputConfiguration.

The following modes for the Send-Primacy are defined:

- www.onvif.org/ver20/HalfDuplex/Server
The server is allowed to send audio data to the client. The client shall not send audio data via the backchannel to the NVT in this mode.
- www.onvif.org/ver20/HalfDuplex/Client
The client is allowed to send audio data via the backchannel to the server. The NVT shall not send audio data to the client in this mode.
- www.onvif.org/ver20/HalfDuplex/Auto
It is up to the device how to deal with sending and receiving audio data.

Acoustic echo cancellation is out of ONVIF scope.

11.15 Stream URI

11.15.1 General

Media stream can be set up and controlled by a RTSP protocol.

11.15.2 Request stream URI

This operation requests a URI that can be used to initiate a live media stream using RTSP as the control protocol. The returned URI shall remain valid indefinitely even if the profile is changed. The ValidUntilConnect, ValidUntilReboot and Timeout Parameter shall be set accordingly (ValidUntilConnect=false, ValidUntilReboot=false, timeout=PT0S). An NVT shall support the retrieval of a media stream URI for a specific media profile through the GetStreamUri command (see Table 184).

For full compatibility with other ONVIF services a device should not generate Uris longer than 128 octets.

Table 184 – GetStreamUri command

GetStreamUri		Request-Response
Message name	Description	
GetStreamUriRequest	<i>The StreamSetup element contains two parts. StreamType defines if a unicast or multicast media stream is requested. Transport specifies a chain of transport protocols defining the tunnelling of the media stream over different network protocols.</i> <i>The ProfileToken element indicates the media profile to use and will define the configuration of the content of the stream.</i> tt:StreamSetup StreamSetup [1][1] tt:ReferenceToken ProfileToken [1][1]	
GetStreamUriResponse	<i>Contains the stable Uri to be used for requesting the media stream as well as parameters defining the lifetime of the Uri. The ValidUntilConnect and ValidUntilReboot parameter shall be set to false, the timeout parameter shall be set to PT0S to indicate that this stream Uri is indefinitely valid even if the profile changes.</i> xs:anyURI Uri [1][1] xs:boolean InvalidAfterConnect [1][1] xs:boolean InvalidAfterReboot [1][1] xs:duration Timeout [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The media profile does not exist.	
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	Specification of StreamType or Transport part in StreamSetup is not supported.	
env:Sender ter:OperationProhibited ter:StreamConflict	Specification of StreamType or Transport part in StreamSetup causes conflict with other streams.	
env:Receiver ter:Action ter:IncompleteConfiguration	The specified media profile does contain either unused sources or encoder configurations without a corresponding source.	

11.16 Snapshot

11.16.1 General

A single snapshot image, consisting of one frame of video can be obtained.

11.16.2 Request snapshot URI

A Network client uses the GetSnapshotUri command to obtain a JPEG snhapshot from the NVT. The returned URI shall remain valid indefinitely even if the profile is changed. The ValidUntilConnect, ValidUntilReboot and Timeout Parameter shall be set accordingly (ValidUntilConnect=false, ValidUntilReboot=false, timeout=PT0S). The URI can be used for acquiring a JPEG image through a HTTP GET operation. The image encoding will always be JPEG regardless of the encoding setting in the media profile. A NVT shall support this command (see Table 185).

Table 185 – GetSnapshotUri command

GetSnapshotUri		Request-Response
Message name	Description	
GetSnapshotUriRequest	<i>The ProfileToken element indicates the media profile to use and will define the source and dimensions of the snapshot.</i> tt:ReferenceToken ProfileToken [1][1]	
GetSnapshotUriResponse	<i>Contains a stable Uri to be used for acquiring a snapshot in JPEG format as well as parameters defining the lifetime of the Uri. The ValidUntilConnect and ValidUntilReboot parameter shall be set to false, the timeout parameter shall be set to PT0S to indicate that this stream URI is indefinitely valid even if the profile changes.</i> xs:anyURI Uri [1][1] xs:boolean InvalidAfterConnect [1][1] xs:boolean InvalidAfterReboot [1][1] xs:duration Timeout [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The media profile does not exist.</i>	
env:Receiver ter:Action ter:IncompleteConfiguration	<i>The specified media profile does not contain either a reference to a video encoder configuration or a reference to a video source configuration.</i>	

11.17 Multicast

See 12.1 for a detailed discussion of NVT and client multicast streaming.

11.17.1 Start multicast streaming

This command starts multicast streaming using a specified media profile of an NVT. Streaming continues until StopMulticastStreaming is called for the same Profile. The streaming shall continue after a reboot of the NVT until a StopMulticastStreaming request is received. The multicast address, port and TTL are configured in the VideoEncoderConfiguration, AudioEncoderConfiguration and MetadataConfiguration respectively. An NVT that supports video, audio or metadata multicast streaming shall support the starting of a multicast stream through the StartMulticastStreaming command (see Table 186).

Table 186 – StartMulticastStreaming command

StartMulticastStreaming		Request-Response
Message name	Description	
StartMulticastStreaming-Request	<i>Contains the token of the Profile that is used to define the multicast stream.</i> tt:ReferenceToken ProfileToken [1][1]	
StartMulticastStreaming-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender	<i>The profile does not exist.</i>	
ter:InvalidArgVal		
ter:NoProfile		
env:Receiver	<i>The specified media profile does not contain either a reference to a video encoder, a video source configuration, a audio source or to audio encoder configuration or a reference to a metadata configuration</i>	
ter:Action		
ter:IncompleteConfiguration		

11.17.2 Stop multicast streaming

This command stop multicast streaming using a specified media profile of an NVT. An NVT that supports video, audio or metadata multicast streaming shall support the stopping of a multicast stream through the StopMulticastStreaming command (see Table 187).

Table 187 – StopMulticastStreaming command

StopMulticastStreaming		Request-Response
Message name	Description	
StopMulticastStreaming-Request	<i>Contains the token of the Profile that is used to define the multicast stream.</i> tt:ReferenceToken ProfileToken [1][1]	
StopMulticastStreaming-Response	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The profile does not exist.</i>	
env:Receiver ter:Action ter:IncompleteConfiguration	<i>The specified media profile does not contain either a reference to a video encoder, a video source configuration, a audio source or to audio encoder configuration or a reference to a metadata configuration</i>	

11.18 Synchronization points

11.18.1 General

Synchronization points allow clients to decode and correctly use all data after the synchronization point.

11.18.2 Set synchronization point

For example, if a video stream is configured with a large I-frame distance and a client loses a single packet, the client does not display video until the next I-frame is transmitted. In such cases, the client can request a Synchronization Point which enforces the NVT to add an I-Frame as soon as possible. Clients can request Synchronization Points for profiles. The NVT shall add synchronization points for all streams associated with this profile.

Similarly, a synchronization point is used to get an update on full PTZ or event status through the metadata stream.

If a video stream is associated with the profile, an I-frame shall be added to this video stream. If an event stream is associated to the profile, the synchronization point request shall be handled as described in 15.6). If a PTZ metadata stream is associated to the profile, the PTZ position shall be repeated within the metadata stream.

An NVT that supports MPEG-4 or H.264 shall support the request for an I-Frame through the SetSynchronizationPoint command (see Table 188).

Table 188 – SetSynchronizationPoint command

SetSynchronizationPoint		Request-response
Message name	Description	
SetSynchronizationPointRequest	Contains a Profile reference for which a Synchronization Point is requested. tt:ReferenceToken ProfileToken [1][1]	
SetSynchronizationPointResponse	This message is empty.	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	The profile does not exist.	

11.19 Service specific fault codes

The Table 189 below lists the media service specific fault codes. Additionally, each command can also generate a generic fault, see Table 6.

The specific faults are defined as subcode of a generic fault, see 5.11.2.1. The parent generic subcode is the *subcode* at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 189 – Media service specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Receiver	ter:ActionNotSupported	No audio capability	NVT does not support audio.
	ter:AudioNotSupported		
env:Receiver	ter:Action	Maximum number reached	The maximum number of supported profiles has been reached.
	ter:MaxNVTProfiles		
env:Receiver	ter:ActionNotSupported	No audio output capability	Audio or Audio Outputs are not supported by the NVT
	ter:AudioOutputNotSupported		
env:Receiver	ter:ActionNotSupported	No audio decoding capability	Audio or Audio Decoding is not supported by the NVT
	ter:AudioDecodingNotSupported		
env:Receiver	ter:Action	Configuration not complete	Entities required by this action are missing in the specified profile.
	ter:IncompleteConfiguration		
env:Receiver	ter:Action	Conflict when using new settings	The new settings conflicts with other uses of the configuration.
	ter:ConfigurationConflict		
env:Sender	ter:InvalidArgVal	Profile token already exists	A profile with the token ProfileToken already exists.
	ter:ProfileExists		
env:Sender	ter:InvalidArgVal	Configuration token does not exist	The requested configuration indicated by the ConfigurationToken does not exist.
	ter:NoConfig		
env:Sender	ter:InvalidArgVal	Profile token does not exist	The requested profile token ProfileToken does not exist.
	ter:NoProfile		
env:Sender	ter:Action	Fixed profile cannot be deleted	The fixed Profile cannot be deleted.
	ter:DeletionOfFixedProfile		
env:Sender	ter:InvalidArgVal	Parameters cannot be set	The configuration parameters are not possible to set.
	ter:ConfigModify		
env:Sender	ter:ActionNotSupported	No video analytics capability	NVT does not support video analytics.
	ter:VideoAnalyticsNot-Supported		
env:Sender	ter:InvalidArgVal	Invalid Stream setup	Specification of StreamType or Transport part in StreamSetup is not supported.
	ter:InvalidStreamSetup		
env:Sender	ter:OperationProhibited	Stream conflict	Specification of StreamType or Transport part in StreamSetup causes conflict with other streams.
	ter:StreamConflict		

12 Real time streaming

This clause describes real-time streaming of video, audio and metadata. There is *no specific* service associated with the real-time streaming. The real-time configurations via Web Service commands are defined in the Media Service and the ReceiverService.

12.1 Media stream protocol

12.1.1 Transport format

Real-time Transport Protocol (RTP) is a media transfer protocol (see 12.1.2). The following four subclauses describe RTP data transfer.

12.1.1.1 RTP data transfer via UDP

UDP has the smallest overhead and is able to transfer real-time data in an efficient manner. A device shall support the RTP/UDP protocol and the device should support RTP/UDP multicasting.

12.1.1.2 RTP/TCP

If there is a packet loss during media transfer via UDP, then the standard allows for RTP data transfer via TCP as an alternative means of media transport. A device MAY support the RTP/TCP based option. If the device supports the RTP/TCP protocol, then this protocol shall conform to [RFC 4571] (Framing Real-time Transport Protocol and RTP Control Protocol [RTCP] Packets over Connection-Oriented Transport).

12.1.1.3 RTP/RTSP/TCP

The device should support media transfer using RTP/RTSP to traverse a firewall using an RTSP tunnel. This protocol shall conform to [RFC 2326], 10.12.

12.1.1.4 RTP/RTSP/HTTP/TCP

The data stream shall be sent via HTTP to traverse a firewall. A device shall support media transfer using RTP/RTSP/HTTP/TCP. And if a device supports TLS1.0, the data stream shall be sent or received via HTTPS to traverse a firewall, and a device shall support media transfer using RTP/RTSP/HTTPS/TCP.

This protocol shall conform to [RFC 2326] (RTSP, 12.2.1.1: Embedded [Interleaved] Binary Data).

This tunnelling method shall also conform to QuickTime available from Apple Inc. The mandatory parts of the following document shall be implemented by an NVT.

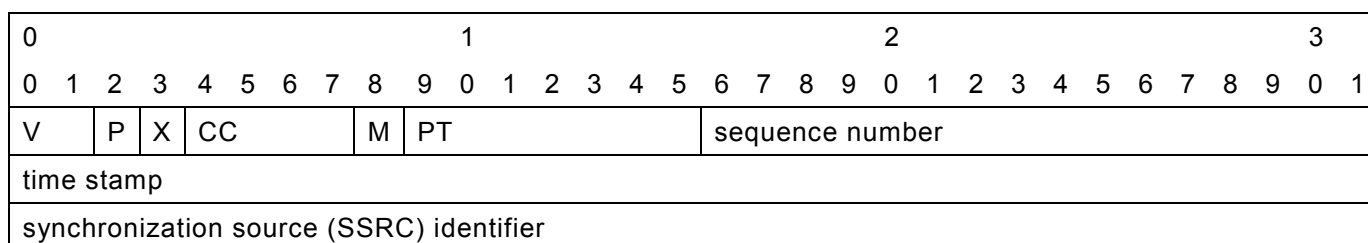
<http://developer.apple.com/quicktime/icefloe/dispatch028.html>

12.1.2 Media transport

12.1.2.1 RTP

The Real-time Transport Protocol provides real-time transfer for media streams between two end points. The RTP protocol provides support for re-ordering, de-jittering and media synchronization. It has an RTP header layout as shown in Figure 14.

All media streams transferred by the RTP protocol shall conform to [RFC 3550], [RFC 3551], [RFC 3984], [RFC 3016] and JPEG over RTP (see 12.1.3).

**Figure 14 – RTP header**

IEC 2754/13

An RTP header shall be filled up with the values shown in Table 190.

Table 190 – RTP header value

Header field	Value	Description
Version (V): 2 bits	2	
Padding (P): 1 bit	0/1	If the payload includes padding octet, this should be set to "1"
Extension (X): 1 bit	0/1	Depends on the use of extension of RTP header. The specification defines two scenarios where a RTP header extension could be used to transmit additional information: 1) "JPEG over RTP" (see 12.1.3). 2) Replay (see Clause 21) If the header extension is used the Extension bit shall be set.
CSRC count (CC): 4 bits	0	
Marker (M): 1 bit	0/1	The usage shall be conform to related RFCs (e.g. [RFC 3984] for H.264 Video) or to this standard e.g. "JPEG over RTP" (see 12.1.3) or RTP streaming of metadata (see 12.1.2.1).
Payload type (PT): 7 bits	See [RFC 3551] Clause 6.	
Sequence Number: 16 bits		The initial value of the "sequence number" should be random (unpredictable) to make known-plaintext attacks on encryption more difficult. This number increments by one for each RTP data packet sent
timestamp: 32 bits		The initial value of the "timestamp" should be random (unpredictable) to make known-plaintext attacks on encryption more difficult. See 12.1.2.2 for further details of Media Synchronization. The usage of the timestamp is dependent on the codec.
SSRC 32 bits		The synchronization source for the data stream. This specification makes no restrictions on the use of this field.

RTP for metadata stream

Metadata streams are also transported by RTP. The usage of payload type, marker and timestamp for RTP header for the metadata stream is defined in the following way:

- A dynamic payload type (96-127) shall be used for payload type which is assigned in the process of a RTSP session setup.
- The RTP marker bit shall be set to “1” when the XML document is closed.
- It is RECOMMENDED to use an RTP timestamp representing the creation time of the RTP packet with a RTP clock rate of 90 000 Hz. Only UTC timestamps shall be used within the metadata stream. The synchronization of video and audio data streams is done using RTCP.

The Metadata payload is an XML document with root node `tt:MetaDataStream`. There is no limitation on the size of the XML document. When a synchronization point (see 11.18.2) is requested for the stream, the previous XML document shall be closed and a new one started. It is RECOMMENDED to start new XML documents after 1 second, at the longest. The RTP timestamp of the Metadata stream has no specific meaning. The Metadata stream multiplexes Metadata from different sources. This specification defines placeholders for the Scene Description of the Video Analytics, the PTZ Status of the PTZ controller and the Notifications of the Event Configuration. A device can select which of these parts should be multiplexed into the Metadata during the Media Configuration (see clause 11.2.10). Each part can appear multiple times in arbitrary order within the document. A Metadata connection can be bi-directional using the backchannel mechanism (see 12.3).

Metadata stream contains the following elements:

- VideoAnalyticsStream;
- PTZStream;
- EventStream.

The place-holders for the different metadata sources have the following XMLstructure:

```
<xs:complexType name="VideoAnalyticsStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="Frame" type="tt:Frame"/>
    ...
  </xs:choice>
</xs:complexType>

<xs:complexType name="PTZStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="PTZStatus"/>
    ...
  </xs:choice>
</xs:complexType>

<xs:complexType name="EventStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="wsnt:NotificationMessage"/>
    ...
  </xs:choice>
</xs:complexType>
```

The following is an example of a metadata XML document:

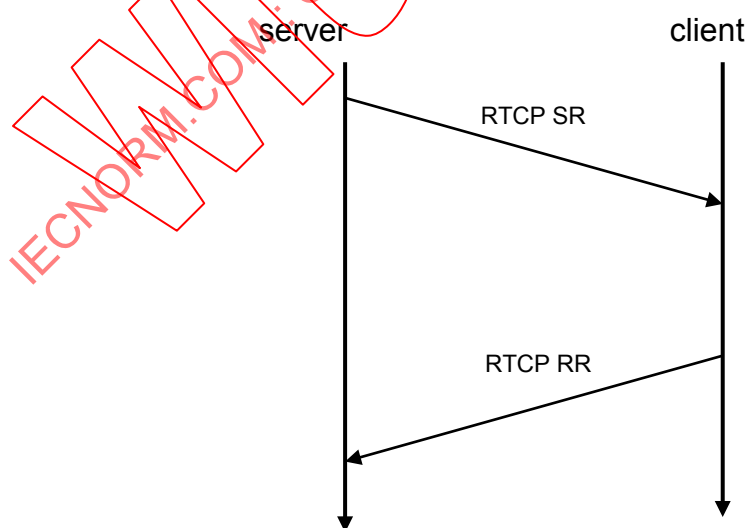
```
<?xml version="1.0" encoding="UTF-8"?>
<tt:MetaDataStream xmlns:tt="http://www.onvif.org/ver10/schema">
  <tt:VideoAnalytics>
    <tt:Frame UtcTime="2008-10-10T12:24:57.321">
      ...
    </tt:Frame>
    <tt:Frame UtcTime="2008-10-10T12:24:57.621">
      ...
    </tt:Frame>
  </tt:VideoAnalytics>
</tt:MetaDataStream>

<?xml version="1.0" encoding="UTF-8"?>
<tt:MetaDataStream xmlns:tt="http://www.onvif.org/ver10/schema">
  <tt:Event>
    <wsnt:NotificationMessage>
      <wsnt:Message>
        <tt:Message UtcTime= "2008-10-10T12:24:57.628">
          ...
        </tt:Message>
      </wsnt:Message>
    </wsnt:NotificationMessage>
  </tt:Event>
</tt:MetaDataStream>
```

12.1.2.2 RTCP

The RTP Control Protocol provides feedback on quality of service being provided by RTP and synchronization of different media streams. The RTCP protocol shall conform to [RFC 3550]. The protocol uses sender reports (SR) and receiver reports (RR) for conveying the information between server and client as shown in Figure 15

For a feedback request, [RFC 4585] and [RFC 5104] should be supported.



IEC 2755/13

Figure 15 – RTCP sequence

Media synchronization

A client MAY receive audio and video streams simultaneously from more than one device. In this case, each stream uses a different clock (from data acquisition to packet receiving). RTCP Sender Reports (SR) are used to synchronize different media streams. RTCP SRs shall conform to [RFC 3550].

The RTCP Sender Report (SR) packet has fields for the RTP timestamp and for a wall clock timestamp (absolute date and time, 64 bit NTP [Network Time Protocol]). See Figure 16.

A device shall support RTCP Sender Report for media synchronization. The client should use RTCP for the media synchronization.

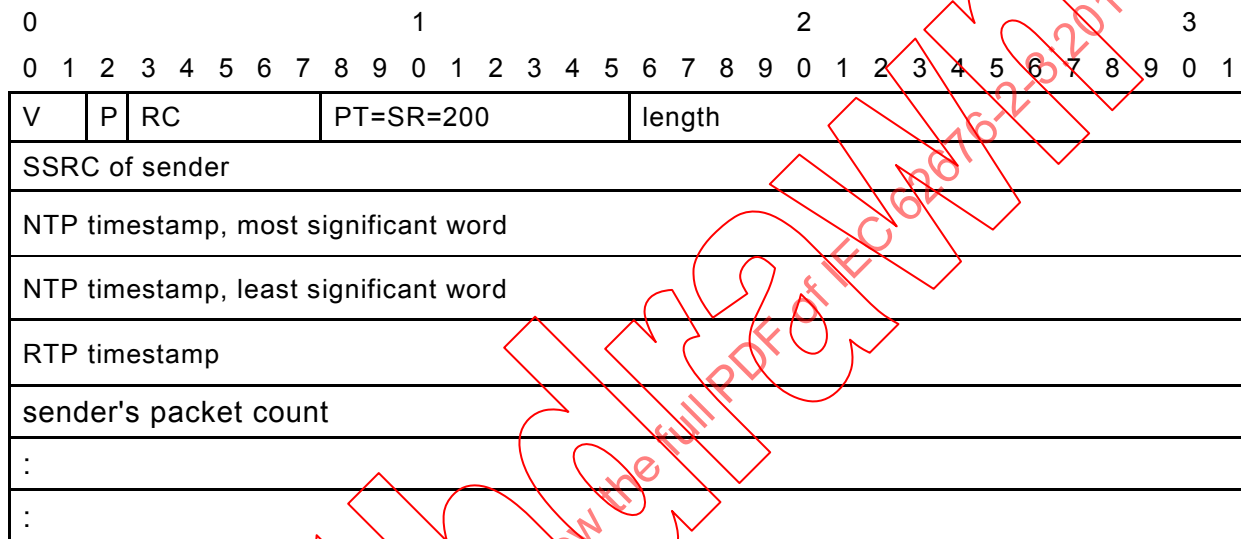


Figure 16 – RTCP Sender Report

IEC 2756/13

The wall clock should be common in the device and each timestamp value should be determined properly. The client can synchronize different media streams at the appropriate timing based on the RTP clock and wall clock timestamps (see Figure 17).

In case of multiple devices, the NTP timestamp should be common to all devices, and the NTP server should be required in the system ³.

³ The client can get information about "NTP server availability" from the devices by using the GetNTP command. Refer to 8.2.5.

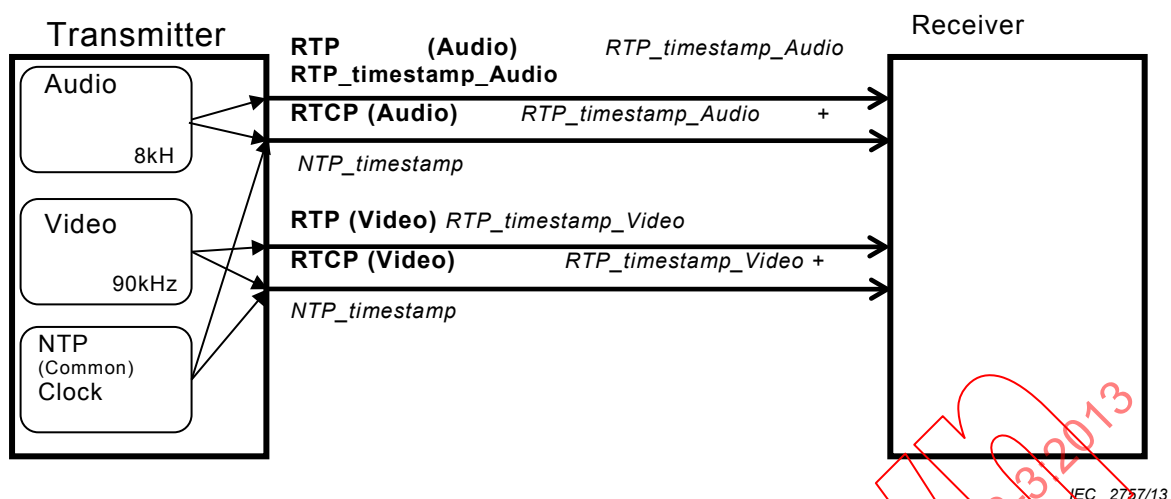


Figure 17 – Media synchronization

12.1.3 Synchronization point

Synchronization points allow clients to decode and correctly use data after the synchronization point. A synchronization point MAY be requested by a client in case of decoder error (e.g. in consequence of packet loss) to enforce the device to add an I-Frame as soon as possible or to request the current PTZ or event status.

In addition to the WebService based methods (see 11.18.2 and 15.6) a device shall support, this standard RECOMMENDS the PLI messages as described in [RFC 4585] to request an SynchronizationPoint.

12.1.4 JPEG over RTP

12.1.4.1 Overall packet structure

The syntax for transmitting JPEG streams follows [RFC 2435]. The syntax does allow embedding additional data, beyond the limits of [RFC 2435], by using an optional RTP header extension, as specified below, with some of the RTP packets. This option, however, changes the exact semantics for frames which include such packets.

The overall format of the JPEG RTP packet is shown in Figure 18.

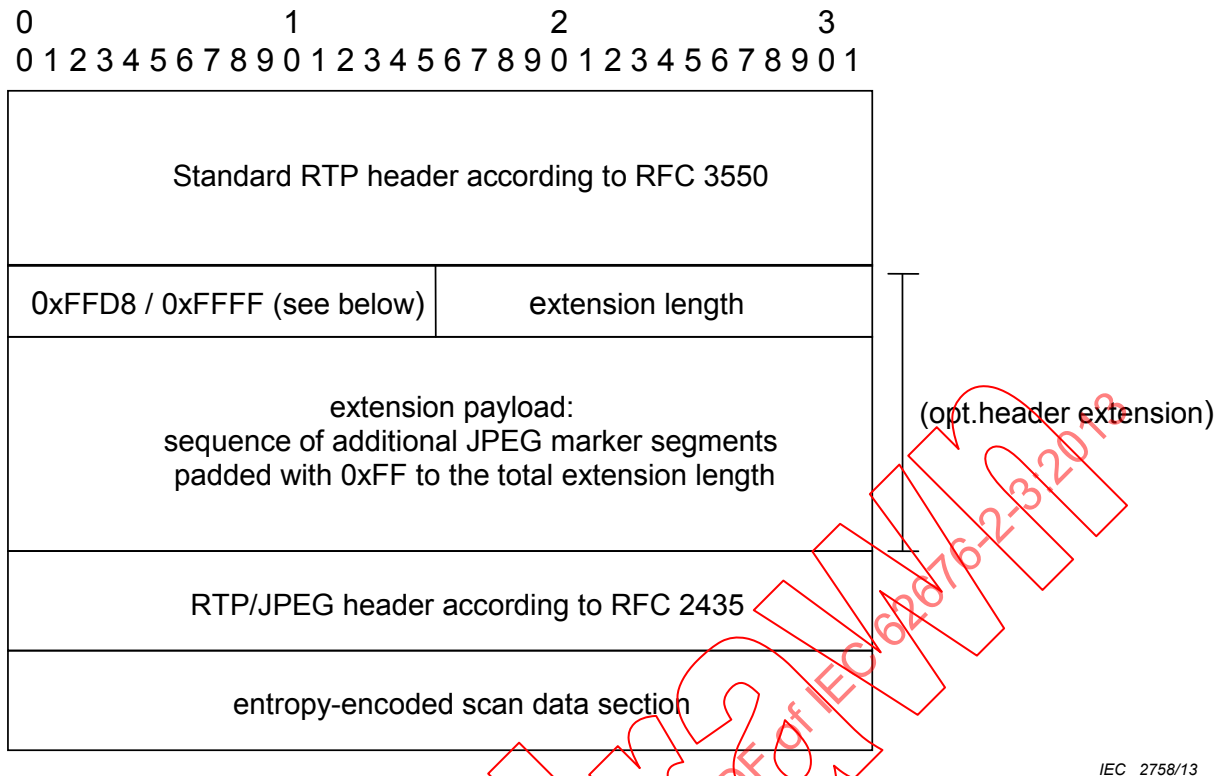


Figure 18 – RTP/JPEG packet structure (only the typical content is listed for the extension payload)

In order to distinguish an optional RTP header extension from possible other header extensions, the first 16 bits (the first two octets of the four-octet extension header) of an RTP shall have the value 0xFFD8 (JPEG SOI marker) for the initial packet and 0xFFFF for other RTP packets within a frame.

As required by [RFC 3550], the presence of the optional header extension shall be signalled via the X-bit of the RTP header. The extension length field within the header extension counts the number of 32-bit items following as extension payloads. For example, a zero-length field following the 32-bit extension header represents an empty header extension).

The entropy-encoded scan data section may not be present in all RTP packets. A complete RTP/JPEG header however shall be present in the initial packet of every frame and all packets containing an entropy-encoded scan data section, otherwise it may be missing.

The fragment offset field within the RTP/JPEG header, according to [RFC 2435], should be used as if no header extension would be present. Additionally, if a packet does not contain an entropy-encoded scan data segment, but contains a header extension the fragment offset field shall not be zero if any packets containing an entropy-encoded scan data section for the same frame have been transmitted. If the initial packet of a frame contains no header extension, according to this standard, its fragment offset field shall be zero, otherwise it should be zero. All packets including an RTP/JPEG header with a fragment offset of zero and a Q value between 128-255 shall include a quantization table header according to 3.1.8 of [RFC 2435], other packets shall NOT include this header.

12.1.4.2 Logical decoding specification

For the decoding specification, it is assumed that the original packet order within the RTP stream has been restored according to the RTP sequence numbering.

If the initial packet of a frame contains no RTP header extension as specified above, decoders shall generate the complete scan header and perform the decoding as specified by [RFC 2435]. The scan data sections and payloads of any header extension conforming to this specification, up to and including the next RTP packet with its marker bit set, shall be concatenated as they occur within the stream ignoring their fragment offset values.

Otherwise (at least an empty header extension as specified above is present in the initial packet of a frame), the following rules apply for each such frame:

- If the initial packet of a frame does not contain an entropy-encoded scan data segment, but contains a header extension as specified above, then decoders shall concatenate its header extension payload with (possibly empty or not existing) header extension payload(s) conforming to this specification of the subsequent packets up to and including the first packet with the RTP marker bit set or containing an entropy-encoded scan data segment.
- The concatenated initial RTP header extension payload (sequence) shall be logically prepended with a JPEG SOI marker (0xFFD8).
- If the Q-value of the RTP/JPEG scan header within the initial packet of a frame is not zero, the quantization tables shall be pre-initialized according to the rules of [RFC 2435]. If Q is equal to zero the quantization tables shall be copied from the previous frame, allowing for DQT markers within this initial header extension payload (sequence) to override them.
- If this frame is the initial frame of a sequence, the Huffman tables shall be pre-initialized according to [RFC 2435]. The Huffman tables for all subsequent frames shall be copied from the previous frame, allowing the frames to be overridden by DHT markers within the initial header extension payload (sequence).
- If the initial RTP header extension payload (sequence) supplies no DRI marker, but the RTP/JPEG header of the initial packet of a frame contains an RTP/JPEG restart marker, a DRI marker corresponding to the rules of [RFC 2435] shall be appended to the initial header extension payload (sequence). Otherwise, if the initial RTP header extension (sequence) supplies a DRI marker, the marker shall take precedence over any other RTP/JPEG restart marker according to [RFC 2435] for the same frame. However, for compatibility with decoders conforming to [RFC 2435] only, encoders normally should use an RTP/JPEG restart marker with consistent values, if restart intervals are to be used.
- DRI markers shall NOT be derived from previous frames.
- If the initial RTP header extension payload (sequence) supplies no SOF marker, which otherwise takes precedence, a SOF marker shall be appended to it with the following values:
 - If both the width and height field of the RTP/JPEG header are zero, the SOF marker of the previous frame shall be used.
 - Otherwise it shall be derived according to the rules of [RFC 2435].

However, as long as the (rounded up) image size fits within the range as specified in [RFC 2435], encoders should specify the image size within the RTP/JPEG header consistent with the values of an additional SOF header.

- If the initial header extension payload (sequence) supplies no SOS marker, a corresponding marker shall be derived according to [RFC 2435] and appended to it, otherwise the SOS marker in the extension takes precedence.

An SOS marker shall NOT be derived from previous frames.

If the SOS marker is present and not followed by entropy-encoded scan data within the extension, the marker shall be the final marker within the initial extension payload (sequence) of a frame. Necessary padding with 0xFF-octets shall NOT follow this marker but MAY precede it.

- The remaining entropy-encoded scan data and header extensions payloads shall be logically appended in the same order as they occur within the RTP stream up to the

end of the frame as indicated by the RTP marker bit. A final EOI marker shall also be added if it is not yet present within the logical sequence for this frame.

For each frame, the resulting sequence up to and including the first (possibly added) EOI marker shall be a valid (possibly abbreviated) JPEG stream, resulting in one complete image from the decoding process for this frame. The meaning of any data after this first EOI marker for each frame is outside the scope of this standard.

12.1.4.3 Supported colour spaces and sampling factors

A transmitter should use only greyscale and YCbCr colour space. A Client shall support both greyscale and YCbCr.

The sampling factors for YCbCr shall correspond to the values supported by [RFC 2435]. For example, a sampling factor of 4:2:0 (preferred) or 4:2:2.

12.1.4.4 Pixel aspect ratio handling

The pixel aspect ratio of JPEG files can be specified within the JFIF marker. If the pixel aspect ratio is different from the standard 1:1 and 1:2 ratio according to [RFC 2435], this marker should be transmitted in the initial header extension payload (sequence) of every frame to specify the (for interlaced material field-based) pixel aspect ratio.

12.1.4.5 Interlaced handling

Interlaced video is encoded as two independent fields and signalled as specified by [RFC 2435] within the RTP/JPEG header.

Both fields shall use the same colour space, sampling factors and pixel aspect ratio.

Interlaced encoding should NOT be used if the frame was originally scanned progressively.

12.2 Media control protocol

12.2.1 Stream control

The media stream is controlled using the protocol defined in the URI. The general control flow is depicted in Figure 19. The URI is returned in response to the GetStreamUri command defined in 11.15.2.

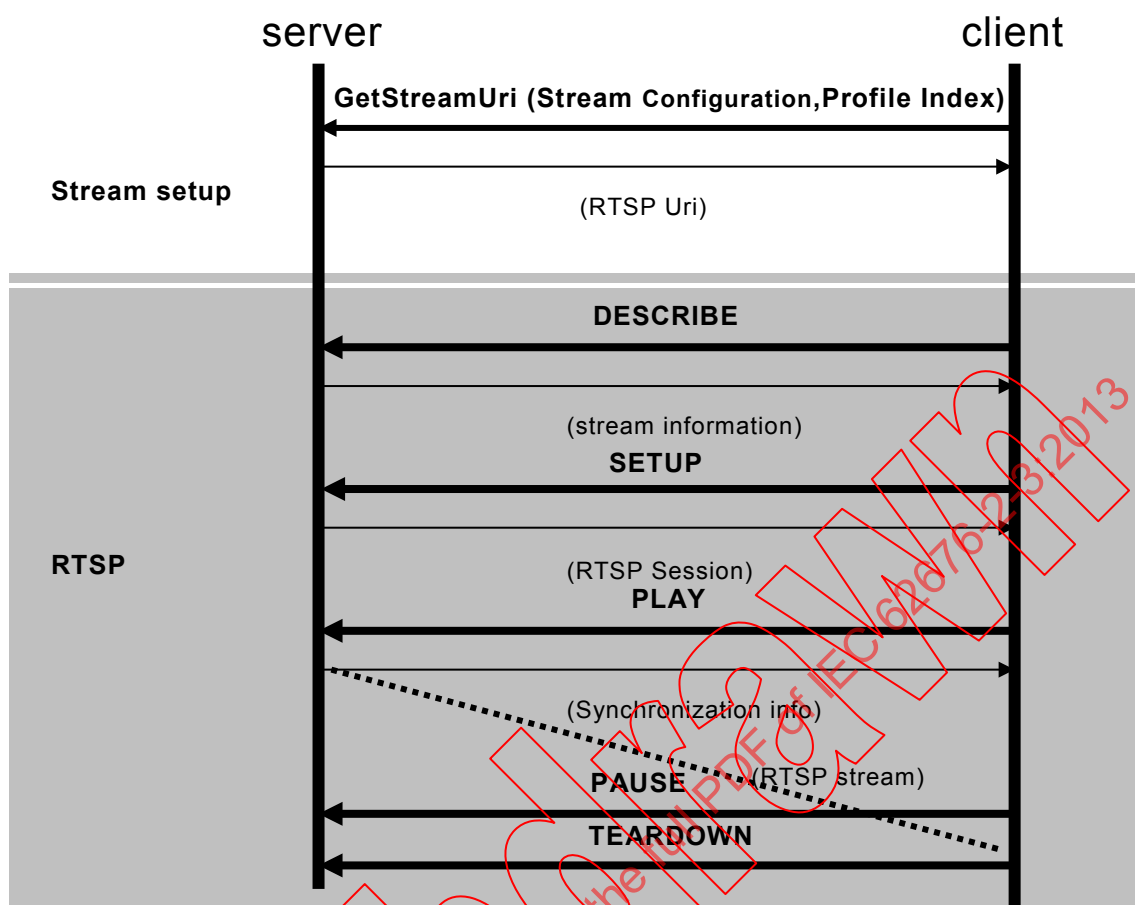


Figure 19 – Stream control

IEC 2759/13

12.2.1.1 RTSP

All devices and clients shall support RTSP ([RFC 2326]) for session initiation and playback control (see Table 191). RTSP shall use TCP as its transport protocol, the default TCP port for RTSP traffic is 554. The Session Description Protocol (SDP) shall be used to provide media stream information and SDP shall conform to [RFC 4566].

Table 191 – RTSP methods

Method	Direction	SPEC	Description
OPTIONS	R->T T->R	M X	Required to get optional method capability and to allow different versions in the future.
DESCRIBE	R->T	M	Required to retrieve media parameters within the designated profile.
ANNOUNCE	R->T T->R	X	
SETUP	R->T	M	Required to set media session parameters.
PLAY	R->T	M	Required to start media stream.
PAUSE	R->T	O	Required to temporarily stop media stream. Handling multiple streams in a narrow bandwidth network, by suspending RTP stream, the traffic can be well controlled by reducing redundant data and congested network traffic can be avoided.
TEARDOWN	R->T	M	Required to release a media session.
GET_PARAMETER	R->T T->R	O	
SET_PARAMETER	R->T T->R	O O	An optional method to keep an RTSP session alive (R->T direction only).
REDIRECT	T->R	X	
RECORD	R->T	X	
Keys: X: Not supported, M: Mandatory, O: Optional			

12.2.1.1.1 Keep-alive method for RTSP session

A RTSP client keeps the RTSP Session alive and prevents it from session timeout (see [RFC 2326], 12.37). This specification recommends the following methods to keep RTSP alive for both Unicast and Multicast streaming.

- 1) The client can optionally set the Timeout parameter (in seconds) using the `Set<configurationEntity>EncoderConfiguration` command defined in 11.2, otherwise a default value of "60" is used.
- 2) In all RTSP SETUP responses, a transmitter should include the Timeout value according to [RFC 2326], 12.37 and the transmitter should use the Timeout value for keep-alive.
- 3) To keep the RTSP Session alive, a client shall call the RTSP server using any RTSP method or send RTCP receiver reports. `SET_PARAMETER` is the RECOMMENDED RTSP method to use.

See Figure 20 for a typical example of the control flow.

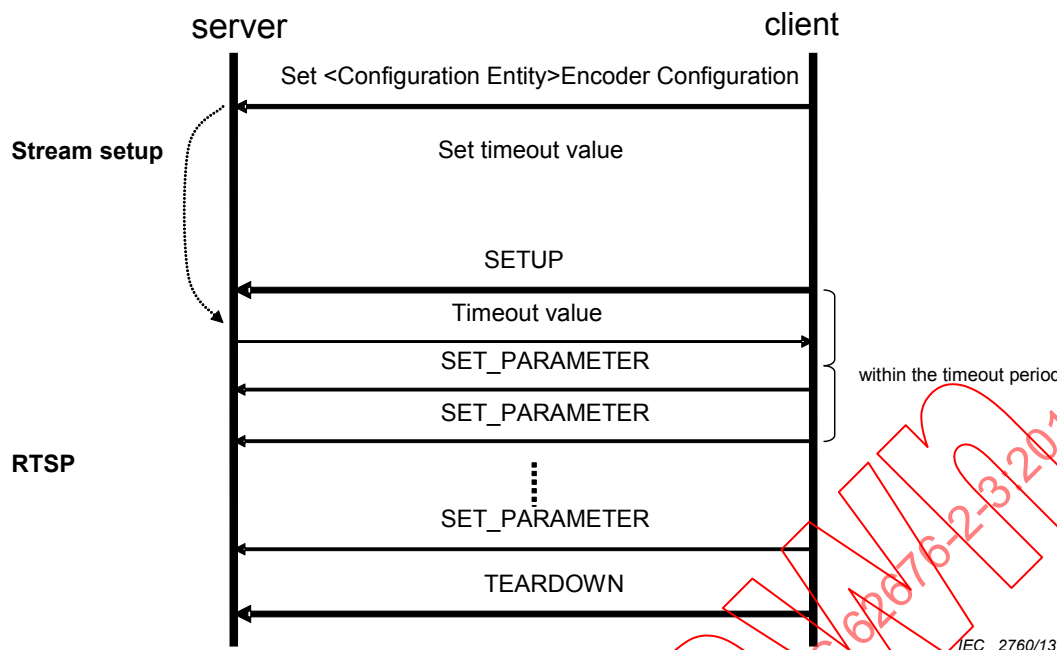


Figure 20 – Keep Alive

12.2.1.1.2 RTSP Audio and Video Synchronization

In order that clients may immediately begin synchronizing video and audio streams, and computing absolute UTC timestamps for incoming packets for recording purposes, a transmitter should include the following header fields in the RTSP PLAY response:

- Range ([RFC 2326], 12.29). This shall include a start time in clock units ([RFC 2326], 3.7), *not* SMPTE or NPT units.
- RTP-Info ([RFC 2326], 12.33). This shall include an *rtptime* value which corresponds to the start time specified in the Range header.

Example:

```

client->>server: PLAY rtsp://example.com/onvif_camera/video RTSP/1.0
                  CSeq: 4
                  Range: npt=now-
                  Session: 12345678

server->>client:  RTSP/1.0 200 OK
                  CSeq: 4
                  Session: 12345678
                  Range: 20100217T143720.257Z-
                  RTP-Info: url=rtsp://example.com/onvif_camera/video;
seq=1234;rtptime=3450012
  
```

12.2.1.1.3 RTSP session for a Metadata stream

In the case of a metadata stream, the SDP description “application” should be used in the DESCRIBE response for media type and “vnd.onvif.metadata” should be used for encoding a name.

Example RTSP DESCRIBE message exchange between an RTSP Server (server) and an RTSP client (client):

```

client->server:    DESCRIBE rtsp://example.com/onvif_camera RTSP/1.0
                  CSeq: 1

server->client:    RTSP/1.0 200 OK
                  CSeq: 1
                  Content-Type: application/sdp
                  Content-Length: XXX
                  v=0
                  o=- 2890844256 2890842807 IN IP4 172.16.2.93
                  s=RTSP Session
                  m=audio 0 RTP/AVP 0
                  a=control:rtsp://example.com/onvif_camera /audio
                  m=video 0 RTP/AVP 26
                  a=control:rtsp://example.com/onvif_camera /video
                  m=application 0 RTP/AVP 107
                  a=control:rtsp://example.com/onvif_camera/metadata
                  a=recvonly
                  a=rtpmap
                  a=rtpmap:107 vnd.onvif.metadata/90000

```

12.2.1.1.4 RTSP message example

This example shows the message transfer between an RTSP client (client) and an RTSP server (server). The client requests one audio and one video stream from the device. The Stream Uri “rtsp://example.com/onvif_camera” can be retrieved using the GetStreamUri command. Refer to 11.15.2.

```

client->server:    DESCRIBE rtsp://example.com/onvif_camera
RTSP/1.0
                  CSeq: 1

server->client:    RTSP/1.0 200 OK
                  CSeq: 1
                  Content-Type: application/sdp
                  Content-Length: XXX
                  v=0
                  o=- 2890844256 2890842807 IN IP4 172.16.2.93
                  s=RTSP Session
                  m=audio 0 RTP/AVP 0
                  a=control:rtsp://example.com/onvif_camera/audio
                  m=video 0 RTP/AVP 26
                  a=control:rtsp://example.com/onvif_camera/video

client->server:    SETUP rtsp://example.com/onvif_camera/audio RTSP/1.0
                  CSeq: 2
                  Transport: RTP/AVP;unicast;client_port=8002-
8003

server->client:    RTSP/1.0 200 OK
                  CSeq: 2
                  Transport: RTP/AVP;unicast;client_port=8002-
8003;
                  server_port=9004-9005
                  Session: 12345678; timeout=60

client->server:    SETUP rtsp://example.com/onvif_camera/video RTSP/1.0
                  CSeq: 3

```

```

8005
Transport: RTP/AVP;unicast;client_port=8004-
Session: 12345678

server->client: RTSP/1.0 200 OK
                  CSeq: 3
                  Transport: RTP/AVP;unicast;client_port=8004-
8005;
                  server_port=9006-9007
                  Session: 12345678; timeout=60

client->server: PLAY rtsp://example.com/onvif_camera RTSP/1.0
                  CSeq: 4
                  Range: npt=now-
                  Session: 12345678

server->client: RTSP/1.0 200 OK
                  CSeq: 4
                  Session: 12345678
RTP-Info:          url=rtsp://example.com/onvif_camera/video;
seq=1234;rtptime=3450012, url=rtsp://example.com/onvif_camera/audio;
                  seq=22434;rtptime=1234566

client->server: TEARDOWN rtsp://example.com/onvif_camera RTSP/1.0
                  CSeq: 5
                  Session: 12345678

server->client: RTSP/1.0 200 OK
                  CSeq: 5
                  Session: 12345678

```

12.2.1.2 RTSP over HTTP

The RTSP over HTTP/HTTPS shall be supported in order to traverse a firewall. See 12.1.1.4 RTP/RTSP/HTTP/TCR.

12.3 Back channel connection

This subclause describes how a bidirectional connection can be established between a client and a server. The backchannel connection handling is done using RTSP [RFC 2326]. Therefore a mechanism is introduced which indicates that a client wants to built up a backchannel connection. RTSP provides feature-tags to deal with such functionality additions.

A device that supports bi-directional connections (e.g audio or metadata connections) shall support the introduced RTSP extensions.

12.3.1 RTSP Require – Tag

The RTSP standard [RFC 2326] can be extended by using additional headers objects. For that purpose a Require tag is introduced to handle special functionality additions (see [RFC 2326], 1.5 Extending Rtp and 12.32 Require).

The Require-tag is used to determine the support of this feature. This header shall be included in any request where the server is required to understand that feature to correctly perform the request.

A device that supports backchannel shall understand the backchannel tag:

- www.onvif.org/ver20/backchannel

An RTSP client that wants to built up an RTSP connection with a data backchannel shall include the Require header in its requests.

12.3.2 Connection setup for a bi- directional connection

A client shall include the feature tag in its DESCRIBE request to indicate that a bidirectional data connection shall be established.

A server that understands this Require tag shall include an additional media stream in its SDP file as configured in its Media Profile.

An RTSP server that does not understand the backchannel feature tag or does not support bidirectional data connections shall respond with an error code *551 Option not supported* according to the RTSP standard. The client can then try to establish an RTSP connection without backchannel.

A SDP file is used to describe the session. The server shall include the *a=sendonly* or the *a=recvonly* attributes in each media section of the SDP file to indicate the direction the media data will be send.

The server shall list all supported decoding codecs as own media section and the client chooses which one is used.

12.3.2.1 Example 1: Server without backchannel support

```
Client - Server:      DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                      CSeq: 1
                      User-Agent: ONVIF Rtsp client
                      Accept: application/sdp
                      Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 551 Option not supported
                      CSeq: 1
                      Unsupported: www.onvif.org/ver20/backchannel
```

12.3.2.2 Example 2: Server with Onvif backchannel support

```
Client - Server:      DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                      CSeq: 1
                      User-Agent: ONVIF Rtsp client
                      Accept: application/sdp
                      Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 200 OK
                      CSeq: 1
                      Content-Type: application/sdp
                      Content-Length: xxx

                      v=0
o= 2890842807 IN IP4 192.168.0.1
s=RTSP Session with audiobackchannel
m=video 0 RTP/AVP 26
a=control:rtsp://192.168.0.1/video
                      a=recvonly
                      m=audio 0 RTP/AVP 0
a=control:rtsp://192.168.0.1/audio
a=recvonly
m=audio 0 RTP/AVP 0
a=control:rtsp://192.168.0.1/audioback
a=rtpmap:0 PCMU/8000
a=sendonly
```

This SDP file completely describes the RTSP session. The Server gives the client its control URLs to setup the streams.

In the next step the client can setup the sessions:

```
Client - Server:      SETUP rtsp://192.168.0.1/video RTSP/1.0
                      CSeq: 2
                      Transport: RTP/AVP;unicast;client_port=4588-
4589

Server - Client:      RTSP/1.0 200 OK
                      CSeq: 2
Session: 123124;timeout=60
Transport:RTP/AVP;unicast;client_port=4588-4589; server_port=6256-6257

Client - Server:      SETUP rtsp://192.168.0.1/audio RTSP/1.0
                      CSeq: 3
Session: 123124
Transport: RTP/AVP;unicast;client_port=4578-4579

Server - Client:      RTSP/1.0 200 OK
                      CSeq: 3
Session: 123124;timeout=60
Transport:RTP/AVP;unicast;client_port=4578-4579; server_port=6276-6277

Client - Server:      SETUP rtsp://192.168.0.1/audioback RTSP/1.0
                      CSeq: 4
Session: 123124
Transport: RTP/AVP;unicast;client_port=6296-6297
Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 200 OK
                      CSeq: 4
Session: 123124;timeout=60
Transport:RTP/AVP;unicast;client_port=6296-6297; server_port=2346-2347
```

The third setup request establishes the audio backchannel connection.

In the next step the client starts the session by sending a PLAY request.

```
Client - Server:      PLAY rtsp://192.168.0.1 RTSP/1.0
                      CSeq: 5
                      Session: 123124
Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 200 OK
                      CSeq: 5
                      Session: 123124;timeout=60
```

After receiving the OK response to the PLAY request the client MAY start sending audio data to the server. It shall not start sending data to the server before it has received the response.

The Require-header indicates that a special interpretation of the PLAY command is necessary. The command covers both starting of the video and audio stream from NVT to the client and starting the audio connection from client to server.

To terminate the session the client sends a TEARDOWN request.

```
Client - NVT:         TEARDOWN rtsp://192.168.0.1 RTSP/1.0
                      CSeq: 6
                      Session: 123124
Require: www.onvif.org/ver20/backchannel
```



```
NVT - Client:   RTSP/1.0 200 OK
                CSeq: 6
                Session: 123124
```

12.3.3 Multicast streaming

If the client intends to send its data in multicast it uses the transport parameter in the SETUP request to tell the server the multicast address and port.

Example: Multicast setup

```
Client - Server:      SETUP rtsp://192.168.0.1/audioback RTSP/1.0
                      CSeq: 4
Session: 123124
Transport: RTP/AVP;multicast;destination=224.2.1.1;port=60000-60001;ttl=128
Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 200 OK
                      CSeq: 4
Session: 123124;timeout=60
Transport: RTP/AVP;multicast;destination=224.2.1.1;port=60000-
60001;ttl=128;mode="PLAY"
```

12.4 Error handling

RTSP and HTTP protocol errors are classified into different categories (for example, status codes 1xx, 2xx, 3xx, 4xx and 5xx respectively). The device and the client shall support and handle these status codes. For RTSP status code definitions refer to [RFC 2326], 11.0. For HTTP status code definitions refer HTTP/1.1 [RFC 2616], 10.0

13 Receiver configuration

This service offers commands to manage Receiver objects, which are used to receive media streams from other devices. A Receiver object contains the information how to setup the stream, the mode of the receiver and the Stream Uri (MediaUri). A device shall at least support Media Uris of 128 octet length. The Receiver - MaximumRTSPURILength capability indicates the maximum length supported by the device. The Receiver Service shall be implemented by devices that can receive media streams.

The IP or DNS address in the transmit URI given to the receiver, is the address that the device hosting the receiver service will use to access the transmit device. If, for example, the client has to communicate through a NAT router to access the transmitter and the receiver, the transmitter address that the client gives the receiver (in this case a local network address) may not be the same address that the client would use to access the transmitter (in this case an external network address).

A device shall support RTP transfer via RTP, see 12.1.1.1, and RTP transfer via RTSP/HTTP/TCP, see 12.1.1.4. A device may support other RTP transport protocols and shall indicate what it supports with the appropriate capability, see Receiver category in Table 11.

13.1 Persistence

All the objects created within the receiver service shall be persistent – i.e. they shall survive a power cycle. Likewise, all the configuration data in the objects shall be persistent.

13.2 Receiver modes

A receiver can operate in three distinct modes:

- Always Connect – the receiver attempts to maintain a persistent connection to the configured endpoint;
- Never Connect – the receiver does not attempt to connect;
- Auto Connect – the receiver connects on demand, as required by consumers of the media streams.

13.3 Receiver commands

This subclause describes the commands offered by the Receiver Service.

13.3.1 Get receivers

This operation lists all receivers that currently exist on the device. The receiver service shall support this command (see Table 192).

Table 192 – GetReceivers command

GetReceivers	
Message name	Description
GetReceiversRequest	<i>This message is empty.</i>
GetReceiversResponse	Contains a list of receivers tt:Receiver Receivers [0][unbounded]
Fault codes	Description
<i>No specific fault codes.</i>	

13.3.2 Get receiver

This operation retrieves the details of a specific receiver whose token is known to the client. The Receiver Service shall support this command (see Table 193).

Table 193 – GetReceiver command

GetReceiver	
Message name	Description
GetReceiverRequest	<i>Contains the token of the requested receiver.</i> tt:ReferenceToken ReceiverToken [1][1]
GetReceiverResponse	<i>Contains the details of the requested receiver.</i> tt:Receiver Receiver [1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	<i>The receiver indicated by ReceiverToken does not exist.</i>

13.3.3 Create receiver

This operation creates a new receiver. The Receiver Service shall support this command (see Table 194).

Table 194 – CreateReceiver command

CreateReceiver	
Message name	Description
CreateReceiverRequest	Contains the initial configuration of the receiver. tt:ReceiverConfiguration Configuration [1][1]
CreateReceiverResponse	Contains the details of the receiver that was created. tt:Receiver Receiver [1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	The specified configuration is invalid.
env:Receiver ter:Action ter:MaxReceivers	The maximum supported number of receivers has been reached.

13.3.4 Delete receiver

This operation deletes an existing receiver. A receiver MAY NOT be deleted if it is in use. The Receiver Service shall support this command (see Table 195).

Table 195 – DeleteReceiver command

DeleteReceiver	
Message name	Description
DeleteReceiverRequest	Contains the token of the receiver to be deleted. tt:ReferenceToken ReceiverToken [1][1]
DeleteReceiverResponse	This message is empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	The receiver indicated by ReceiverToken does not exist.
env: Receiver ter:Action ter:CannotDeleteReceiver	It is not possible to delete the specified receiver, for example because it is currently in use.

13.3.5 Configure receiver

This operation configures a receiver. The Receiver Service shall support this command (see Table 196).

Table 196 – ConfigureReceiver command

ConfigureReceiver	
Message name	Description
ConfigureReceiverRequest	Contains the token of the requested receiver and the new configuration. tt:ReferenceToken ReceiverToken [1][1] tt:ReceiverConfiguration Configuration [1][1]
ConfigureReceiverResponse	This message is empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	The receiver indicated by ReceiverToken does not exist.
env:Sender ter:InvalidArgVal ter:BadConfiguration	The specified configuration is invalid.

13.3.6 SetReceiverMode

This operation may be used to set the mode of the receiver independently of the rest of its configuration. The Receiver Service shall support this command (see Table 197).

Table 197 – SetReceiverMode command

SetReceiverMode	
Message name	Description
SetReceiverModeRequest	Contains the token of the requested receiver and the new mode. tt:ReferenceToken ReceiverToken [1][1] tt:ReceiverMode ReceiverMode [1][1]
SetReceiverModeResponse	This message is empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	The receiver indicated by ReceiverToken does not exist.

13.3.7 GetReceiverState

This operation determines whether the receiver is currently disconnected, connected or attempting to connect. The Receiver Service shall support this command (see Table 198).

Table 198 – GetReceiverState command

GetReceiverState	
Message name	Description
GetReceiverStateRequest	Contains the token of the requested receiver. tt:ReferenceToken ReceiverToken [1][1]
GetReceiverStateResponse	Contains the current state of the receiver. tt:ReceiverState State [1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	The receiver indicated by ReceiverToken does not exist.

13.4 Events

The receiver service shall dispatch events through the event service. It shall be capable of generating the events listed in this subclause whenever the condition that fires the event occurs.

13.4.1 ChangeState

Whenever a receiver changes state, the device shall dispatch the following event:

```
Topic: tns1: Receiver/ChangeState
<tt:MessageDescription IsProperty="false">
  <tt:Source>
```

```

    <tt:SimpleItemDescription Name="ReceiverToken" Type="tt:ReceiverToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="NewState" Type="tt:ReceiverState"/>
    <tt:SimpleItemDescription Name="MediaUri" Type="tt:MediaUri" minOccurs="0"/>
  </tt:Data>
</tt:MessageDescription>

```

13.4.2 Connection Failed

If a receiver fails to establish a connection, the device shall dispatch the following event:

```

Topic: tns1: Receiver/ConnectionFailed
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="ReceiverToken" Type="tt:ReceiverToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="MediaUri" Type="tt:MediaUri"/>
  </tt:Data>
</tt:MessageDescription>

```

13.5 Service specific fault codes

Table 199 lists the display service specific fault codes. Additionally, each command can also generate a generic fault, see Table 6.

The specific faults are defined as subcode of a generic fault, see 5.11.2.1. The parent generic subcode is the *subcode* at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 199 – Service specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
Env:Sender	ter:InvalidArgVal	Receiver does not exist.	The receiver indicated by ReceiverToken does not exist.
	ter:UnknownToken		
Env:Sender	ter:Action	Maximum number of receivers has been reached.	The maximum supported number of receivers has been reached.
	ter:MaxReceivers		
Env:Sender	ter:InvalidArgVal	The StreamSetup is not supported.	Specification of StreamType or Transport part in ReceiverConfiguration StreamSetup is not supported.
	ter:InvalidStreamSetup		
Env:Sender	ter:Action	It is not possible to delete the receiver.	It is not possible to delete the specified receiver, for example because it is currently in use.
	ter:CannotDeleteReceiver		

14 Display service

A display device has a fixed number of video outputs, each of which may be attached to a monitor. A client can request the video outputs of the device using the DeviceIO service. Each of these outputs is configured with a layout (e.g. single view or split screen). The layout defines a number of video panes, each of which occupies an area of the physical display.

A network video display MAY also have fixed number of audio inputs and audio outputs. Each of these outputs MAY be associated with a pane. Associating an audio input or output with a pane maps the audio and video streams from a transmitter device automatically to the correct outputs. The pane also contains a pointer to a receiver where the necessary information to connect the display device to a transmitter is stored.

The Display Service offers functions to configure the Panes and describe and change the layout of the display device. The possible layouts and coding capabilities of a video output can be requested.

A display device shall support the Display Service as defines in Annex C.5.

14.1 Panes

A Pane is a display area on the monitor that is attached to a video output. A pane has a PaneConfiguration that describes which entities are associated with the pane. The PaneConfiguration includes:

Pane Token: A unique identifier in the display device.

Pane Name: Configuration name.

AudioOutputToken: A pointer to the audio output that is associated with the pane. A client can retrieve the available audio outputs of a device using the GetAudioOutputs command of the DeviceIO service.

AudioSourceToken: A pointer to the audio source that is associated with this pane. The audio connection from a display device to the NVT is established using the backchannel mechanism. A client can retrieve the available audio sources of a device using the GetAudioSources command of the DeviceIO service.

AudioEncoderConfiguration: The configuration of the audio encoder including codec, bitrate and sample rate.

ReceiverToken: A pointer to a Receiver that has the necessary information to receive Data from a Transmitter. This Receiver can be connected and the network video display displays the received data on the specified outputs. A client can retrieve the available Receivers using the GetReceivers command of the Receiver Service.

A client has to configure the pane according to the connection to be established by setting the AudioOutput and/or AudioSourceToken. If a Token is not set, the corresponding session will not be established.

Changing the PaneConfiguration or the parameters of a referenced receiver shall not affect the RTSP connection. If a client intends to apply the new parameters it shall restart the RTSP connection.

The pane layout (see 14.2) of the video output defines if and where (position, size) a Pane is currently visible. The receiver shall only establish a RTSP connection to receive data if the pane is visible. Layout changes shall NOT affect running streams.

14.1.1 GetPaneConfigurations

This command lists all currently defined Panes of a device for a specified video output (regardless if this pane is visible at a moment). A display device shall support the retrieval of its configured panes through this command (see Table 200).

Table 200 – GetPaneConfigurations

GetPaneConfigurations		Request-Response
Message name	Description	
GetPaneConfigurationsRequest	<i>The VideoOutput element specifies the Video Output whose PaneConfigurations are requested.</i> tt:ReferenceToken VideoOutput[1][1]	
GetPaneConfigurationsResponse	<i>Contains a list of defined Panes of the specified VideoOutput. Each VideoOutput has at least one PaneConfiguration.</i> tt:PaneConfiguration PaneConfiguration [1][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	

14.1.2 GetPaneConfiguration

If the pane token is already known this command can be used to get the pane configuration. A display device shall support the retrieval of a specific pane configuration through this command (see Table 201).

Table 201 – GetPaneConfiguration

GetPaneConfiguration		Request-Response
Message name	Description	
GetPaneConfigurationRequest	<i>This message contains the a token of the pane whose configuration is requested. It also conatins a VideoOutput token that specifies the Video Output that contains the requested pane.</i> <i>This message also contains the token of the requested pane</i> tt:ReferenceToken VideoOutput[1][1] tt:ReferenceToken Pane[1][1]	
GetPaneConfigurationResponse	<i>Contains the requested PaneConfiguration</i> tt:PaneConfiguration PaneConfiguration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoPane	<i>The requested pane does not exist</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	

14.1.3 SetPaneConfigurations

This command changes the Configuration of all existing panes in one step. The message contains all PaneConfigurations (modified and not modified ones) of the video output. A display device shall support the modification of its panes configuration through this command (see Table 202).

Table 202 – SetPaneConfigurations

SetPaneConfigurations		Request-Response
Message name	Description	
SetPaneConfigurationsRequest	<i>This message contains the configuration of all panes of the specified VideoOutput.</i> <i>The PaneConfiguration element contains the modified configuration.</i> tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][unbounded]	
SetPaneConfigurationsResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>The configuration is not possible to set</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	

14.1.4 SetPaneConfiguration

This command changes the Configuration of a single pane. A display device shall support the modification of a single pane configuration through this command (see Table 203).

Table 203 – SetPaneConfiguration

SetPaneConfiguration		Request-Response
Message name	Description	
SetPaneConfigurationRequest	<i>This message contains the token of the video output and the new PaneConfiguration .</i> tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][1]	
SetPaneConfigurationResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoPane	<i>The requested pane does not exist</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	
env:Sender ter:InvalidArgVal ter:InvalidConfig	<i>The configuration is not possible to set</i>	

14.1.5 CreatePaneConfiguration

This command creates a pane configuration. A display device that supports dynamic creation of panes shall support the creation of a pane configuration through this command (see Table 204).

Table 204 – CreatePaneConfiguration

CreatePaneConfiguration		Request-Response
Message name	Description	
CreatePaneConfigurationRequest	<i>This message contains the token of the video output and the new PaneConfiguration</i> tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][1]	
CreatePaneConfigurationResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:MaxNumberOfPane	<i>The maximum number of panes is reached</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	
env:Sender ter:InvalidArgVal ter:InvalidConfig	<i>The configuration is not possible to set</i>	

14.1.6 DeletePaneConfiguration

This command deletes a pane configuration. A display device that supports dynamic deletion of panes shall support the deletion of a pane configuration through this command (see Table 205).

Table 205 – DeletePaneConfiguration

DeletePaneConfiguration		Request-Response
Message name	Description	
DeletePaneConfigurationRequest	<i>This message contains the token of the video output and the new PaneConfiguration.</i> tt:ReferenceToken VideoOutput[1][1] tt:ReferenceToken PaneToken[1][1]	
DeletePaneConfigurationResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:FixedPane	<i>It is not possible to delete this pane configuration</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output does not exist</i>	
env:Sender ter:InvalidArgVal ter:NoPane	<i>The requested pane configuration does not exist.</i>	

14.2 Layout

The layout assigns a pane configuration to a certain area of the display. The layout settings directly affect a specific video output. The layout consists of a list of PaneConfigurations and their associated display areas. If the device supports overlapping panes then the order the panes are displayed on the monitor is defined by the order of the PaneConfigurations in the list. The first Pane in the list is the one that is displayed in the foreground.

A device MAY either provide a fixed number of supported layouts or it is possible to configure the layout free.

14.2.1 GetLayout

This command returns the current layout of a video output. A display device shall support the retrieval of its layout through this command (see Table 206).

Table 206 – GetLayout

GetLayout		Request-Response
Message name	Description	
GetLayoutRequest	<i>Contains the VideoOutputToken of the output the display is connected to.</i> tt:ReferenceToken VideoOutput[1][1]	
GetLayoutResponse	<i>Contains the current layout of the VideoOutput</i> tt:Layout Layout[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output indicated with VideoOutput does not exist.</i>	

14.2.2 SetLayout

The SetLayout operation can be used to change the layout of a display (e.g. change from single view to split screen view). A display device shall support the change of layout through this command (see Table 207).

Table 207 – SetLayout

SetLayout		Request-Response
Message name	Description	
SetLayoutRequest	<i>This message contains the token of the video output and the modified layout</i> tt:ReferenceToken VideoOutput [1][1] tt:Layout Layout[1][1]	
SetLayoutResponse	<i>This message is an empty message</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidLayout	<i>It is not possible to set the layout</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested Video Output indicated with VideoOutput does not exist.</i>	

14.3 Display options

14.3.1 General

The display options contain the supported layouts (LayoutOptions) and the decoding and encoding capabilities (CodingCapabilities) of the device. The GetDisplayOptions command returns both, Layout and Coding Capabilities, of a VideoOutput.

14.3.2 LayoutOptions

The LayoutOptions describe the fixed and predefined layouts of a device. If the device does not offer fixed layouts and allows setting the layout free this element is empty.

14.3.3 Coding Capabilities

The network video display is able to decode audio and video streams using suitable decoding algorithms. The network video display supports any audio and video decoders, bitrates and resolution according to the manufacturer's choice.

In order to ensure interoperability between the different devices, this standard mandates the following decoder profiles:

The NVD shall support JPEG QVGA.

The NVD shall support G.711 μ Law (if it supports audio).

These are the same codecs that are mandatory for the NVT.

There are no parameters to configure a decoder; a decoder shall decode all content (according to its capabilities) it receives. In case of decoding errors the decoder should try to request a synchronization point and try to continue decoding. It shall generate an event as defined in 14.4.2.

The CodingCapabilities Element gives an indication about the decoding and encoding capabilities of the device.

14.3.4 GetDisplayOptions

This command lists layout and coding capabilities of a video output. A display device shall support the retrieval of its DisplayOptions through this command (see Table 208).

Table 208 – GetDisplayOptions

GetDisplayOptions		Request-Response
Message name	Description	
GetDisplayOptionsRequest	<i>Contains a PaneToken that indicates what media profile to delete.</i> tt:ReferenceToken VideoOutput[1][1]	
GetDisplayOptionsResponse	<i>This message contains the token of the video output that the options are intended for.</i> tt:LayoutOptions LayoutOptions[0][1] tt:CodingCapabilities CodingCapabilities[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>The requested VideoOutputToken does not exist</i>	

14.4 Events

14.4.1 General

The display service shall dispatch events through the event service.

14.4.2 Decoding error event

The device shall be capable of generating the following event whenever it receives a bitstream that it is not able to decode.

There are several reasons why a decoder is not able to decode the bitstream. The following error codes are defined and shall be used by the device to inform the client about decoder errors:

- 1) “unsupported codec or unsupported codec profile” – the device is not able to decode the bitstream, because the codec or the profile is not supported by the device. The client should the try to reconfigure the transmitter according to the CodingCapabilities of the device;
- 2) “packet error” – there are missing or unexpected packets in the bitstream.

Other vendor specific codes are also allowed.

```

Topic: tns1:VideoDecoder/DecodingError
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="VideoOutputToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>

```



```

<tt:Data>
  <tt:SimpleItemDescription Name="PaneReference"
Type="tt:ReferenceToken"/>
  <tt:SimpleItemDescription Name="Error"
Type="xs:string" minOccurs="0"/>
</tt:Data>
</tt:MessageDescription>

```

14.5 Service specific fault codes

Table 209 lists the display service specific fault codes. Additionally, each command can also generate a generic fault, see Table 6.

The specific faults are defined as subcode of a generic fault, see 5.11.2.1. The parent generic subcode is the *subcode* at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 209 – Service specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Sender	ter:InvalidArgVal	The Video Output does not exist	The requested Video Output does not exist
	ter:NoVideoOutput		
env:Sender	ter:InvalidArgVal	The PaneConfiguration is fixed.	It is not possible to delete this pane configuration
	ter:FixPane		
env:Sender	ter:InvalidArgVal	The maximum number of panes is reached.	It is not possible to create a Pane because the maximum number of panes is reached
	ter:MaximumNumberOfPan es		
env:Sender	ter:InvalidArgVal	The PaneConfiguration does not exist	The requested Pane Configuration does not exist.
	ter:NoPane		
env:Sender	ter:InvalidArgVal	It is not possible to set the configuration	The requested configuration is not supported by the device
	ter:InvalidConfig		
env:Sender	ter:Action	It is not possible to set the layout	The requested layout is not supported by the device
	ter:InvalidLayout		

15 Event handling

An event is an action or occurrence detected by a device that a client can subscribe to. Events are handled through the event service. A device shall provide an event services as defined in C.6. Both an NVT and NVC shall support [WS-Addressing] for event services.

Event Handling in this standard is based on the [WS-BaseNotification] and [WS-Topics] specifications. This standard requires the implementation of the basic notification interface as described in 15.1, which conforms completely to the [WS-BaseNotification] specification. In addition, the device shall implement the Real-time Pull-Point Notification Interface and the Notification Streaming Interface as introduced in 15.2 and 15.3, respectively.

This standard introduces notification message extensions that allow a client to track object properties (such as video analytics object properties) through events. Properties are defined in 15.4.

The description of event payload and their filtering within subscriptions is discussed in 15.5. Subclause 15.6 describes how Synchronization Point can be requested by clients using one of the three Notification Interfaces. Subclause 15.7 describes the integration of Topics. Subclause 15.9 discusses the handling of faults.

The last subclause demonstrates in detail the usage of the Real-Time Pull-Point Notification Interface including Message Filtering and Topic Set. Examples for the basic notification interface can be found in the corresponding [WS-BaseNotification] specification.

15.1 Basic notification interface

Subclause 15.1.1 briefly introduces the Basic Notification Interface of the [WS-BaseNotification] specification. Subclause 15.1.2 summarizes the mandatory and the optional interfaces of the [WS-BaseNotification] specification.

15.1.1 General

The following logical entities participate in the notification pattern:

- client: implements the NotificationConsumer interface;
- event service: implements the NotificationProducer interface;
- subscription manager: implements the BaseSubscriptionManager interface.

The Event Service and the Subscription Manager should be instantiated on a device.

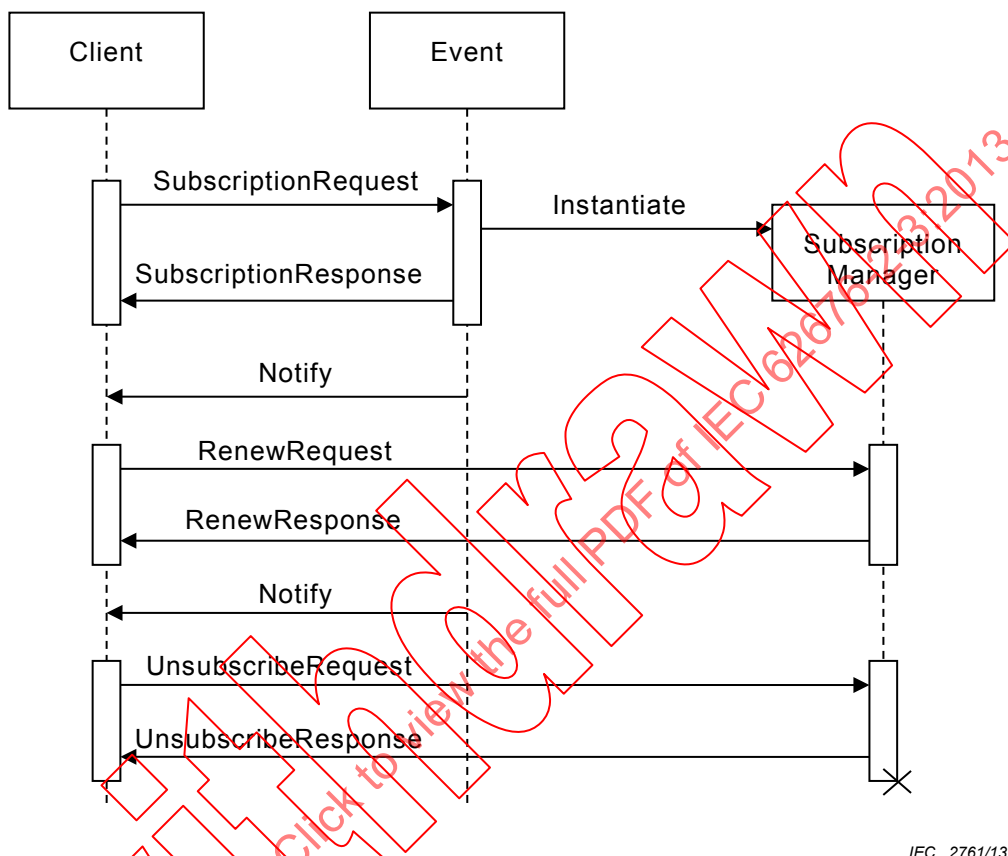
Typical messages exchanged between the entities are shown in the sequence diagram in Figure 21. First, the client establishes a connection to the Event Service. The client can then subscribe for certain notifications by sending a SubscriptionRequest. If the Event Service accepts the Subscription, it dynamically instantiates a SubscriptionManager representing the Subscription. The Event Service shall return the WS-Endpoint-Address of the SubscriptionManager in the SubscriptionResponse.

In order to transmit notifications matching the Subscription, another connection is established from the Event Service to the client. Via this connection, the Event Service sends a one-way Notify message to the NotificationConsumer interface of the client. Corresponding notifications can be sent at any time by the Event Service to the client, while the Subscription is active.

To control the Subscription, the client directly addresses the SubscriptionManager returned in the SubscriptionResponse. In the SubscriptionRequest the client can specify a termination time. The SubscriptionManager is automatically destroyed when the termination time is

reached. RenewRequests can be initiated by the client in order to postpone the termination time. The client can also explicitly terminate the SubscriptionManager by sending an UnsubscribeRequest. After a successful Unsubscription, the SubscriptionManager no longer exists.

The interaction between EventService and SubscriptionManager is not further specified by the [WS-BaseNotification] and is up to the implementation of the device.



IEC 2761/13

Figure 21 – Sequence diagram for the base notification interface

15.1.2 Requirements

This subclause details those interfaces of the [WS-BaseNotification] that a device shall provide.

An ONVIF compliant device shall support the NotificationProducer Interface of the [WS-BaseNotification]. As a result, the NotificationProducer Resource Properties are OPTIONAL (see 15.5). The device shall support TopicExpression and MessageContent filters with at least the dialects described in 15.5.5 and 15.7.3. If the device does not accept the InitialTerminationTime of a subscription, it shall provide a valid InitialTerminationTime within the Fault Message. The device shall be able to provide notifications using the Notify wrapper of the [WS-BaseNotification] specification. The SubscriptionPolicy wsnt:UseRaw is OPTIONAL for the device. Although the [WS-BaseNotification] has CurrentTime and TerminationTime as optional elements in a SubscribeResponse, an ONVIF compliant device shall list them in SubscribeResponses. The device MAY respond to any GetCurrentMessage request with a Fault message indicating that no current message is available on the requested topic.

The implementation of the Pull-Point Interface of the [WS-BaseNotification] on a device is OPTIONAL.

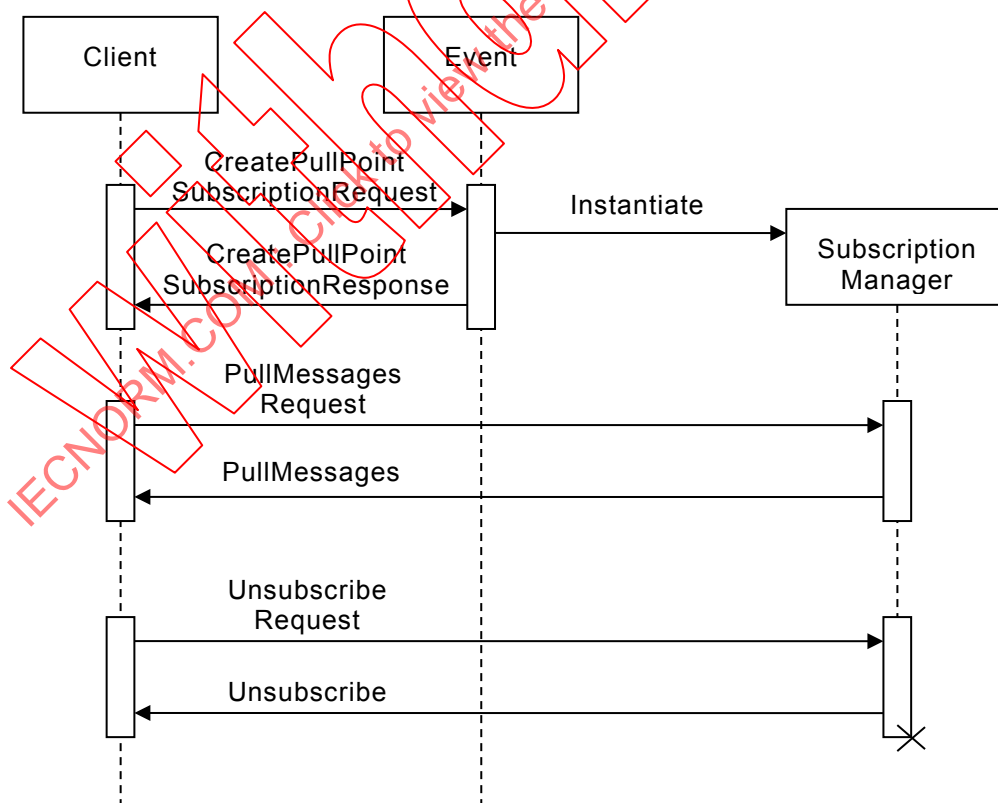
An ONVIF compliant device shall implement the Base Subscription Manager Interface of the [WS-BaseNotification] specification consisting of the Renew and Unsubscribe operations. The Pausable Subscription Manager Interface is OPTIONAL. The implementation of Subscriptions as WS-Resources is OPTIONAL.

15.2 Real-time Pull-Point Notification Interface

This subclause introduces the Real-time Pull-Point Notification Interface. This interface provides a firewall friendly notification interface that enables real-time polling and initiates all client communications.

This interface is used in the following way:

- 1) the client asks the device for a PullPointSubscription with the CreatePullPointSubscriptionRequest message. The request contains a detailed description of the Subscription. The ConsumerReference shall be omitted, in contrast to the subscription of the Basic Notification Interface (see 15.1);
- 2) the device evaluates the Subscription and returns either a CreatePullPointSubscriptionResponse when the Subscription is accepted or one of the Fault codes;
- 3) if the Subscription is accepted, the response contains a WS-EndpointReference to a SubscriptionManager. This WS-Endpoint shall provide a PullMessages operation, which is used by the client to retrieve Notifications and by the Base Subscription Manager Interface described in the [WS-BaseNotification] specification. The Base Subscription Manager Interface consists of PullMessages, Renew and Unsubscribe operations. The sequence diagram of the interaction is shown in Figure 22. The PullMessagesRequest contains Timeout and MessageLimit parameters;



IEC 2762/13

Figure 22 – Sequence diagram for the Real-time Pull-Point Notification Interface

- 4) the device shall immediately respond with notifications that have been aggregated on behalf of the client. If there are no aggregated notifications, the device waits for its

response until either a notification is produced for the client or the specified Timeout is exceeded. In any case, the response will contain, at most, the number of notifications specified by the MessageLimit parameter. The client can poll the notifications in real-time when it starts a new PullMessagesRequest immediately after each PullMessagesResponse;

- 5) if neither a termination time nor a relative termination time is set in the CreatePullPointSubscriptionRequest, each PullMessagesRequest shall be interpreted as a keep-alive for the corresponding PullPointSubscription. The termination time is recomputed according to the relative termination time if available or according to a device internal default value. To inform the client about the updated termination time, the PullMessagesResponse shall contain the CurrentTime and TerminationTime elements. When the PullMessagesRequest is used as keep-alive for the corresponding PullPointSubscription, the RenewRequest, defined by the [WS-BaseNotification], need not be called by a client. Nevertheless, the device shall support it for the PullPointSubscription.

15.2.1 Create pull point subscription

The device shall provide the CreatePullPointSubscription command given in Table 210.

Table 210 – CreatePullPointSubscription command

CreatePullPointSubscription		Request-response
Message name	Description	
CreatePullPointSubscriptionRequest	<i>This message contains the same elements as the SubscriptionRequest of the [WS-BaseNotification] without the ConsumerReference.</i> wsnt:FilterType Filter [0][1] wsnt:AbsoluteOrRelativeTimeType InitialTerminationTime [0][1] xs:any SubscriptionPolicy [0][1]	
CreatePullPointSubscriptionResponse	<i>The response contains the same elements as the SubscriptionResponse of the [WS-BaseNotification]:</i> wsa:EndpointReferenceType SubscriptionReference [1][1] xs:dateTime CurrentTime [1][1] xs:dateTime TerminationTime [1][1]	
Fault codes	Description	
	<i>The same faults as for Subscription Request of the [WS-BaseNotification] are used.</i>	

15.2.2 Pull messages

The device shall provide the following PullMessages command for all SubscriptionManager endpoints returned by the CreatePullPointSubscription command (see Table 211).

Table 211 – PullMessages command

PullMessages		Request-response
Message name	Description	
PullMessagesRequest	<i>This message shall be addressed to a SubscriptionManager in order to pull notifications:</i> xs:duration Timeout [1][1] xs:int MessageLimit [1][1]	
PullMessagesResponse	<i>The response contains a list of notifications together with an updated TerminationTime for the SubscriptionManager:</i> xs:dateTime CurrentTime [1][1] xs:dateTime TerminationTime [1][1] wsnt:NotificationMessageHolderType NotificationMessage [0][unbounded]	
PullMessagesFaultResponse	<i>The Timeout exceeds the upper limit supported by the device. The Fault Message shall contain the upper limits for both parameters.</i> xs:duration MaxTimeout[1][1] xs:int MaxMessageLimit[1][1]	
Fault codes	Description <i>No specific fault codes.</i>	

15.3 Notification streaming interface

Clause 10 describes the creation, deletion and modification of metadata configurations. Certain metadata configurations can contain multiple subscriptions whose structure is the same as that for a notification subscription. When a metadata configuration containing subscriptions has been assigned to a profile, a client uses that profile to get an RTP stream that includes the configured notifications as metadata. The notification streaming via RTP shall be implemented by an ONVIF compliant device.

The [WS-BaseNotification] defines the element `wsnt:NotificationMessage` to pack the Message Payload, the Topic and the ProducerReference. The structure of this message is the same as that for direct notification requests (the format is described in 15.5). Multiple instances of the `wsnt:NotificationMessage` elements can be placed within a metadata document introduced in the Real-time Viewing section.

There is no explicit SubscriptionReference with streaming notifications. Therefore, the `wsnt:NotificationMessage` shall NOT contain the SubscriptionReference element.

15.4 Properties

A Property is a collection of name and value pairs representing a unique and addressable set of data. They are uniquely identified by the combination of their Topic, Source and Key values and are packaged like ordinary events. A Property also contains an additional flag, stating whether it is newly created, has changed or has been deleted.

- When a client subscribes to a topic representing a certain property, the device shall provide notifications informing the client of all objects with the requested property, which are alive at the time of the subscription. An client *can* also request the values of all currently alive properties the client has subscribed to at any time by asking for a synchronization point (see 15.6).

The property interface is defined in this standard in order to group all property related events together and to present uniformly to clients. It is RECOMMENDED to use the property interface wherever applicable. Subclause 15.5 explains the structure of events and properties in detail.

15.4.1 Property example

The following video analytics example demonstrates the dynamic behaviour of properties: The rule engine interface of the video analytics detector can define fields. Such a detector field is described by a polygon in the image plane. For each object in the scene, the rule engine determines which objects are within the polygon. A client can access this information by subscribing to the corresponding ObjectsInside property of the detector field. Each time an object appears in the scene, a new ObjectsInside property is created. The client is informed by a corresponding “property created” notification indicating if the object appeared inside or outside the polygon. Each time an object enters or leaves the polygon, a “property changed” notification is produced indicating that the ObjectsInside property for this object has changed. When an object leaves the scene, the corresponding ObjectsInside property is deleted and the client is informed via a “property deleted” notification.

15.5 Notification structure

The following code is the schema for the wsnt:NotificationMessage [WS-BaseNotification]:

```
<xs:complexType name="NotificationMessageHolderType" >
  <xs:sequence>
    <xs:element ref="wsnt:SubscriptionReference" minOccurs="0" />
    <xs:element ref="wsnt:Topic" minOccurs="0" />
    <xs:element ref="wsnt:ProducerReference" minOccurs="0" />
    <xs:element name="Message">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##any" processContents="lax" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:element name="NotificationMessage"

  type="wsnt:NotificationMessageHolderType"/>
```

This corresponds to the following XML structure:

```
<wsnt:NotificationMessage>
  <wsnt:SubscriptionReference>
    wsa:EndpointReferenceType
  </wsnt:SubscriptionReference>
  <wsnt:Topic Dialect="xs:anyURI">
    ...
  </wsnt:Topic>?
  <wsnt:ProducerReference>
    wsa:EndpointReferenceType
  </wsnt:ProducerReference>
  <wsnt:Message>
```



```

...
</wsnt:Message>
</wsnt:NotificationMessage>

```

where the `wsnt:Message` element contains the actual notification payload. The XML type of the `Message` element can be specified within a `TopicTree` definition (see 15.7).

Subclause 15.5.1 gives an overview of the information a client retrieves through notifications. Subclause 15.5.2 gives a detailed formatting of the `Message` payload, and 15.5.4 introduces a description language for the `Message` payload. Subclause 15.5.5 defines the grammar used in a subscription to filter notifications by their `Message` content.

15.5.1 Notification information

A notification answers at least the following questions:

- When did it happen?
- Who produced the event?
- What happened?

The “when” question is answered by adding a time attribute to the `Message` element of the `NotificationMessage`. An ONVIF compliant device shall include the time attribute to the `Message` element.

The “who” question is split into two parts. One part is the `WS-Endpoint` which identifies the device or a service within the device where the notification has been produced. Therefore, the `WS-Endpoint` should be specified within the `ProducerReference` Element of the `NotificationMessage`. The second part is the identification of the component within the `WS-Endpoint`, which is responsible for the production of the notification. Depending on the component multiple parameters or none may be needed to identify the component uniquely. These parameters are placed as `Items` within the `Source` element of the `Message` container.

The “what” question is answered in two steps. First, the `Topic` element of the `NotificationMessage` is used to categorize the Event. Second, items are added to the `Data` element of the `Message` container in order to describe the details of the Event.

When the topic points to properties (see 15.4), the client uses the `NotificationProducer`, the `Topic`, the `Source` `Items` and optional `Key` `Items` (see 15.5) in order to identify the property. These values shall result in a unique identifier.

Event example

- The subsequent example demonstrates the different parts of the notification:

```

<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
  tns1:PTZController/PTZPreset/Reached
</wsnt:Topic>
<wsnt:Message>
  <tt:Message UtcTime="...">
    <tt:Source>
      <tt:SimpleItem Name="PTZConfigurationToken"
Value="PTZConfig1"/>
    </tt:Source>
    <tt>Data>
      <tt:SimpleItem Name="PresetToken" Value="Preset5"/>
      <tt:SimpleItem Name="PresetName"
Value="ParkingLot"/>
    </tt>Data>
  </tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>

```


The Item “PTZConfigurationToken” identifies uniquely the component, which is responsible for the detection of the Event. In this example, the component is a PTZ Node referenced by the PTZ Configuration “PTZConfig1”. The event `tns1:PTZController/PTZPreset/Reached` indicates that the PTZ unit has arrived at a preset. The data block contains the information which preset it is. Thereby, the Preset is identified by a PresetToken “Preset5” which is named “PresetName”.

15.5.2 Message format

The Message element of the NotificationMessage is defined in [ONVIF Schema]. The definition is presented below⁴:

```
<xs:element name="Message" type="Message">
  <xs:element name="Message">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Source" type="tt:ItemList" minOccurs="0"/>
        <xs:element name="Key" type="tt:ItemList" minOccurs="0"/>
        <xs:element name="Data" type="tt:ItemList" minOccurs="0"/>
        ...
      </xs:sequence>
      <xs:attribute name="UtcTime" type="xs:time" use="required"/>
      <xs:attribute name="PropertyOperationType" type="tt:PropertyOperationType"/>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="ItemList">
    <xs:sequence>
      <xs:element name="SimpleItem" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="Name" type="xs:string" use="required"/>
          <xs:attribute name="Value" type="xs:anySimpleType" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="ElementItem" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:any namespace="##any"/>
          </xs:sequence>
          <xs:attribute name="Name" type="xs:string" use="required"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>

  <xs:simpleType name="PropertyOperationType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Initialized"/>
      <xs:enumeration value="Deleted"/>
      <xs:enumeration value="Changed"/>
    </xs:restriction>
  </xs:simpleType>
```

The Items within the Message element are grouped into three categories: Source, Key, and Data. The Key group shall NOT be used by notifications which are not related to properties. Multiple Simple and Element Items can be placed within each group. Each Item has a name and a value. In the case of an ElementItem, the value is expressed by one XML element within the ElementItem element. In the case of a SimpleItem, the value shall be specified by

⁴ Please note that the schema is included here for *information only*. [ONVIF Schema] contains the normative schema definition.

the value attribute. The name of all Items shall be unique within all Items contained in any group of this Message.

It is RECOMMENDED to use SimpleItems instead of ElementItems whenever applicable, since SimpleItems ease the integration of Messages into a generic client. The exact type information of both Simple and ElementItems can be extracted from the TopicSet (see 15.7), where each topic can be augmented by a description of the message payload.

The PropertyOperation shall be present when the notification relates to a property. The operation mode “Initialized” shall be used to inform a client about the creation of a property. The operation mode “Initialized” shall be used when a synchronization point has been requested.

15.5.3 Property example, continued

The example in 15.4.1 required an optional Key Item. The example in this subclause demonstrates the application of Key Items. The rule engine can contain FieldDetector rules. These rules define an ObjectsInside property for each object in the scene. When a new object appears outside of such a Field, the following notification is produced:

```
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
  tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
  <tt:Message UtcTime="..." PropertyOperation="Initialized">
    <tt:Source>
      <tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="Rule" Value="myImportantField"/>
    </tt:Source>
    <tt:Key>
      <tt:SimpleItem Name="ObjectId" Value="5"/>
    </tt:Key>
    <tt>Data>
      <tt:SimpleItem Name="IsInside" Value="false"/>
    </tt>Data>
  </tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
```

The Source Items describe the Rule which produced the notification. When multiple objects are in the scene, each of these objects has its own ObjectsInside property. Therefore, the Object ID is used as an additional Key Item in order to make the property unique. The IsInside Item is a Boolean value indicating whether the object is inside or outside of the Field.

When the object enters the Field, the rule produces a “property changed” message and resembles the following:

```
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
  tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
  <tt:Message UtcTime="..." PropertyOperation="Changed">
    <tt:Source>
      <tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="Rule" Value="myImportantField"/>
    </tt:Source>
    <tt:Key>
      <tt:SimpleItem Name="ObjectId" Value="5"/>
    </tt:Key>
    <tt>Data>
```

```

        <tt:SimpleItem Name="IsInside" Value="true"/>
    </tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
Finally, when the object leaves the scene, a "property deleted" message is
produced:
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
    tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
    <tt:Message UtcTime="..." PropertyOperation="Deleted">
        <tt:Source>
            <tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
            <tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
            <tt:SimpleItem Name="Rule" Value="myImportantField"/>
        </tt:Source>
        <tt:Key>
            <tt:SimpleItem Name="ObjectId" Value="5"/>
        </tt:Key>
    </tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>

```

In this case, the Data item can be omitted because the object and its corresponding property no longer exists.

15.5.4 Message description language

The structure of the Message payload was introduced in the previous subclause. The structure contains three groups: Source, Key, and Data. Each group contains a set of Simple and ElementItems. For each topic, a device can describe which Item will be part of a notification produced by this topic using a message description language. The following description language describes the mandatory message items⁵:

```

<xs:complexType name="MessageDescription">
    <xs:sequence>
        <xs:element name="Source" type="tt:ItemListDescription"
            minOccurs="0"/>
        <xs:element name="Key" type="tt:ItemListDescription" minOccurs="0"/>
        <xs:element name="Data" type="tt:ItemListDescription" minOccurs="0"/>
        ...
    </xs:sequence>
    <xs:attribute name="IsProperty" type="xs:boolean"/>
</xs:complexType>

<xs:complexType name="ItemListDescription">
    <xs:sequence>
        <xs:element name="SimpleItemDescription"
            minOccurs="0" maxOccurs="unbounded">
            <xs:complexType>
                <xs:attribute name="Name" type="xs:string" use="required"/>
                <xs:attribute name="Type" type="xs:QName" use="required"/>
            </xs:complexType>
        </xs:element>
        <xs:element name="ElementItemDescription"
            minOccurs="0" maxOccurs="unbounded">
            <xs:complexType>
                <xs:attribute name="Name" type="xs:string" use="required"/>
                <xs:attribute name="Type" type="xs:QName" use="required"/>
            </xs:complexType>
        </xs:element>
    </xs:sequence>

```

⁵ Please note that the schema is included here for *information only*. [ONVIF Schema] contains the normative schema definition.

```

</xs:sequence>
</xs:complexType>

```

The Name attribute of an Item shall be unique within all Items independent from the group (Source, Key, Data) they are coming from. The IsProperty attribute shall be set to true when the described Message relates to a property. If the Message, however, does not relate to a property, the Key group shall NOT be present. The Type attribute of a SimpleItemDescriptor shall match the SimpleElement definition of an XML schema. Similarly, the Type attribute of an ElementItemDescriptor shall match a global element declaration of an XML schema.

The location of all schema files used to describe Message payloads are listed in the GetEventPropertiesResponse message in 15.8.

Message description example

The following code is an example of a Message Description corresponding to the Property example of 15.5.3:

```

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescriptionDescription
      Name="VideoSourceConfigurationToken"
      Type="tt:ReferenceToken"/>
    <tt:SimpleItemDescriptionDescription
      Name="VideoAnalyticsConfigurationToken"
      Type="tt:ReferenceToken"/>
    <tt:SimpleItemDescriptionDescription Name="Rule"
      Type="xs:string"/>
  </tt:Source>
  <tt:Key>
    <tt:SimpleItemDescriptionDescription Name="ObjectId"
      Type="tt:ObjectRefType"/>
  </tt:Key>
  <tt>Data>
    <tt:SimpleItemDescriptionDescription Name="IsInside"
      Type="xs:boolean"/>
  </tt>Data>
</tt:MessageDescription>

```

15.5.5 Message content filter

In the Subscription request, a client can filter notifications by TopicExpression (see 15.7.3) and by MessageContent. For the latter, the [WS-BaseNotification] proposes the XPath 1.0 dialect. Due to the specific Message structure required by this specification, the specification requires a subset of the XPath 1.0 syntax. An ONVIF compliant device shall implement the subset of XPath 1.0. The corresponding dialect can be referenced with the following URI:

Dialect=<http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter>

Precedence and associativity:

The 'and' operation has higher precedence than the 'or' operation. Both 'and' and 'or' operations are left associative.

The precedence and associativity of 'and' and 'or' operations in the following grammar definition are identical to XPath 1.0 specifications.

The structure of the Expressions is as follows:

[1] Expression::= BoolExpr | Expression 'and' Expression
 | Expression 'or' Expression | '(' Expression ')' | 'not' '(' Expression ')'

[2] BoolExpr::= 'boolean' '(' PathExpr ')'

[4] Prefix ::= NamespacePrefix ':' | "

[6] AttrExpr ::= AttrComp | AttrExpr 'and' AttrExpr | AttrExpr 'or' AttrExpr | '(' AttrExpr ')' | 'not' '(' AttrExpr ')'

[8] Attribute ::= '@Name' | '@Value'

Finally, arbitrary boolean combinations of these tests are possible. The following expressions can be formulated:

- ```
boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken" and
 @Value="1"])
```

- ```
not( boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken" ] ) )
```

- ```
boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken" and
 @Value="2"])
and
boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken" and
 @Value="1"])
```

- ```
boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken" and
                                @Value="1" ] )
and
not (boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken"
1 ] ))
```

- ```
boolean(//tt:SimpleItem[@Name="IsInside" and @Value="true"])
and
boolean(//tt:SimpleItem[@Name="Rule" and
@Value="myImportantField"])
```

Properties, introduced in 15.4, inform a client about property creation, changes and deletion in a uniform way. When a client wants to synchronize its properties with the properties of the device, it can request a synchronization point which repeats the current status of all

properties to which a client has subscribed. The PropertyOperation of all produced notifications is set to “Initialized” (see 15.5). The Synchronization Point is requested directly from the SubscriptionManager which was returned in either the SubscriptionResponse or in the CreatePullPointSubscriptionResponse. The property update is transmitted via the notification transportation of the notification interface. The following operation given in Table 212 shall be provided by all Subscription Manager Endpoints:

**Table 212 – SetSynchronizationPoint command**

| SetSynchronizationPoint          |                                    | Request-Response |
|----------------------------------|------------------------------------|------------------|
| Message name                     | Description                        |                  |
| SetSynchronizationPoint-Request  | <i>This message is empty.</i>      |                  |
| SetSynchronizationPoint-Response | <i>This message is empty.</i>      |                  |
| Fault codes                      | Description                        |                  |
|                                  | <i>No command specific faults!</i> |                  |

When a client uses the notification streaming interface, the client should use the SetSynchronizationPoint operation defined in the media service, see 11.18.

## 15.7 Topic structure

This standard extends the Topic framework defined in the [WS-Topics] specification. Subclause 15.7.1 describes an ONVIF Topic Namespace, which should be taken as a basis for vendor specific topics. The Annex A shows typical examples for such extensions.

Subclause 15.7.2 defines an interface to topic properties. This interface shall be implemented by an ONVIF compliant device. Subclause 15.7.3 incorporates the Message Description Language defined in 15.5.4 into the TopicSet structure. All topics grown from the ONVIF Topic Namespace describes the type of a topic according to 15.7.3. Subclause 15.7.3 defines the Topic Expression Dialects which are supported by a device.

### 15.7.1 ONVIF topic namespace

The [WS-Topics] specification distinguishes between the definition of a Topic Tree belonging to a certain Topic Namespace and the Topic Set supported by a certain Web Service. This distinction allows vendors to refer to a common Topic Namespace while only using a portion of the defined Topics.

If the Topic Tree of an existing Topic Namespace covers only a subset of the topics available by a device, the Topic Tree can be grown by defining a new Topic Namespace. A new Topic Namespace is defined by appending a new topic to an existing Topic Namespace as described in the [WS-Topics] specification.

The following root topics are defined in the ONVIF Namespace. All notifications referring to these topics shall use the Message Format as described in 15.5.2.

```
<wstop:TopicNamespace name="ONVIF"
 targetNamespace="http://www.onvif.org/ver10/topics" >
 <wstop:Topic name="Device"/>
 <wstop:Topic name="VideoSource"/>
 <wstop:Topic name="VideoEncoder"/>
 <wstop:Topic name="VideoAnalytics"/>
 <wstop:Topic name="RuleEngine"/>
```

```

<wstop:Topic name="PTZController"/>
<wstop:Topic name="AudioSource"/>
<wstop:Topic name="AudioEncoder"/>
<wstop:Topic name="UserAlarm"/>
<wstop:Topic name="MediaControl"/>
<wstop:Topic name="RecordingConfig"/>
<wstop:Topic name="RecordingHistory"/>
<wstop:Topic name="VideoOutput"/>
<wstop:Topic name="AudioOutput"/>
<wstop:Topic name="VideoDecoder"/>
<wstop:Topic name="AudioDecoder"/>
<wstop:Topic name="Receiver"/>
</wstop:TopicNamespace>

```

### 15.7.2 Topic type information

The type information is added below a topic element by adding a MessageDescription element of type MessageDescriptionType defined in 15.5.4. Topic elements can be identified by the wstop:topic attribute with value "true".

The following example demonstrates how Topics of a TopicSet are augmented with Message Descriptions:

```

<tnsl:RuleEngine wstop:topic="true">
 <tnsl:LineDetector wstop:topic="true">
 <tnsl:Crossed wstop:topic="true">
 <tt:MessageDescription>
 <tt:Source>
 <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
 Type="tt:ReferenceToken"/>
 <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
 Type="tt:ReferenceToken"/>
 <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
 </tt:Source>
 <tt:Data>
 <tt:SimpleItemDescription Name="ObjectId"
 Type="tt:ObjectRefType"/>
 </tt:Data>
 </tt:MessageDescription>
 </tnsl:Crossed>
 </tnsl:LineDetector>
 <tnsl:FieldDetector wstop:topic="true">
 <tnsl:ObjectsInside wstop:topic="true">
 <tt:MessageDescription IsProperty="true">
 <tt:Source>
 <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
 Type="tt:ReferenceToken"/>
 <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
 Type="tt:ReferenceToken"/>
 <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
 </tt:Source>
 <tt:Key>
 <tt:SimpleItemDescription Name="ObjectId"
 Type="tt:ObjectRefType"/>
 </tt:Key>
 <tt:Data>
 <tt:SimpleItemDescription Name="IsInside" Type="xs:boolean"/>
 </tt:Data>
 </tt:MessageDescription>
 </tnsl:ObjectsInside>
 </tnsl:FieldDetector>
</tnsl:RuleEngine>

```

### 15.7.3 Topic filter

An ONVIF compliant device shall support the Concrete Topic Expressions defined in the [WS-Topics] specification. This specification defines the identification of a specific Topic within



Topic Trees. The following Dialect shall be specified when a Concrete Topic Expression is used as TopicExpression of a Subscription Filter:

`http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete`

The following Topic Expression syntax shall be supported by a device.

The syntax extends the Concrete Topic Expressions by an “or” operation and topic subtree matching string. This extended syntax allows selection of an arbitrary TopicSet within a single Subscription. The grammar is described in the same way as the Topic Expressions of the [WS-Topics 1.3] specification:

[3] TopicExpression ::= TopicPath ('|' TopicPath)\*

[4] TopicPath ::= RootTopic ChildTopicExpression\* ('//.')?

[5] RootTopic ::= QName

If a namespace prefix is included in the RootTopic, it shall correspond to a valid Topic Namespace definition and the local name shall correspond to the name of a root Topic defined in that namespace.

[6] ChildTopicExpression ::= '/' ChildTopicName

[7] ChildTopicName ::= QName | NCName

The NCName or local part of the QName shall correspond to the name of a Topic within the descendant path from the RootTopic, where each forward slash denotes another level of child Topic elements in the path.

In order to reference this TopicExpression Dialect, the following URI shall be used:

Dialect=`http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet`

If the TopicExpression ends with the characters “//.” this indicates that the TopicExpression matches a Topic sub-tree. For example:

“tns1:RuleEngine/FieldDetector//.”

This identifies the sub-tree consisting of tns1:RuleEngine/FieldDetector and all its descendents.

The following examples demonstrate the usage of the ConcreteSet topicExpression:

- Look for notifications which have the VideoAnalytics topic as parent topic:

```
<wsnt:TopicExpression Dialect =
```

- `"http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet" >`
- `tns1:VideoAnalytics//.`
- `</wsnt:TopicExpression>`

- Look for notifications which have the VideoAnalytics topic or the RuleEngine as parent topic:

```
<wsnt:TopicExpression Dialect =
```

- `"http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet" >`



- `tns1:VideoAnalytics//.|tns1:RuleEngine//.`
  - `</wsnt:TopicExpression>`
  - Look for notifications produced by either a LineDetector or a FieldDetector:
- ```
<wsnt:TopicExpression Dialect =
```
- `"http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">`
 - `tns1:RuleEngine/FieldDetector//.|tns1:RuleEngine/LineDetector//.`
 - `</wsnt:TopicExpression>`

15.8 Get event properties

The [WS-BaseNotification] specification defines a set of OPTIONAL WS-ResourceProperties. This specification does not require the implementation of the WS-ResourceProperty interface. Instead, the subsequent direct interface shall be implemented by an ONVIF compliant device in order to provide information about the FilterDialects, Schema files and topics supported by the device (see Table 213).

Table 213 – GetEventProperties command

GetEventProperties		Request-response
Message name	Description	
GetEventPropertiesRequest	<i>This is an empty message.</i>	
GetEventPropertiesResponse	<code>xs:anyURI TopicNamespaceLocation [1][unbounded]</code> <code>xs:boolean FixedTopicSet [1][1]</code> <code>wstop:TopicSetType TopicSet [1][1]</code> <code>xs:anyURI TopicExpressionDialect [1][unbounded]</code> <code>xs:anyURI MessageContentFilterDialect [1][unbounded]</code> <code>xs:anyURI ProducerPropertiesFilterDialect [0][unbounded]</code> <code>xs:anyURI MessageContentSchemaLocation [1][unbounded]</code>	
Fault codes	Description	
	<i>No command specific faults!</i>	

An ONVIF compliant device shall respond and declare if its TopicSet is fixed or not, which Topics are provided, and which Dialects are supported.

The following TopicExpressionDialects are mandatory for an ONVIF compliant device (see 15.7.3):

- `http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete`
- `http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet`
- The following MessageContentFilterDialects are mandatory for the an ONVIF compliant device (see 15.5.5):

<http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter>

This specification does not require the support of any ProducerPropertiesDialect by a device.

The Message Content Description Language, introduced in 15.5.4, allows referencing of vendor-specific types. In order to ease the integration of such types into a client application,

the `GetEventPropertiesResponse` shall list all URI locations to schema files whose types are used in the description of notifications, with `MessageContentSchemaLocation` elements. This list shall at least contain the URI of the ONVIF schema file.

15.9 SOAP fault messages

If a device encounters a failure while processing [WS-BaseNotification] messages from either a client or Subscription Manager, then the device shall generate a SOAP 1.2 fault message.

All SOAP 1.2 fault messages shall be generated according to [WS-BaseNotification] and [WS-Topics] specifications.

15.10 Notification example

The following example is a complete communication pattern for notifications. It uses the Real-time Pull-Point Notification Interface to receive notifications.

15.10.1 GetEventPropertiesRequest

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/GetEventPropertiesRequest
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:GetEventProperties>
    </tet:GetEventProperties>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.2 GetEventPropertiesResponse

In this example, the device response uses the ONVIF topic namespace (the description can be downloaded from <http://www.onvif.org/onvif/ver10/topics/topicns.xml>). The topic set does not change over time and consists of the single topic `tns1:RuleEngine/LineDetector/Crossed`. The Message associated with this topic contains information about the `VideoSourceConfigurationToken`, the `VideoAnalyticsConfigurationToken` and the object which has crossed the line. The device supports two `TopicExpressionDialects`.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1"
  xmlns:tns1="http://www.onvif.org/ver10/topics"
  xmlns:tt="http://www.onvif.org/ver10/schema">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/GetEventPropertiesResponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:GetEventPropertiesResponse>
      <tet:TopicNamespaceLocation>
        http://www.onvif.org/onvif/ver10/topics/topicns.xml
      </tet:TopicNamespaceLocation>
    </tet:GetEventPropertiesResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

```

<wsnt:FixedTopicSet>
  true
</wsnt:FixedTopicSet>
<wstop:TopicSet>
  <tnsl:RuleEngine>
    <tnsl:LineDetector>
      <tnsl:Crossed wstop:topic="true">
        <tt:MessageDescription>
          <tt:Source>
            <tt:SimpleItemDescription
Name="VideoSourceConfigurationToken"
                                Type="tt:ReferenceToken"/>
            <tt:SimpleItemDescription
Name="VideoAnalyticsConfigurationToken"
                                Type="tt:ReferenceToken"/>
          </tt:Source>
          <tt:Data>
            <tt:SimpleItemDescription Name="ObjectId"
                                Type="tt:ObjectRefType"/>
          </tt:Data>
        </tt:MessageDescription>
      </tnsl:Crossed>
    </tnsl:LineDetector>
  </tnsl:RuleEngine>
</wstop:TopicSet>
<wsnt:TopicExpressionDialect>
  http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet
</wsnt:TopicExpressionDialect>
<wsnt:TopicExpressionDialect>
  http://docs.oasis-open.org/wsnt/t-1/TopicExpression/ConcreteSet
</wsnt:TopicExpressionDialect>
<wsnt:MessageContentFilterDialect>
  http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter
</wsnt:MessageContentFilterDialect>
<tt:MessageContentSchemaLocation>
  http://www.onvif.org/onvif/ver10/schema/onvif.xsd
</tt:MessageContentSchemaLocation>
</tet:GetEventPropertiesResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.10.3 CreatePullPointSubscription

A client can subscribe to specific notifications with the information from the TopicProperties. The following XML example shows the subscription for notifications produced by the Rule Engine of the device. The client reacts only to notifications that reference VideoAnalyticsConfiguration "2" and VideoSourceConfiguration "1". The Subscription has a timeout of one minute. If the subscription is not explicitly renewed or messages are not pulled regularly, it will be terminated automatically after this time.

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1"
  xmlns:tnsl="http://www.onvif.org/ver10/topics">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/CreatePullPointSubscriptionRequest
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:CreatePullPointSubscription>
      <tet:Filter>
        <wsnt:TopicExpression
Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">

```

```

        tns1:RuleEngine//.
    </wsnt:TopicExpression>
    <wsnt:MessageContent
Dialect="http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter">
        boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken"
            and @Value="2"] ) and
        boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken"
            and @Value="1"] )
    </wsnt:MessageContent>
</tet:Filter>
<tet:InitialTerminationTime>
    PT1M
</tet:InitialTerminationTime>
</tet:CreatePullPointSubscription>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.10.4 CreatePullPointSubscriptionResponse

When the device accepts the Subscription, it returns the `http://160.10.64.10/Subscription?Idx=0` URI which represents the Endpoint of this Subscription. Additionally, the client is informed about the `CurrentTime` of the device and the `TerminationTime` of the created Subscription.

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/CreatePullPointSubscriptionResponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:CreatePullPointSubscriptionResponse>
      <tet:SubscriptionReference>
        <wsa:Address>
          http://160.10.64.10/Subscription?Idx=0
        </wsa:Address>
      </tet:SubscriptionReference>
      <wsnt:CurrentTime>
        2008-10-09T13:52:59
      </wsnt:CurrentTime>
      <wsnt:TerminationTime>
        2008-10-09T13:53:59
      </wsnt:TerminationTime>
    </tet:CreatePullPointSubscriptionResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.10.5 PullMessagesRequest

The client sends a `PullMessagesRequest` to the Endpoint given in the `CreatePullPointSubscriptionResponse` to get Notifications corresponding to a certain Subscription. The following sample request contains a Timeout of five (5) seconds and limits the total number of messages in the response to two (2).

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1" >
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/PullPointSubscription/PullMessagesRequest
    </wsa:Action>

```

```
<wsa:To>http://160.10.64.10/Subscription?Idx=0</wsa:To>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
  <tet:PullMessages>
    <tet:Timeout>
      PT5S
    </tet:Timeout>
    <tet:MessageLimit>
      2
    </tet:MessageLimit>
  </tet:PullMessages>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.6 PullMessagesResponse

The following PullMessageResponse contains two notifications which match the subscription. The Response informs the client that two objects have crossed lines corresponding to rules “MyImportantFence1” and “MyImportantFence2”.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1"
  xmlns:tns1="http://www.onvif.org/ver10/topics"
  xmlns:tt="http://www.onvif.org/ver10/schema">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/PullPointSubscription/PullMessagesResponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:PullMessagesResponse>
      <tet:CurrentTime>
        2008-10-10T12:24:58
      </tet:CurrentTime>
      <tet:TerminationTime>
        2008-10-10T12:25:58
      </tet:TerminationTime>
      <wsnt:NotificationMessage>
        <wsnt:Topic>
          Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
            tns1:RuleEngine/LineDetector/Crossed
          </wsnt:Topic>
          <wsnt:Message>
            <tt:Message UtcTime="2008-10-10T12:24:57.321">
              <tt:Source>
                <tt:SimpleItem Name="VideoSourceConfigurationToken"
                  Value="1"/>
                <tt:SimpleItem Name="VideoAnalyticsConfigurationToken"
                  Value="2"/>
                <tt:SimpleItem Value="MyImportantFence1" Name="Rule"/>
              </tt:Source>
              <tt:Data>
                <tt:SimpleItem Name="ObjectId" Value="15" />
              </tt:Data>
            </tt:Message>
          </wsnt:Message>
        </wsnt:NotificationMessage>
        <wsnt:NotificationMessage>
          <wsnt:Topic>
            Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
              tns1:RuleEngine/LineDetector/Crossed
            </wsnt:Topic>
            <wsnt:Message>
              <tt:Message UtcTime="2008-10-10T12:24:57.789">
```

```

<tt:Source>
  <tt:SimpleItem Name="VideoSourceConfigurationToken"
    Value="1"/>
  <tt:SimpleItem Name="VideoAnalyticsConfigurationToken"
    Value="2"/>
  <tt:SimpleItem Value="MyImportantFence2" Name="Rule"/>
</tt:Source>
<tt:Data>
  <tt:SimpleItem Name="ObjectId" Value="19"/>
</tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
</tet:PullMessagesResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.10.7 UnsubscribeRequest

A client has to terminate a subscription explicitly with an UnsubscribeRequest that the device can immediately free resources. The request is directed to the Subscription Endpoint returned in the CreatePullPointSubscriptionResponse.

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" >
  <SOAP-ENV:Header>
    <wsa:Action>
      http://docs.oasis-open.org/wsn/bw-
      2/SubscriptionManager/UnsubscribeRequest
    </wsa:Action>
    <wsa:To>http://160.10.64.10/Subscription?Idx=0</wsa:To>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <wsnt:Unsubscribe/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
UnsubscribeResponse
The Subscription Endpoint is no longer available once the device replies
with an UnsubscribeResponse.
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" >
  <SOAP-ENV:Header>
    <wsa:Action>
      http://docs.oasis-open.org/wsn/bw-
      2/SubscriptionManager/UnsubscribeResponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <wsnt:UnsubscribeResponse/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.11 Service specific fault codes

The event service does not define any service specific faults except those defined in [WS-BaseNotification].

16 PTZ control

The PTZ control provides operations used to perform device pan, tilt and zoom control. A device with pan, tilt and zoom capability shall support the PTZ control. Similarly, a device with either just zoom or pan and tilt capability shall support the PTZ control. The PTZ control is defined in C.9. The mandatory operations are indicated under the command descriptions.

The PTZ control covers a wide category of camera devices. A dome model PTZ device is assumed to be able to change the viewing direction of the camera independently from the zoom.

PTZ movement is controlled using a coordinate system model. A device lists the set of coordinate systems it supports. This specification provides a set of generic Coordinate Spaces that is applicable for any PTZ device. It is possible to define further coordinate systems which are more appropriate for specific dome hardware. The PTZ control is applicable to the following devices:

- dome or PTZ camera;
- network video encoder with dome or PTZ camera connected via external serial port;
- fixed megapixel camera with digital PTZ;
- fixed camera with zoom.

The PTZ *control structure* consists of three major blocks:

- *PTZ Node* – a low-level PTZ entity that maps to the PTZ device and specifies its capabilities;
- *PTZ Configuration* – PTZ configuration of default coordinate systems and default speeds for a specific PTZ Node;
- *PTZ Control Operation* – move, preset, and auxiliary operations.

The PTZ control is connected to a media profile by including the PTZ configuration into the profile (see 4.8.2) and all PTZ control operations are done by referring to a particular media profile.

The PTZ control does not provide operations to create or manipulate PTZ Nodes. For each available PTZ Node, the device shall provide at least one PTZ Configuration assigned to this PTZ Node. This PTZ Configuration then can be added to Media Profiles, which are used to control the dome. Each media profile contains no more than *one* PTZ configuration. The PTZ Configuration and the VideoSourceConfiguration belong together in the Media Profile because the VideoSourceConfiguration refers to the Camera which is controlled by the PTZ Configuration.

A PTZ-capable device shall provide at least one *ready-to-use-profile* including a PTZConfiguration that covers the most basic settings and a corresponding VideoSourceConfiguration as soon as the underlying PTZ device is ready to operate.

16.1 PTZ Model

The PTZ Model groups the possible movements of the PTZ unit into a Pan/Tilt component and into a Zoom component. To steer the PTZ unit, the service provides absolute move, relative move and continuous move operations. Different coordinate systems and units are used to feed these operations.

The PTZ service provides an AbsoluteMove operation to move the PTZ device to an absolute position. The service expects the absolute position as an argument referencing an absolute coordinate system. The speed of the Pan/Tilt movement and the Zoom movement can be specified optionally. Speed values are positive scalars and do not contain any directional information. It is not possible to specify speeds for Pan and Tilt separately without knowledge about the current position. This approach to specifying a desired position generally produces a non-smooth and non-intuitive action.

A RelativeMove operation is introduced by the PTZ service in order to steer the dome relative to the current position, but without the need to know the current position. The operation expects a positional translation as an argument referencing a relative coordinate system. This standard distinguishes between relative and absolute coordinate systems, since there are

cases where no absolute coordinate system exists for a well-defined relative coordinate system. An optional speed argument can be added to the RelativeMove operation with the same meaning as for the AbsoluteMove operation.

Finally, the PTZ device can be moved continuously via the ContinuousMove command in a certain direction with a certain speed. Thereby, a velocity vector represents both, the direction and the speed information. The latter is expressed by the length of the vector.

The Pan/Tilt and Zoom coordinates can be uniquely specified by augmenting the coordinates with appropriate Space URIs. A Space URI uniquely represents the underlying coordinate system. Subclause 0 defines a standard set of coordinate systems. A PTZ Node shall implement these coordinate systems if the corresponding type of movement is supported by the PTZ Node. In many cases, the Pan/Tilt position is represented by pan and tilt angles in a spherical coordinate system. A digital PTZ, operating on a fixed megapixel camera, may express the camera's viewing direction by a pixel position on a static projection plane. Therefore, different coordinate systems are needed in this case in order to capture the physical or virtual movements of the PTZ device. These and other additional coordinate systems are defined in a separate document [ONVIF PTZ]. Optionally, the PTZ Node may define its own device specific coordinate systems to enable NVCs to take advantage of the specific properties of this PTZ Node.

The PTZ Node description retrieved via the GetNode or GetNodes operation contains all coordinate systems supported by a specific PTZ Node. Each coordinate system belongs to one of the following groups:

- AbsolutePanTiltPositionSpace;
- RelativePanTiltTranslationSpace;
- ContinuousPanTiltVelocitySpace;
- PanTiltSpeedSpace;
- AbsoluteZoomPositionSpace;
- RelativeZoomTranslationSpace;
- ContinuousZoomVelocitySpace;
- ZoomSpeedSpace.

If the PTZ Node does not support the coordinate systems of a certain group, the corresponding move operation will not be available for this PTZ Node. For instance, if the list does not contain an AbsolutePanTiltPositionSpace, the AbsoluteMove operation shall fail when an absolute Pan/Tilt position is specified. The corresponding command section describes those spaces that are required for a specific move command.

16.2 PTZ Node

A PTZ-capable device can have multiple PTZ Nodes. The PTZ Nodes may represent mechanical PTZ drivers, uploaded PTZ drivers or digital PTZ drivers. PTZ Nodes are the lowest level entities in the PTZ control API and reflect the supported PTZ capabilities. The PTZ Node is referenced either by its name or by its reference token. The PTZ Service does not provide operations to create or manipulate PTZ Nodes.

The following properties shall be provided for all PTZ Nodes:

- Token – a unique identifier that is used to reference PTZ Nodes;
- Name – a name given by the installer;
- SupportedPTZSpaces – a list of Coordinate Systems available for the PTZ Node. For each Coordinate System, the PTZ Node shall specify its allowed range;
- MaximumNumberOfPresets – all preset operations shall be available for this PTZ Node if one preset is supported;

- HomeSupported – a boolean operator specifying the availability of a home position. If set to true, the Home Position Operations shall be available for this PTZ Node;
- AuxiliaryCommands – a list of supported Auxiliary commands. If the list is not empty, the Auxiliary Operations shall be available for this PTZ Node.

16.2.1 GetNodes

A PTZ-capable device shall implement this operation and return all PTZ Nodes available on the device (see Table 214).

Table 214 – GetNodes command

GetNodes		Request-Response
Message name	Description	
GetNodesRequest	<i>This is an empty message.</i>	
GetNodesResponse	<i>The response message contains a list of the existing PTZ Nodes on the device.</i> tt:PTZNode PTZNode[0][unbounded]	
Fault codes	Description	
env:Receiver	<i>PTZ is not supported by the device.</i>	
ter:ActionNotSupported		
ter:PTZNotSupported		

16.2.2 GetNode

A PTZ-capable device shall implement the GetNode operation and return the properties of the requested PTZ Node, if it exists (see Table 215). Otherwise, the device shall respond with an appropriate Fault message.

Table 215 – GetNode command

GetNode		Request-Response
Message name	Description	
GetNodeRequest	<i>This message contains a reference to the requested PTZNode</i> tt:ReferenceToken NodeToken[1][1]	
GetNodeResponse	<i>The PTZNode response message contains the requested PTZNode.</i> tt:PTZNode PTZNode[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoEntity	<i>No such PTZNode on the device</i>	
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.3 PTZ configuration

The PTZConfiguration contains a reference to the PTZ Node in which it belongs. This reference cannot be changed by an NVC.

The following elements are part of the PTZ Configuration:

PTZNodeToken – A mandatory reference to the PTZ Node that the PTZ Configuration belongs to.

- **DefaultAbsolutePanTiltPositionSpace** – if the PTZ Node supports absolute Pan/Tilt movements, it shall specify one Absolute Pan/Tilt Position Space as default;
- **DefaultRelativePanTiltTranslationSpace** – if the PTZ Node supports relative Pan/Tilt movements, it shall specify one RelativePan/Tilt Translation Space as default;
- **DefaultContinuousPanTiltVelocitySpace** – if the PTZ Node supports continuous Pan/Tilt movements, it shall specify one Continuous Pan/Tilt Velocity Space as default;
- **DefaultPanTiltSpeedSpace** – if the PTZ Node supports absolute or relative movements, it shall specify one Pan/Tilt Speed Space as default;
- **DefaultAbsoluteZoomPositionSpace** – if the PTZ Node supports absolute zoom movements, it shall specify one Absolute Zoom Position Space as default;
- **DefaultRelativeZoomTranslationSpace** – if the PTZ Node supports relative zoom movements, it shall specify one Relative Zoom Translation Space as default;
- **DefaultContinuousZoomVelocitySpace** – if the PTZ Node supports continuous zoom movements, it shall specify one Continuous Zoom Velocity Space as default;
- **DefaultZoomSpeedSpace** – if the PTZ Node supports either absolute or relative movements, it shall specify one Zoom Speed Space as default;

- DefaultPTZSpeed – if the PTZ Node supports absolute or relative PTZ movements, it shall specify corresponding default Pan/Tilt and Zoom speeds;
- DefaultPTZTimeout – if the PTZ Node supports continuous movements, it shall specify a default timeout, after which the movement stops;
- PanTiltLimits – the Pan/Tilt limits element should be present for a PTZ Node that supports an absolute Pan/Tilt. If the element is present it signals the support for configurable Pan/Tilt limits. If limits are enabled, the Pan/Tilt movements shall always stay within the specified range. The Pan/Tilt limits are disabled by setting the limits to –INF or +INF;
- ZoomLimits – the Zoom limits element should be present for a PTZ Node that supports absolute zoom. If the element is present it signals the supports for configurable Zoom limits. If limits are enabled the zoom movements shall always stay within the specified range. The Zoom limits are disabled by settings the limits to -INF and +INF

The default Position/Translation/Velocity Spaces are introduced to allow NVCs sending move requests without the need to specify a certain coordinate system. The default Speeds are introduced to control the speed of move requests (absolute, relative, preset), where no explicit speed has been set.

The allowed pan and tilt range for Pan/Tilt Limits is defined by a two-dimensional space range that is mapped to a specific Absolute Pan/Tilt Position Space. At least one Pan/Tilt Position Space is required by the PTZNode to support Pan/Tilt limits. The limits apply to all supported absolute, relative and continuous Pan/Tilt movements. The limits shall be checked within the coordinate system for which the limits have been specified. That means that even if movements are specified in a different coordinate system, the requested movements shall be transformed to the coordinate system of the limits where the limits can be checked. When a relative or continuous movements is specified, which would leave the specified limits, the PTZ unit has to move along the specified limits. The Zoom Limits have to be interpreted accordingly.

16.3.1 GetConfigurations

A PTZ-capable device shall return all available PTZConfigurations through the GetConfigurations operation (see Table 216).

Table 216 – GetConfigurations command

GetConfigurations		Request-Response
Message name	Description	
GetConfigurations	<i>This is an empty message.</i>	
GetConfigurationsResponse	<i>The response contains all existing PTZConfigurations on the device.</i> tt:PTZConfiguration PTZConfiguration [0][unbounded]	
Fault codes	Description	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.3.2 GetConfiguration

A PTZ-capable device shall return the requested PTZ Configuration, if it exists, through the GetConfiguration operation (see Table 217).

Table 217 – GetConfiguration command

GetConfiguration		Request-Response
Message name	Description	
GetConfigurationRequest	<i>This message contains a reference to the requested PTZConfiguration.</i> tt:ReferenceToken ConfigurationToken[1][1]	
GetConfigurationResponse	<i>The response contains the requested PTZConfiguration</i> tt:PTZConfiguration PTZConfiguration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration does not exist.</i>	
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported by the device.</i>	

16.3.3 GetConfigurationOptions

A PTZ-capable device shall implement the GetConfigurationOptions operation. It returns the list of supported coordinate systems including their range limitations. Therefore, the options MAY differ depending on whether the PTZ Configuration is assigned to a Profile containing a Video Source Configuration. In that case, the options may additionally contain coordinate systems referring to the image coordinate system described by the Video Source Configuration. Each listed coordinate system belongs to one of the groups listed in 16.1. If the PTZ Node supports continuous movements, it shall return a Timeout Range within which Timeouts are accepted by the PTZ Node (see Table 218).

Table 218 – GetConfigurationOptions command

GetConfigurationOptions		Request-Response
Message name	Description	
GetConfigurationOptions-Request	This message contains a token to a PTZ configuration. ConfigurationToken specifies an existing configuration that the options are intended for. tt:ReferenceToken ConfigurationToken [1][1]	
GetConfigurationOptions-Response	<i>This message contains the PTZ configuration options.</i> tt:PTZConfigurationOptions PTZConfigurationOptions[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration does not exist.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.3.4 SetConfiguration

A PTZ-capable device shall implement the SetConfiguration operation (see Table 219). The ForcePersistence flag indicates if the changes remain after reboot of the device.

Table 219 – SetConfiguration command

SetConfiguration		Request-Response
Message name	Description	
SetConfigurationRequest	<i>The PTZConfiguration element contains the modified PTZ configuration. The configuration shall exist in the device.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:PTZConfiguration PTZConfiguration[1][1] xs:boolean ForcePersistence[1][1]	
SetConfigurationResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The configuration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The configuration parameters are not possible to set.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The new settings conflict with other uses of the configuration.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.4 Move operations

This subclause describes three operations to move the PTZ unit absolutely, relatively or continuously. All operations require a *ProfileToken* referencing a Media Profile including a PTZConfiguration.

All Move commands are non-blocking, meaning they do not wait until the requested move operation has finished. The last move operation can be overwritten by sending another move request.

16.4.1 AbsoluteMove

If a PTZ Node supports absolute Pan/Tilt or absolute Zoom movements, it shall support the AbsoluteMove operation (see Table 220). The Position argument of this command specifies the absolute position to which the PTZ Unit moves. It splits into an optional Pan/Tilt element and an optional Zoom element. If the Pan/Tilt position is omitted, the current Pan/Tilt movement shall NOT be affected by this command. The same holds for the zoom position.

The spaces referenced within the Position shall be absolute position spaces supported by the PTZ Node. If the Space information is omitted, the corresponding default spaces of the PTZ configuration, a part of the specified Media Profile, is used. A device may support absolute Pan/Tilt movements, absolute Zoom movements or no absolute movements by providing only absolute position spaces for the supported cases.

An existing Speed argument overrides the DefaultSpeed of the corresponding PTZ configuration during movement to the requested position. If spaces are referenced within the Speed argument, they shall be Speed Spaces supported by the PTZ Node.

The operation shall fail if the requested absolute position is not reachable.

Table 220 – AbsoluteMove command

AbsoluteMove		Request-Response
Message name	Description	
AbsoluteMoveRequest	<i>This message contains a reference to the media profile, a Position vector specifying the absolute target position and an optional Speed.</i> tt:ReferenceToken ProfileToken [1][1] tt:PTZVector Position [1][1] tt:PTZSpeed Speed [0][1]	
AbsoluteMoveResponse	<i>This is an empty message</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	<i>A space is referenced in an argument which is not supported by the PTZ Node</i>	
env:Sender ter:InvalidArgVal ter:InvalidPosition	<i>The requested position is out of bounds.</i>	
env:Sender ter:InvalidArgVal ter:InvalidSpeed	<i>The requested speed is out of bounds.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.4.2 RelativeMove

If a PTZ Node supports relative Pan/Tilt or relative Zoom movements, then it shall support the RelativeMove operation (see Table 221). The Translation argument of this operation specifies the difference from the current position to the position to which the PTZ device is instructed to move. The operation is split into an optional Pan/Tilt element and an optional Zoom element. If the Pan/Tilt element is omitted, the current Pan/Tilt movement shall NOT be affected by this command. The same holds for the zoom element.

The spaces referenced within the Translation element shall be Translation spaces supported by the PTZ Node. If the Space information is omitted for the Translation argument, the

corresponding default spaces of the PTZ configuration, which is part of the specified Media Profile, is used. A device may support relative Pan/Tilt movements, relative Zoom movements or no relative movements by providing only translation spaces for the supported cases.

An existing Speed argument overrides the DefaultSpeed of the corresponding PTZ configuration during movement by the requested translation. If spaces are referenced within the Speed argument, they shall be Speed Spaces supported by the PTZ Node.

The command can be used to stop the PTZ Unit at its current position by sending zero values for Pan/Tilt and Zoom. Stopping shall have the very same effect independent of the relative space referenced.

If the requested translation leads to an absolute position which cannot be reached, the PTZ Node shall move to a reachable position along the border of valid positions.

Withdrawing
IECNORM.COM: Click to view the full PDF of IEC 62676-2-3:2013

Table 221 – RelativeMove command

RelativeMove		Request-Response
Message name	Description	
RelativeMoveRequest	<i>This message contains a reference to the media profile, a positional Translation relative to the current position and an optional Speed parameter.</i> tt:ReferenceToken ProfileToken [1][1] tt:PTZVector Translation [1][1] tt:PTZSpeed Speed [0][1]	
RelativeMoveResponse	<i>This is an empty message</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	<i>A space is referenced in an argument which is not supported by the PTZ Node</i>	
env:Sender ter:InvalidArgVal ter:InvalidTranslation	<i>The requested translation is out of bounds.</i>	
env:Sender ter:InvalidArgVal ter:InvalidSpeed	<i>The requested speed is out of bounds.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.4.3 ContinuousMove

A PTZ-capable device shall support continuous movements (see Table 222). The Velocity argument of this command specifies a signed speed value for the Pan, Tilt and Zoom. The combined Pan/Tilt element is optional and the Zoom element itself is optional. If the Pan/Tilt element is omitted, the current Pan/Tilt movement shall NOT be affected by this command. The same holds for the Zoom element. The spaces referenced within the Velocity element shall be Velocity spaces supported by the PTZ Node. If the Space information is omitted for the Velocity argument, the corresponding default spaces of the PTZ configuration belonging to the specified Media Profile is used. A device MAY support continuous Pan/Tilt movements and/or continuous Zoom movements by providing only velocity spaces for the supported cases.

An existing Timeout argument overrides the DefaultPTZTimeout parameter of the corresponding PTZ configuration for this Move operation. The Timeout parameter specifies how long the PTZ Node continues to move.

The command can be used to stop the PTZ device at its current position by sending zero values for the Pan/Tilt and Zoom parameters. Stopping shall have the same effect independent of the velocity space referenced. This command has the same effect on a continuous move as the stop command specified in 16.4.4.

If the requested velocity leads to absolute positions which cannot be reached, the PTZ Node shall move to a reachable position along the border of its range. A typical application of the Continuous Move operation is controlling PTZ via joystick.

IECNORM.COM: Click to view the full PDF of IEC 62676-2-3:2013

Table 222 – ContinuousMove command

ContinuousMove		Request-Response
Message name	Description	
ContinuousMoveRequest	<i>This message contains a reference to the media profile, a Velocity vector specifying the velocity of pan, tilt and zoom, and an optional Timeout parameter.</i> tt:ReferenceToken ProfileToken [1][1] tt:PTZSpeed Velocity [1][1] xs:duration Timeout [0][1]	
ContinuousMoveResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	<i>A space is referenced in an argument which is not supported by the PTZ Node.</i>	
env:Sender ter:InvalidArgVal ter:TimeoutNotSupported	<i>The specified timeout argument is not within the supported timeout range.</i>	
env:Sender ter:InvalidArgVal ter:InvalidVelocity	<i>The requested velocity is out of bounds.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.4.4 Stop

A PTZ-capable device shall support the Stop operation (see Table 223). If no Stop filter arguments are present, this command stops all ongoing pan, tilt and zoom movements. The Stop operation can be filtered to stop a specific movement by setting the corresponding stop argument.

Table 223 – Stop (PTZ) command

Stop	Request-Response
Message name	Description
StopRequest	<i>This message contains a reference to the MediaProfile and parameters that indicate what should be stopped.</i> tt:ReferenceToken ProfileToken[1][1] xs:boolean PanTilt [0][1] xs:boolean Zoom0[1]
StopResponse	<i>This is an empty message.</i>
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>

16.4.5 GetStatus

A PTZ-capable device shall be able to report its PTZ status through the GetStatus command (see Table 224). The PTZ Status contains the following information:

- Position (optional) – specifies the absolute position of the PTZ unit together with the Space references. The default absolute spaces of the corresponding PTZ configuration shall be referenced within the Position element;
- MoveStatus (optional) – indicates if the Pan/Tilt/Zoom device unit is currently moving, idle or in an unknown state;
- Error (optional) – states a current PTZ error;
- UTC Time – specifies the UTC time when this status was generated.

Table 224 – GetStatus (PTZ) command

GetStatus		Request-Response
Message name	Description	
GetStatusRequest	<i>This message contains a reference to the media profile where the PTZStatus should be requested.</i> tt:ReferenceToken ProfileToken[1][1]	
GetStatusResponse	<i>This message contains the PTZStatus for the requested MediaProfile.</i> tt:PTZStatus PTZStatus[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env:Receiver ter:Action ter:NoStatus	<i>No PTZ status is available in the requested Media Profile.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.5 Preset operations

This subclause describes operations that manage the presets of a PTZ Node. These operations shall be implemented for PTZ Nodes supporting presets. All operations require a *ProfileToken* referencing a Media Profile including a PTZConfiguration.

16.5.1 SetPreset

The SetPreset command saves the *current* device position parameters so that the device can move to the saved preset position through the GotoPreset operation (see Table 225).

In order to create a new preset, the SetPresetRequest contains no PresetToken. If creation is successful, the Response contains the PresetToken which uniquely identifies the Preset. An existing Preset can be overwritten by specifying the PresetToken of the corresponding Preset. In both cases (overwriting or creation) an optional PresetName can be specified. The operation fails if the PTZ device is moving during the SetPreset operation.

The device MAY internally save additional states such as imaging properties in the PTZ Preset which then should be recalled in the GotoPreset operation.

Table 225 – SetPreset command

SetPreset		Request-Response
Message name	Description	
SetPresetRequest	<i>This message contains a reference to the MediaProfile and the requested name or token for the preset.</i> tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[0][1] xs:string PresetName[0][1]	
SetPresetResponse	<i>This message contains a reference to the Preset which has been set.</i> tt:ReferenceToken PresetToken[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:PresetExist	<i>The requested name already exist for another preset.</i>	
env:Sender ter:InvalidArgVal ter:InvalidPresetName	<i>The PresetName is either too long or contains invalid characters.</i>	
env:Receiver ter:Action ter:MovingPTZ	<i>Preset cannot be set while PTZ unit is moving.</i>	
env:Receiver ter:Action ter:TooManyPresets	<i>Maximum number of Presets reached.</i>	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoToken	<i>The requested preset token does not exist.</i>	

env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>

16.5.2 GetPresets

The GetPresets operation (see Table 226) returns the saved Presets consisting of the following elements:

- Token – a unique identifier to reference the Preset;
- Name – an optional mnemonic name;
- PTZ Position – an optional absolute position. If the PTZ Node supports absolute Pan/Tilt position spaces, the Pan/Tilt position shall be specified. If the PTZ Node supports absolute zoom position spaces, the zoom position shall be specified.

Table 226 – GetPresets command

GetPresets		Request-Response
Message name	Description	
GetPresetsRequest	<i>This message contains a reference to the MediaProfile where the operation should take place.</i> tt:ReferenceToken ProfileToken[1][1]	
GetPresetsResponse	<i>This message contains a list of presets which are available for the requested MediaProfile.</i> tt:PTZPreset Preset[0][unbounded]	
Fault codes		Description
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.5.3 GotoPreset

The GotoPreset operation recalls a previously set Preset (see Table 227). If the speed parameter is omitted, the default speed of the corresponding PTZ Configuration shall be used. The speed parameter can only be specified when Speed Spaces are available for the PTZ Node. The GotoPreset command is a non-blocking operation and can be interrupted by other move commands.

Table 227 – GotoPreset command

GotoPreset		Request-Response
Message name	Description	
GotoPresetRequest	<i>This message contains a reference to the MediaProfile where the move to the preset identified by its token should take place.</i> tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[1][1] tt:PTZSpeed Speed[0][1]	
GotoPresetResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token, ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoToken	<i>The requested preset token does not exist.</i>	
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	<i>A space is referenced in an argument which is not supported by the PTZ Node.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env:Sender ter:InvalidArgs ter:InvalidSpeed	<i>The requested speed is out of bounds.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.5.4 RemovePreset

The RemovePreset operation removes a previously set Preset (see Table 228).

Table 228 – RemovePreset command

RemovePreset		Request-Response
Message name	Description	
RemovePresetRequest	<i>This message contains a reference to the MediaProfile where the preset identified by the token should be removed.</i> tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[1][1]	
RemovePresetResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoToken	<i>The requested preset token does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.6 Home position operations

This subclause describes operations used to manage the Home Position of a PTZ Node. These operations shall be implemented for PTZ Nodes supporting home positions. All operations require a *ProfileToken* referencing a Media Profile including a PTZConfiguration.

The “home” position MAY be set by the SetHome operation or is a fix position of the PTZ unit.

16.6.1 GotoHomePosition

This operation moves the dome to its home position (see Table 229). If the speed parameter is omitted, the default speed of the corresponding PTZ Configuration shall be used. The speed parameter can only be specified when Speed Spaces are available for the PTZ Node. The command is non-blocking and can be interrupted by other move commands.

Table 229 – GotoHomePosition command

GotoHomePosition		Request-Response
Message name	Description	
GotoHomePositionRequest	<i>This message contains a reference to the MediaProfile where the operation should take place.</i> tt:ReferenceToken ProfileToken[1][1] tt:PTZSpeed Speed[0][1]	
GotoHomePositionResponse	<i>This is an empty message.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Receiver ter:Action ter:NoHomePosition	<i>No home position has been defined for this Profile.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.6.2 SetHomePosition

The SetHome operation saves the *current* position parameters as the home position, so that the GotoHome operation can request that the device move to the home position (see Table 230).

The SetHomePosition command shall return with a failure if the “home” position is fixed and cannot be overwritten. If the SetHomePosition is successful, it shall be possible to recall the Home Position with the GotoHomePosition command.

Table 230 – SetHomePosition command

SetHomePosition		Request-Response
Message name	Description	
SetHomePositionRequest	<i>This message contains a reference to the MediaProfile where the home position should be set.</i> tt:ReferenceToken ProfileToken[1][1]	
SetHomePositionResponse	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env:Receiver ter:Action	<i>The home position is fixed and cannot be overwritten.</i>	
ter:CannotOverwriteHome		
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.7 Auxiliary operations

16.7.1 General

This subclause describes operations to manage auxiliary commands of a PTZ Node, such as an Infrared (IR) lamp, a heater or a wiper.

These operations shall be implemented for PTZ nodes indicating auxiliary commands in the node properties. All operations require a *ProfileToken* referencing a Media Profile including a PTZConfiguration.

16.7.2 SendAuxiliaryCommand

This operation is used to call an auxiliary operation on the device (see Table 231). The supported commands can be retrieved via the PTZ Node properties. The AuxiliaryCommand should match the supported command listed in the PTZ Node; no other syntax is supported. If the PTZ Node lists the *irlampon* command, then the AuxiliaryCommand argument would be *irlampon*. The SendAuxiliaryCommand shall be implemented when the PTZ Node supports auxiliary commands.

Table 231 – Send Auxiliary command

SendAuxiliaryCommand		Request-Response
Message name	Description	
SendAuxiliaryCommandRequest	<i>This message contains a reference to the MediaProfile where the Auxiliary request should be done and the Auxiliary request data.</i> tt:ReferenceToken ProfileToken[1][1] tt:AuxiliaryData AuxiliaryData[1][1]	
SendAuxiliaryCommandResponse	<i>The response contains the auxiliary response.</i> tt:AuxiliaryData AuxiliaryResponse[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The requested profile token ProfileToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>The requested profile token does not reference a PTZ configuration.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ is not supported.</i>	

16.8 Predefined PTZ spaces

Spaces are used to specify absolute, relative and continuous movements. Whereas absolute movements require an absolute position, relative movements are specified by a position translation. Continuous movements require the specification of a velocity (relative movement over time). For these three cases, different coordinate systems are used describing the desired movement. The Generic Spaces do not absolutely specify the underlying PTZ Model, so that it can be applied to any PTZ hardware. Additional Spaces are defined in the document [ONVIF PTZ].

16.8.1 Absolute position spaces

16.8.1.1 Generic pan/tilt position space

The Generic Pan/Tilt Position Space shall be provided by every PTZ Node that supports absolute Pan/Tilt, since it does not relate to a specific physical range. Instead, the range should be defined as the full range of the PTZ unit normalized to the range -1 to 1 resulting in the following space description:

```
<tt:AbsolutePanTiltPositionSpace>
```

```

<tt:SpaceURI>

http://www.onvif.org/ver10/tptz/PanTiltSpaces/PositionGenericSpace

</tt:SpaceURI>

<tt:XRange>

  <tt:Min>-1.0</tt:Min>

  <tt:Max>1.0</tt:Max>

</tt:XRange>

<tt:YRange>

  <tt:Min>-1.0</tt:Min>

  <tt:Max>1.0</tt:Max>

</tt:YRange>

</tt:AbsolutePanTiltPositionSpace>

```

16.8.1.2 Generic zoom position space

The Generic Zoom Position Space shall be provided by every PTZ Node that supports absolute Zoom, since it does not relate to a specific physical range. Instead, the range should be defined as the full range of the Zoom normalized to the range 0 (wide) to 1 (tele). There is no assumption about how the generic zoom range is mapped to magnification, FOV or other physical zoom dimension. This results in the following space description:

```

<tt:AbsoluteZoomPositionSpace>

  <tt:SpaceURI>

    http://www.onvif.org/ver10/tptz/ZoomSpaces/PositionGenericSpace

  </tt:SpaceURI>

  <tt:XRange>

    <tt:Min>0.0</tt:Min>

    <tt:Max>1.0</tt:Max>

  </tt:XRange>

</tt:AbsoluteZoomPositionSpace>

```

16.8.2 Relative translation spaces

A Relative Pan/Tilt Translation Space moves the PTZ unit a certain translation in a certain direction without knowing the camera's current Pan/Tilt position.

16.8.2.1 Generic pan/tilt translation space

The Generic Pan/Tilt Translation Space shall be provided by every PTZ Node that supports relative Pan/Tilt, since it does not relate to a specific physical range. Instead, the range should be defined as the full positive and negative translation range of the PTZ unit normalized to the range -1 to 1, where positive translation would mean clockwise rotation or movement in right/up direction resulting in the following space description:

```
<tt:RelativePanTiltTranslationSpace>
```

```
  <tt:SpaceURI>
```

```
  http://www.onvif.org/ver10/tptz/PanTiltSpaces/TranslationGenericSpace
```

```
  </tt:SpaceURI>
```

```
  <tt:XRange>
```

```
    <tt:Min>-1.0</tt:Min>
```

```
    <tt:Max>1.0</tt:Max>
```

```
  </tt:XRange>
```

```
  <tt:YRange>
```

```
    <tt:Min>-1.0</tt:Min>
```

```
    <tt:Max>1.0</tt:Max>
```

```
  </tt:YRange>
```

```
</tt:RelativePanTiltTranslationSpace>
```

16.8.2.2 Generic zoom translation space

The Generic Zoom Translation Space shall be provided by every PTZ Node that supports relative Zoom, since it does not relate to a specific physical range. Instead, the corresponding absolute range should be defined as the full positive and negative translation range of the Zoom normalized to the range -1 to 1, where a positive translation maps to a movement in TELE direction. The translation is signed to indicate direction (negative is to wide, positive is to tele). There is no assumption about how the generic zoom range is mapped to magnification, FOV or other physical zoom dimension. This results in the following space description:

```
<tt:RelativeZoomTranslationSpace>
```

```
  <tt:SpaceURI>
```

```
  http://www.onvif.org/ver10/tptz/ZoomSpaces/TranslationGenericSpace
```

```
  </tt:SpaceURI>
```

```
  <tt:XRange>
```

```
    <tt:Min>-1.0</tt:Min>
```

```
    <tt:Max>1.0</tt:Max>
```

```
</tt:XRange>
```

```
</tt:RelativeZoomTranslationSpace>
```

16.8.3 Continuous velocity spaces

The Continuous Velocity Spaces are used to continuously move the PTZ unit in a certain direction.

16.8.3.1 Generic pan/tilt velocity space

The Generic Pan/Tilt Velocity Space shall be provided by every PTZ Node, since it does not relate to a specific physical range. Instead, the range should be defined as a range of the PTZ unit's speed normalized to the range -1 to 1, where a positive velocity would map to clockwise rotation or movement in the right/up direction. A signed speed can be independently specified for the pan and tilt component resulting in the following space description:

```
<tt:ContinuousPanTiltVelocitySpace>

  <tt:SpaceURI>

http://www.onvif.org/ver10/tptz/PanTiltSpaces/VelocityGenericSpace

  </tt:SpaceURI>

  <tt:XRange>

    <tt:Min>-1.0</tt:Min>

    <tt:Max>1.0</tt:Max>

  </tt:XRange>

  <tt:YRange>

    <tt:Min>-1.0</tt:Min>

    <tt:Max>1.0</tt:Max>

  </tt:YRange>

</tt:ContinuousPanTiltVelocitySpace>
```

16.8.3.2 Generic zoom velocity space

The Generic Zoom Velocity Space specifies a zoom factor velocity without knowing the underlying physical model. The range should be normalized from -1 to 1, where a positive velocity would map to TELE direction. A Generic Zoom Velocity Space description resembles the following:

```
<tt:ContinuousZoomVelocitySpace>

  <tt:SpaceURI>

http://www.onvif.org/ver10/tptz/ZoomSpaces/VelocityGenericSpace

  </tt:SpaceURI>
```



```

<tt:XRange>

  <tt:Min>-1.0</tt:Min>

  <tt:Max>1.0</tt:Max>

</tt:XRange>

</tt:ContinuousZoomVelocitySpace>

```

16.8.4 Speed spaces

The Speed Spaces specify the speed for a Pan/Tilt and Zoom movement when moving to an absolute position or to a relative translation. In contrast to the Velocity Spaces, Speed Spaces do not contain any directional information. The Speed of a combined Pan/Tilt movement is represented by a single non-negative scalar value.

16.8.4.1 Generic pan/tilt speed space

The Generic Pan/Tilt Speed Space shall be provided by every PTZ Node that supports configurable speed for Pan/Tilt, since it does not relate to a specific physical range. Instead, the range should be defined as the full range of the Speed range normalized to the range 0 (stopped) to 1 (full speed). This results in the following space description:

```

<tt:PanTiltSpeedSpace>

  <tt:SpaceURI>

http://www.onvif.org/ver10/tptz/PanTiltSpaces/GenericSpeedSpace

  </tt:SpaceURI>

  <tt:XRange>

    <tt:Min>0.0</tt:Min>

    <tt:Max>1.0</tt:Max>

  </tt:XRange>

</tt:PanTiltSpeedSpace>

```

16.8.4.2 Generic zoom speed space

The Generic Zoom Speed Space shall be provided by every PTZ Node that supports configurable speed for Zoom, since it does not relate to a specific physical range. Instead, the range should be defined as the full range of the Speed range normalized to the range 0 (stopped) to 1 (full speed). This results in the following space description:

```

<tt:ZoomSpeedSpace>

  <tt:SpaceURI>

http://www.onvif.org/ver10/tptz/ZoomSpaces/ZoomGenericSpeedSpace

  </tt:SpaceURI>

  <tt:XRange>

```

<tt:Min>0.0</tt:Min>

</tt:XRange>

</tt:ZoomSpeedSpace>

16.9 Service specific fault codes

Table 232 below lists the PTZ service specific fault codes. Each command can generate a generic fault, see Table 6.

The specific faults are defined as subcode of a generic fault, see 5.11.2.1. The parent generic sub code is the *subcode* at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 232 – PTZspecific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Receiver	ter:Action	Preset cannot be set	Preset cannot be set while the PTZ unit is moving.
	ter:MovingPTZ		
env:Receiver	ter:Action	Number of presets limit reached	Maximum number of Presets reached.
	ter:TooManyPresets		
env:Receiver	ter:ActionNotSupported	PTZ not supported	PTZ is not supported by the device.
	ter:PTZNotSupported		
env:Sender	ter:InvalidArgVal	Token already exist	The requested name or token already exist for another preset.
	ter:PresetExist		
env:Receiver	ter:Action	No PTZ status available	No PTZ status is available in the requested Media Profile.
	ter:NoStatus		
env:Receiver	ter:Action	Conflict when using new settings	The new settings result in an inconsistent configuration.
	ter:ConfigurationConflict		
env:Receiver	ter:Action	Home position cannot be overwritten	The home position is fixed and cannot be overwritten.
	ter:CannotOverwriteHome		
env:Sender	ter:InvalidArgVal	No such PTZ node	No such PTZ Node on the device
	ter:NoEntity		
env:Sender	ter:InvalidArgVal	No such configuration	No such configuration exist.
	ter:NoConfig		

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Sender	ter:InvalidArgVal	The paramters could not be set	The configuration parameters are not possible to set.
	ter:ConfigModify		
env:Sender	ter:InvalidArgVal	Destination out of bounds	The requested destination is out of bounds.
	ter:InvalidPosition		
env:Sender	ter:InvalidArgVal	Translation out of bounds	The requested translation is out of bounds.
	ter:InvalidTranslation		
env:Sender	ter:InvalidArgVal	Requested speed out of bounds	The requested speed is out of bounds.
	ter:InvalidSpeed		
env:Sender	ter:InvalidArgVal	Velocity out of bounds	The requested velocity is out of bounds.
	ter:InvalidVelocity		
env:Sender	ter:InvalidArgVal	PresetName too long	The PresetName is either too long or contains invalid characters.
	ter:InvalidPresetName		
env:Sender	ter:InvalidArgVal	Profile miss PTZ configuration	The requested profile token does not reference a PTZ configuration.
	ter:NoPTZProfile		
env:Sender	ter:InvalidArgVal	Profile token does not exist	The requested profile token ProfileToken does not exist.
	ter:NoProfile		
env:Sender	ter:InvalidArgVal	Timeout not supported	The specified timeout argument is not within the supported timeout range.
	ter:TimeoutNotSupported		
env:Sender	ter:InvalidArgVal	Token does not exist.	The requested preset token does not exist
	ter:NoToken		
env:Receiver	ter:Action	No HomePosition	No home position has been defined for this Profile.
	ter:NoHomePosition		
env:Sender	ter:InvalidArgVal	No such space	A space is referenced in an argument which is not

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
	ter:SpaceNotSupported		supported by the PTZ Node.

17 Video analytics

Subclause 4.12 gives a general overview of the ONVIF video analytics architecture. This subclause covers the following main areas of this architecture:

- analytics module interface;
- scene description;
- rules interface;
- event interface.

The event interface is handled through the Event service described in Clause 15. Subclause 17.1 introduces the XML-based scene description, which can be streamed as metadata to clients via RTP (see 12.1.2.1 for more details). The media service provides operations to manage complete analytics configurations consisting of both the rule engine and the analytics engine configuration (see Clause 11). The analytics service allows more fine-grained configuration of individual rules and individual analytics modules (see 17.2 and 0).

An device supporting analytics shall implement the Scene Description and Event Interface, as well as the Configuration of Analytics by the Media Service. If the device additionally supports a rule engine, responsible for analytics engine as defined by this standard, then it shall implement the Rules Analytics Modules Interface.

A complete video analytics configuration can be attached to a profile via the media service. A video analytics configuration becomes connected to a specific video source (see 0). The device shall ensure that a corresponding analytics engine starts operation when a client subscribes directly or indirectly for events produced by the analytics or rule engine or when a client requests the corresponding scene description stream.

17.1 Scene description interface

17.1.1 Overview

This standard defines the XML schema that shall be used to encode Scene Descriptions by a device. The scope of the Scene Description covers basic Scene Elements which can be displayed in a video overlay to the end-user as well as a framework for vendor-specific extensions. Annex B shows additional Scene Elements that may be used for processing vendor-specific rules.

The Video Analytics Engine is configured via Profiles of the MediaControl section. If Video Analytics are available in a Profile, a VideoSourceConfiguration and a VideoAnalyticsConfiguration shall be referenced in the Profile. The Video Analytics Engine then processes frames according to the referenced VideoSourceConfiguration.

17.1.2 Frame related content

The input of the Video Analytics Engine is images from a video source. The extracted scene elements are associated with the image from which they were extracted. An extracted scene is distinguished from the general description of the video source processed by the Video Analytics Engine (information such as video input line, video resolution, frame cropping, frame

rate etc.), the temporal frame association within the input stream, and the spatial positioning of elements within a frame.

The linkage between a Video Source and a Video Analytics Component is part of the Media Control, allowing the Video Analytics to run on a cropped video source with a reduced frame rate.

The temporal and spatial relation of scene elements with respect to the selected video source is discussed in 17.1.2.1 and 17.1.2.2. The appearance and behaviour of tracked objects is discussed in 17.1.3.1. Interactions between objects like splits and merges are described in 17.1.3.2.

A PTZ device can put information about the Pan, Tilt and Zoom at the beginning of a frame, allowing a client to estimate the 3D coordinates of scene elements. Next, the image Coordinate System can be adapted with an optional Transformation Node which is described in the next subsection. Finally, multiple Object Descriptions can be placed and their association can be specified within an ObjectTree Node. Below, the definitions are included for convenience⁶:

```
<xs:complexType name="Frame">
  <xs:sequence>
    <xs:element name="PTZStatus" type="tt:PTZStatus"
      minOccurs="0"/>
    <xs:element name="Transformation" type="tt:Transformation"
      minOccurs="0"/>
    <xs:element name="Object" type="tt:Object" minOccurs="0"
      maxOccurs="unbounded"/>
    <xs:element name="ObjectTree" type="tt:ObjectTree" minOccurs="0"/>
    ...
  </xs:sequence>
  <xs:attribute name="UtcTime" type="xs:dateTime" use="required"/>
  ...
</xs:complexType>

<xs:element name="Frame" type="tt:Frame">
```

Subclause 17.1.2.1 describes how frames processed by the Video Analytics Algorithm are referenced within the Video Analytics stream.

17.1.2.1 Temporal relation

Since multiple Scene Elements can be extracted from the same image, Scene Elements are listed below a Frame Node which establishes the link to a specific image from the video input. The Frame Node contains a MANDATORY UtcTime attribute. This UtcTime timestamp shall enable a client to map the Frame Node exactly to one video frame. For example, the RTP timestamp of the corresponding encoded video frame shall result in the same UTC timestamp after conversion. The synchronization between Video and Metadata streams is further described in the Real-time Viewing 12.1.2.2.

Example:

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  ...
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.521">
  ...
</tt:Frame>
```

⁶ Please note that the schema is included here for information only. [ONVIF Schema] contains the normative schema definition.

17.1.2.2 Spatial relation

Most Scene Elements refer to some part in an image from which information has been extracted. For instance, when tracking objects over time, their position within each frame shall be specified. These positions shall relate to a Coordinate System. The Default Coordinate System is shown in Figure 23. It maps onto the rectangle selected in the VideoSourceConfiguration of the corresponding Profile.

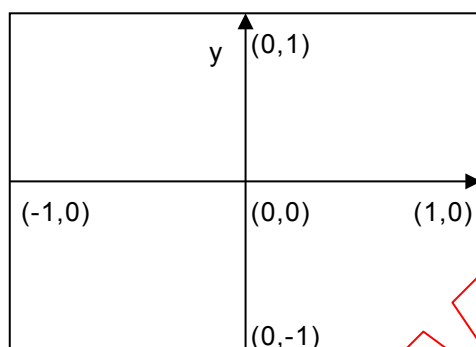


Figure 23 – Default frame coordinate system

This specification allows modification of the Coordinate System for individual nodes of the XML tree. As a result, each Frame Node starts with the Default Coordinate System. Each Child Node inherits the most recent Coordinate System of its parent. A Transformation Node modifies the most recent Coordinate System of its parent. Coordinate specifications are always related to the most recent Coordinate System of the Parent Node.

The specification defines transformation nodes for scaling and translation. The Scene Description contains placeholders where these Transformation Nodes are placed⁷.

```
<xs:complexType name="Transformation">
  <xs:sequence>
    <xs:element name="Translate" type="Vector" minOccurs="0"/>
    <xs:element name="Scale" type="Vector" minOccurs="0"/>
    ...
  </xs:sequence>
</xs:complexType>
```

It follows a mathematical description of coordinate systems and transformations. A coordinate system consists of a translational vector $t = \begin{pmatrix} t_x \\ t_y \end{pmatrix}$ and scaling $s = \begin{pmatrix} s_x \\ s_y \end{pmatrix}$. A point $p = \begin{pmatrix} p_x \\ p_y \end{pmatrix}$ given with respect to this coordinate system is transformed into the corresponding point $q = \begin{pmatrix} q_x \\ q_y \end{pmatrix}$ of the default coordinate system by the following formula: $\begin{pmatrix} q_x \\ q_y \end{pmatrix} = \begin{pmatrix} p_x \cdot s_x + t_x \\ p_y \cdot s_y + t_y \end{pmatrix}$. Similarly, a vector v given with respect to the coordinate system is transformed into the corresponding vector w of the default coordinate system by: $\begin{pmatrix} w_x \\ w_y \end{pmatrix} = \begin{pmatrix} v_x \cdot s_x \\ v_y \cdot s_y \end{pmatrix}$.

⁷ Please note that the schema is included here for *information only*. [ONVIF Schema] contains the normative schema definition.

A Transformation Node has an optional scaling vector $u = \begin{pmatrix} u_x \\ u_y \end{pmatrix}$ and an optional translational vector $v = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$. If the scaling is not specified, its default value $u = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ is assumed.

Similarly, the default value for the translation is $v = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$. The Transformation Node modifies the top-most Coordinate System in the following way:

$\begin{pmatrix} t'_x \\ t'_y \end{pmatrix} = \begin{pmatrix} v_x \times s_x + t_x \\ v_y \times s_y + t_y \end{pmatrix}$, $\begin{pmatrix} s'_x \\ s'_y \end{pmatrix} = \begin{pmatrix} u_x \times s_x \\ u_y \times s_y \end{pmatrix}$, where $\begin{pmatrix} t'_x \\ t'_y \end{pmatrix}$ and $\begin{pmatrix} s'_x \\ s'_y \end{pmatrix}$ replace the top-most Coordinate System.

For example, the coordinates of the scene description are given in a frame coordinate system, where the lower-left corner has coordinates (0,0) and the upper-right corner coordinates (320,240). The Frame Node resembles the following code where the scaling is set to the doubled reciprocal of the frame width and the frame height:

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-1.0"/>
    <tt:Scale x="0.00625" y="0.00834"/>
  </tt:Transformation>
  ...
</tt:Frame>
```

17.1.3 Scene elements

This subclause focuses on Scene Elements generated by object tracking algorithms and defines object handling and object shapes for them.

Frames where no objects have been detected can be skipped within the Scene Description to save bandwidth, as long as the last frame in the Scene Description is empty as well. It is RECOMMENDED that the device regularly sends the Scene Description even if it is empty, in order to indicate that the analytics engine is operational. The device shall send a Scene Description if a SynchronizationPoint is requested for the corresponding stream.

When the receiver of a Scene Description receives an empty frame, the receiver should assume that all subsequent frames are empty as well until the next non-empty frame is received. When the last received frame is non-empty, the receiver should assume that a description of the next processed frame will be transmitted.

17.1.3.1 Objects

Objects are identified via their Object ID. Features relating to one particular object are collected in an Object Node with the corresponding Object ID as an attribute. Associations of objects, like Object Renaming, Object Splits, Object Merges and Object Deletions are expressed in a separate ObjectTree node. An Object ID is implicitly created with the first appearance of the Object ID within an Object Node⁸.

```
<xs:complexType name="ObjectId">
  <xs:attribute name="ObjectId" type="xs:int"/>
</xs:complexType>
```

⁸ Please note that the schema is included here for *information only*. [ONVIF Schema] contains the normative schema definition.


```

<xs:complexType name="Object">
  <xs:complexContent>
    <xs:extension base="ObjectId">
      <xs:sequence>
        <xs:element name="Appearance" type="Appearance" minOccurs="0"/>
        <xs:element name="Behaviour" type="Behaviour" minOccurs="0"/>
        ...
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

The Object Node has two placeholders for Appearance and Behaviour information. The Appearance Node starts with an optional Transformation Node which can be used to change from a frame-centric coordinate system to an object-centric coordinate system. Next, the Shape of an Object can be specified. If an object is detected in a frame, the Shape information should be present in the Appearance description. The video analytics algorithm MAY add Object Nodes for currently not visible Objects, if it is able to infer information for this object otherwise. In such cases, the Shape description MAY be omitted.

Other object features like colour and object class can be added to the Appearance Node. This standard focuses on the Shape Descriptors (see 17.1.3.3). The definition of colour and object class can be found in B.1.

This standard defines two standard Behaviours for Objects. When an Object stops moving, it can be marked as either Removed or Idle. These behaviours shall be listed as Child Nodes of the Behaviour Node of an Object. The presence of a Removed or Idle Node does not automatically delete the corresponding Object ID, making it possible to reuse the same Object ID when the object starts moving again.

An object marked with the Removed Behaviour specifies the place from where the real object was removed. The marker should not be used as the Behaviour of the removed object. It is possible to detect the removal although the action of taking away the object was not detected.

Objects previously in motion can be marked as Idle to indicate that the object stopped moving. As long as such objects don't change, they will not be listed in the Scene Description anymore. When an Idle object appears again in the Scene Description, the Idle flag is removed automatically.

Example:

```

...
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-1.0"/>
    <tt:Scale x="0.003125" y="0.00416667"/>
  </tt:Transformation>
  <tt:Object ObjectId="12">
    <tt:Appearance>
      <tt:Shape>
        <tt:BoundingBox left="20.0" top="30.0" right="100.0"
bottom="80.0"/>
        <tt:CenterOfGravity x="60.0" y="50.0"/>
      </tt:Shape>
    </tt:Appearance>
  </tt:Object>
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.421">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-1.0"/>
    <tt:Scale x="0.003125" y="0.00416667"/>
  </tt:Transformation>
  <tt:Object ObjectId="12">
    <tt:Appearance>

```



```

        <tt:Shape>
            <tt:BoundingBox      left="20.0"      top="30.0"      right="100.0"
bottom="80.0"/>
            <tt:CenterOfGravity x="60.0" y="50.0"/>
        </tt:Shape>
    </tt:Appearance>
    <tt:Behaviour>
        <tt:Idle/>
    </tt:Behaviour>
</tt:Object>
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.521">
    <tt:Transformation>
        <tt:Translate x="-1.0" y="-1.0"/>
        <tt:Scale x="0.003125" y="0.00416667"/>
    </tt:Transformation>
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.621">
    <tt:Transformation>
        <tt:Translate x="-1.0" y="-1.0"/>
        <tt:Scale x="0.003125" y="0.00416667"/>
    </tt:Transformation>
    <tt:Object ObjectId="12">
        <tt:Appearance>
            <tt:Shape>
                <tt:BoundingBox      left="25.0"      top="30.0"      right="105.0"
bottom="80.0"/>
                <tt:CenterOfGravity x="65.0" y="50.0"/>
            </tt:Shape>
        </tt:Appearance>
    </tt:Object>
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.721">
    <tt:Transformation>
        <tt:Translate x="-1.0" y="-1.0"/>
        <tt:Scale x="0.003125" y="0.00416667"/>
    </tt:Transformation>
    <tt:Object ObjectId="19">
        <tt:Appearance>
            <tt:Shape>
                <tt:BoundingBox      left="20.0"      top="30.0"      right="100.0"
bottom="80.0"/>
                <tt:CenterOfGravity x="60.0" y="50.0"/>
            </tt:Shape>
        </tt:Appearance>
        <tt:Behaviour>
            <tt:Removed/>
        </tt:Behaviour>
    </tt:Object>
</tt:Frame>

```

17.1.3.2 Object tree

When two objects come too close to each other, such that the Video Analytics can no longer track them individually, an Object Merge should be signalled by adding a Merge Node to the ObjectTree Node of the Frame Node. The Merge Node contains a From Node listing the merging ObjectIds and a To Node containing the ObjectId. The merged Object is used in future frames as the tracking ID. If the Video Analytics Algorithm detects that one object is occluding the others and is able to track this object further, the occluding object should be put in the To Node.

The separation of objects is indicated by a Split Node. In this case, the From Node contains a single ObjectId representing the object which is split in the current frame. The objects separating from this split object are listed in the To Node. The ObjectId of the From Node can

reappear in the To Node, if this object did occlude the others and the Video Analytics Algorithm was able to track this object during the occlusion.

An Object does not need to be involved in a merge operation in order to be part of a split operation. For example, if an object is moving together with a person, and the person leaves the object somewhere, the object might be detected the first time by the Video Analytics when the person moves away from the object left behind. In such cases, the first appearance of the object can be combined with a Split operation.

When a merged object reappears as an Object Node in a later frame without a split indication, then this object is implicitly split. The Video Analytics Algorithm, however, could not determine where the split object came from.

A Video Analytics Algorithm can track and remember a limited number of objects. In order to indicate that a certain Object has been removed from the memory of the algorithm and therefore never appear again, the SceneDescription can contain a Delete Node within the ObjectTree Node.

If the Video Analytics Algorithm cannot decide during a Split operation the identity of an object, it should use a new ObjectId. When the algorithm has collected sufficient evidence for the identity of this object, it can change the ObjectId via the Rename operation. The Rename operation can also be used when an object reenters the scene and the true identity is discovered after some time.

A deleted ObjectId shall NOT be reused within the Scene Description until the ObjectId container has wrapped around.

Example:

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Object ObjectId="12">
    ...
  </tt:Object>
  <tt:Object ObjectId="17">
    ...
  </tt:Object>
</tt:Frame>

<tt:Frame UtcTime="2008-10-10T12:24:57.421">
  <tt:Object ObjectId="12">
    ...
  </tt:Object>
  <tt:ObjectTree>
    <tt:Merge>
      <tt:From ObjectId="12"/>
      <tt:From ObjectId="17"/>
      <tt:To ObjectId="12"/>
    </tt:Merge>
  </tt:ObjectTree>
</tt:Frame>

<tt:Frame UtcTime="2008-10-10T12:24:57.521">
  <tt:Object ObjectId="12">
    ...
  </tt:Object>
</tt:Frame>

<tt:Frame UtcTime="2008-10-10T12:24:57.621">
  <tt:Object ObjectId="12">
    ...
  </tt:Object>
  <tt:Object ObjectId="17">
    ...
  </tt:Object>
```

```

<tt:ObjectTree>
  <tt:Split>
    <tt:From ObjectId="12"/>
    <tt:To ObjectId="17"/>
    <tt:To ObjectId="12"/>
  </tt:Split>
</tt:ObjectTree>
</tt:Frame>

```

17.1.3.3 Shape descriptor

Shape information shall be placed below the optional Shape Node of in an Object Appearance Node. If present, the Shape Node holds information where the Object under consideration has been detected in the specified frame. A Shape Node shall at least contain two Nodes representing the Bounding Box and the Center Of Gravity of the detected object.

The coarse Bounding Box is further refined with additional Child Nodes, each representing a Shape Primitive. If multiple Shape Primitives are present, their union defines the Object's Shape. In this standard, a generic Polygon Descriptor is provided.

Polygons that describe the shape of an object shall be simple polygons defined by a list of Points.

Two consecutive Points (where the last point is connected with the first one) in the list define a line segment. The order of the Points shall be chosen such that the enclosed Object region can be found on the left-hand side all line segments. The polyline defined by the list of Points shall NOT be self-intersecting.

Example:

```

<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-3".0/>
    <tt:Scale x="0.003125" y="0.00416667"/>
  </tt:Transformation>
  <tt:Object ObjectId="12">
    <tt:Appearance>
      <tt:Shape>
        <tt:BoundingBox left="20.0" top="30.0" right="100.0"
bottom="80.0"/>
        <tt:CenterOfGravity x="60.0" y="50.0"/>
        <tt:Polygon>
          <tt:Point x="20.0" y="30.0"/>
          <tt:Point x="100.0" y="30.0"/>
          <tt:Point x="100.0" y="80.0"/>
          <tt:Point x="20.0" y="80.0"/>
        </tt:Polygon>
      </tt:Shape>
    </tt:Appearance>
  </tt:Object>
</tt:Frame>

```

17.2 Rule interface

The Video Analytics Configuration consists of two parts (see 0). The first part configures the Video Analytics Engine creating the SceneDescription. The second part configures the Rule Engine. For the second part, a XML structure is introduced in 17.2.1 to communicate the configuration of Rules. Subclause 17.2.2 specifies a language to describe the configuration of a specific Rule type. Subclause 17.2.3 defines two standard Rules that should be supported by a device implementing a Rule Engine. Subclause 17.2.4 introduces operations to manage rules. If the device supports a Rule Engine, it shall implement the complete Rule Interface.

17.2.1 Rule representation

The configuration of a rule has two required attributes: one specifies the Name and the other specifies the Type of the Rule. The different configuration parameters are listed below the Parameters element of the Rule element. Each Parameter is either a SimpleItem or an ElementItem (compare with message payload in Clause 15). The Name attribute of each Item shall be unique within the parameter list. SimpleItems have an additional Value attribute containing the value of the parameter. The value of ElementItems is given by the child element of the ElementItem. It is RECOMMENDED to represent as many parameters as possible by SimpleItems.

The following example shows a complete Video Analytics Configuration containing two Rules:

```
<tt:VideoAnalyticsConfiguration>
  <tt:AnalyticsEngineConfiguration>
    ...
  </tt:AnalyticsEngineConfiguration>
  <tt:RuleEngineConfiguration>
    <tt:Rule Name="MyLineDetector" Type="tt:LineDetector">
      <tt:Parameters>
        <tt:SimpleItem Name="Direction" Value="Any"/>
        <tt:ElementItem Name="Segments">
          <tt:Polyline>
            <tt:Point x="10.0" y="50.0"/>
            <tt:Point x="100.0" y="50.0"/>
          </tt:Polyline>
        </tt:ElementItem>
      </tt:Parameters>
    </tt:Rule>
    <tt:Rule Name="MyFieldDetector" Type="tt:FieldDetector">
      <tt:Parameters>
        <tt:ElementItem Name="Field">
          <tt:Polygon>
            <tt:Point x="10.0" y="50.0"/>
            <tt:Point x="100.0" y="50.0"/>
            <tt:Point x="100.0" y="150.0"/>
          </tt:Polygon>
        </tt:ElementItem>
      </tt:Parameters>
    </tt:Rule>
  </tt:RuleEngineConfiguration>
</tt:VideoAnalyticsConfiguration>
```

17.2.2 Rule description language

The description of a Rule contains the type information of all parameters belonging to a certain Rule Type and the description of the output produced by such a rule. The output of the Rule Engine is Events which can either be used in an Event Engine or be subscribed to by a client.

The parameters of a certain Rule Type are listed below the ParameterDescription element. All parameters are either Simple or ElementItems and can be described by either a SimpleItemDescription or an ElementItemDescription. Both ItemDescriptions contain a Name attribute to identify the parameter and a Type attribute to reference a specific XML schema type. In case of the SimpleItemDescription, the Type attribute shall reference a SimpleType schema definition. In case of the ElementItemDescription, the Type attribute shall reference a global element declaration of an XML schema.

The output produced by this Rule Type is described in multiple MessageDescription elements. Each MessageDescription contains a description of the Message Payload according to the Message Description Language detailed in Clause 15. Additionally, the MessageDescription shall contain a ParentTopic element naming the Topic a client has to subscribe to in order to receive this specific output. The Topic shall be specified as a Concrete Topic Expression.

Subclause 17.2.3 demonstrates the usage of the Rule Description Language on two standard rules. Below, the definitions are included for convenience⁹:

```
<xs:element name="RuleDescription" type="tt:ConfigDescription"/>

<xs:complexType name="ConfigDescription">
  <xs:sequence>
    <xs:element name="ParameterDescription"
      type="tt:ItemListDescription"/>
    <xs:element name="Messages" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="tt:MessageDescription">
            <xs:sequence>
              <xs:element name="ParentTopic" type="xs:string"/>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
    ...
  </xs:sequence>
  <xs:attribute name="Name" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="ItemListDescription">
  <xs:sequence>
    <xs:element name="SimpleItemDescription" minOccurs="0"
      maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="ElementItemDescription" minOccurs="0"
      maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

17.2.3 Standard rules

The following standard rules apply to static cameras. In case of a PTZ device, image-based rules should contain an additional `ElementItem`. The `ElementItem` identifies the position of the device for which the rule has been setup. The corresponding `ElementItemDescription` resembles the following:

```
<tt:ElementItemDescription Name="PTZStatus" Type="tt:PTZStatusType">
```

17.2.3.1 LineDetector

The `LineDetector` is defined by a non-intersecting simple polyline. If an `Object` crosses the polyline in the specified direction, the Rule Engine sends a `Crossed` event containing the name of the `LineDetector` and a reference to the object which has crossed the line. As directions, one can select between `Left`, `Right`, and `Any`, where directions `Left` and `Right` refer to the direction walking along the line from the first point to the second point and are the prohibited directions.

⁹ Please note that the schema is included here for *information only*. [ONVIF Schema] contains the normative schema definition.

The LineDetector resembles the following code using the Rule Description Language, detailed in the previous section:

```
<tt:RuleDescription Name="tt:LineDetector">
  <tt:Parameters>
    <tt:SimpleItemDescription Name="Direction" Type="tt:Direction"/>
    <tt:ElementItemDescription Name="Segments" Type="tt:Polyline"/>
  </tt:Parameters>
  <tt:MessageDescription>
    <tt:Source>
      <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
    </tt:Source>
    <tt:Data>
      <tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectId"/>
    </tt:Data>
    <tt:ParentTopic>tns1:RuleEngine/LineDetector/Crossed</tt:ParentTopic>
  </tt:MessageDescription>
</tt:RuleDescription>
```

The code above defines two parameters, Segments and Direction, and produces one Event attached to the topic tns1:RuleEngine/LineDetector/Crossed.

17.2.3.2 FieldDetector

A FieldDetector is defined by a simple non-intersecting polygon. The FieldDetector determines if each object in the scene inside or outside the polygon. This information is put into a property.

The FieldDetector resembles the following code, using the Rule Description Language detailed in the previous section:

```
<tt:RuleDescription Name="tt:FieldDetector">
  <tt:Parameters>
    <tt:ElementItemDescription Name="Field" Type="tt:Polygon"/>
  </tt:Parameters>
  <tt:MessageDescription IsProperty="true">
    <tt:Source>
      <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
    </tt:Source>
    <tt:Key>
      <tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectIdType"/>
    </tt:Key>
    <tt:Data>
      <tt:SimpleItemDescription Name="IsInside" Type="xs:boolean"/>
    </tt:Data>
    <tt:ParentTopic>
      tns1:RuleEngine/FieldDetector/ObjectsInside
    </tt:ParentTopic>
  </tt:MessageDescription>
</tt:RuleDescription>
```

From the Inside property, a client can derive the Entering and the Leaving parameters of the detector. A client can simulate Entering and Leaving events by adding a MessageContent Filter to the subscription, which lets only ObjectsInside messages pass, where the IsInside Item is set to true resp. false.

17.2.4 Operations on rules

If the device supports a Rule Engine as defined by ONVIF, then it shall implement the following operations to manage rules. The Create/Delete/Modify operations are atomic, meaning that either all modifications can be processed or the complete operation shall fail.

17.2.4.1 Get supported rules

The device shall indicate the rules it supports by implementing the subsequent operation (see Table 233). It returns a list of Rule Descriptions according to the Rule Description Language described in 17.2.2. Additionally, it contains a list of URLs that provide the location of the schema files. These schema files describe the types and elements used in the Rule Descriptions. If rule descriptions reference types or elements of the ONVIF schema file, the ONVIF schema file shall be explicitly listed.

Table 233 – GetSupportedRules command

GetSupportedRules		Request response
Message name	Description	
GetSupportedRulesRequest	The request message contains the VideoAnalyticsConfigurationToken for which the supported rules should be listed. tt:ReferenceToken ConfigurationToken [1][1]	
GetSupportedRulesResponse	The response contains the supported rules. tt: SupportedRules SupportedRules [1][1]	
Fault codes		Description
env:Sender		VideoAnalyticsConfiguration does not exist.
ter:InvalidArgVal		
ter:NoConfig		

17.2.4.2 Get rules

The following operation retrieves the currently installed Rules (see Table 234).

Table 234 – GetRules command

GetRules		Request-response
Message name	Description	
GetRulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken for which the rules should be reported.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetRulesResponse	<i>The response is a list of installed rules for the specified configuration.</i> tt:Config Rule [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The VideoAnalyticsConfiguration does not exist.</i>	

17.2.4.3 Create rules

The following operation adds Rules to a VideoAnalyticsConfiguration (see Table 235). If all rules cannot be created as requested, the device responds with a fault message.

Table 235 – CreateRules command

CreateRules		Request-response
Message name	Description	
CreateRulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken to which the listed Rules should be added.</i> tt:ReferenceToken ConfigurationToken [1][1] tt:Config Rule [1][unbounded]	
CreateRulesResponse	This is an empty message.	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The VideoAnalyticsConfiguration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:InvalidRule	<i>The suggested rules configuration is not valid on the device.</i>	
env:Sender ter:InvalidArgVal ter:RuleAlreadyExistent	<i>The same rule name exists already in the configuration.</i>	
enc:Receiver ter:Action ter:TooManyRules	<i>There is not enough space in the device to add the rules to the configuration.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The device cannot create the rules without creating a conflicting configuration.</i>	

17.2.4.4 Modify rules

The following operation modifies Multiple Rules (see Table 236). If all rules cannot be modified as requested, the device responds with a fault message.

Table 236 – ModifyRules command

ModifyRules		Request-response
Message name	Description	
ModifyRulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken for which the listed Rules should be modified.</i>	
	tt:ReferenceToken ConfigurationToken [1][1] tt:Config Rule[1][unbounded]	
ModifyRulesResponse	This is an empty message.	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The VideoAnalyticsConfiguration does not exist.</i>	
env:Sender ter:InvalidArgVal ter:InvalidRule	<i>The suggested rules configuration is not valid on the device.</i>	
env:Sender ter:InvalidArgs ter:RuleNotExistent	<i>The rule name or names do not exist.</i>	
enc:Receiver ter:Action ter:TooManyRules	<i>There is not enough space in the device to add the rules to the configuration.</i>	
env:Receiver ter:Action ter:ConflictingConfig	<i>The device cannot modify the rules without creating a conflicting configuration.</i>	

17.2.4.5 Delete rules

The following operation deletes Multiple Rules (see Table 237). If all rules cannot be deleted as requested, the device responds with a fault message.

Table 237 – DeleteRules command

DeleteRules		Request-response
Message name	Description	
DeleteRulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken from which the listed Rules should be removed.</i> tt:ReferenceToken ConfigurationToken [1][1] xs:string RuleName [1][unbounded]	
DeleteRulesResponse	<i>The response is an empty message.</i>	
Fault codes		Description
env:Sender ter:InvalidArgVal ter:NoConfig		<i>The VideoAnalyticsConfiguration does not exist.</i>
env:Receiver ter:Action ter:ConflictingConfig		<i>The device cannot delete the rules without creating a conflicting configuration.</i>
env:Sender ter:InvalidArgs ter:RuleNotExistent		<i>The rule name or names do not exist.</i>

17.3 Analytics modules interface

The Video Analytics Configuration consists of two parts (see 0). The first part configures the Video Analytics Engine creating the SceneDescription. The second part configures the Rule Engine. Subclause 17.3.1 defines an XML structure for the first part that communicates the configuration of Analytics Modules. Subclause 17.3.2 defines the language that describes the configuration of a specific Analytics Module. Subclause 17.3.3 defines the operations required by the Analytics Modules Interface. If the device supports an Analytics Engine as defined by ONVIF, it shall implement the complete Analytics Modules Interface.

17.3.1 Analytics module configuration

The Analytics Module Configuration is identical to the Rule Configuration, described in 17.2.1. The following example shows a possible configuration of a vendor-specific ObjectTracker. This tracker allows configuration of the minimum and maximum object size with respect to the processed frame geometry.

```
<tt:VideoAnalyticsConfig>
  <tt:AnalyticsEngineConfig>
    <tt:AnalyticsModule Name="MyObjectTracker" Type="nn:ObjectTracker">
      <tt:Parameters>
        <tt:SimpleItem Name="MinObjectWidth" Value="0.01"/>
      </tt:Parameters>
    </tt:AnalyticsModule>
  </tt:AnalyticsEngineConfig>
</tt:VideoAnalyticsConfig>
```

```

        <tt:SimpleItem Name="MinObjectHeight" Value="0.01"/>
        <tt:SimpleItem Name="MaxObjectWidth" Value="0.5"/>
        <tt:SimpleItem Name="MaxObjectHeight" Value="0.5"/>
    </tt:Parameters>
</tt:AnalyticsModule>
</tt:AnalyticsEngineConfig>
<tt:RuleEngineConfig>
    ...
</tt:RuleEngineConfig>
</tt:VideoAnalyticsConfig>

```

17.3.2 Analytics module description language

The Analytics Module reuses the Rule Description Language, described in 17.2.2. The following AnalyticsModuleDescription element replaces the RuleDescription element:

```

<xs:element name="AnalyticsModuleDescription"

    type="tt:ConfigDescription"/>

```

Similar to rules, Analytics Modules produce Events and shall be listed within the Analytics Module Description. The subsequent description corresponds to the example of the previous section. The example module produces a SceneTooCrowded Event when the scene becomes too complex for the module.

```

<tt:AnalyticsModuleDescription Name="nn:ObjectTracker">
    <tt:Parameters>
        <tt:SimpleItemDescription Name="MinObjectWidth" Type="xs:float"/>
        <tt:SimpleItemDescription Name="MinObjectHeight" Type="xs:float"/>
        <tt:SimpleItemDescription Name="MaxObjectWidth" Type="xs:float"/>
        <tt:SimpleItemDescription Name="MaxObjectHeight" Type="xs:float"/>
    </tt:Parameters>
    <tt:MessageDescription>
        <tt:Source>
            <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
                Type="tt:ReferenceToken"/>
            <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
                Type="tt:ReferenceToken"/>
            <tt:SimpleItemDescription Name="AnalyticsModule" Type="xs:string"/>
        </tt:Source>
        <tt:ParentTopic>
            tns1:VideoAnalytics/nn:ObjectTracker/SceneTooCrowded
        </tt:ParentTopic>
    </tt:MessageDescription>
</tt:AnalyticsModuleDescription>

```

17.3.3 Operations on analytics modules

If the device supports an analytics engine as defined by ONVIF, it shall support the subsequent operations to manage analytics modules. The Create/Delete/Modify operations shall be atomic, all modifications can be processed or the complete operation shall fail.

17.3.3.1 GetSupportedAnalyticsModules

The device indicates the analytics modules it supports by implementing the GetSupportedAnalyticsModule operation (see Table 238). It returns a list of Analytics Modules according to the Analytics Module Description Language, described in 17.2.2. Additionally, it contains a list of URLs that provide the location of the schema files. These schema files describe the types and elements used in the Analytics Module Descriptions. If the analytics module descriptions reference types or elements of the ONVIF schema file, the ONVIFschema file shall be explicitly listed.

Table 238 – GetSupportedAnalyticsModules command

GetSupportedAnalyticsModules		Request-response
Message name	Description	
GetSupportedAnalyticsModule sRequest	<i>The request message contains the VideoAnalyticsConfigurationToken for which the supported analytics modules should be listed.</i>	
	tt:ReferenceToken ConfigurationToken [1][1]	
GetSupportedAnalyticsModule sResponse	<i>The response contains the supported analytics modules.</i>	
	SupportedAnalyticsModules [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:NoConfig	<i>VideoAnalyticsConfiguration does not exist.</i>	

17.3.3.2 GetAnalytics Modules

The following operation retrieves the currently installed Analytics Modules (see Table 239).

Table 239 – GetAnalyticsModules command

GetAnalyticsModules		Request-response
Message name	Description	
GetAnalyticsModulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken for which the analytics modules should be reported.</i>	
	tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsModulesResponse	<i>The response is a list of installed analytics modules for the specified configuration.</i>	
	tt:Config AnalyticsModule [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:NoConfig	<i>The VideoAnalyticsConfiguration does not exist.</i>	

17.3.3.3 CreateAnalytics Modules

The following operation adds Analytics Modules to a VideoAnalyticsConfiguration (see Table 240). If all Analytics Modules cannot be created as requested, the device responds with a fault message.

Table 240 – CreateAnalyticsModules command.

CreateAnalyticsModules		Request-response
Message name	Description	
CreateAnalyticsModulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken to which the listed Analytics Modules should be added.</i> tt:ReferenceToken ConfigurationToken [1][1] tt:Config AnalyticsModule [1][unbounded]	
CreateAnalyticsModulesResponse	This is an empty message.	
Fault codes		Description
env:Sender ter:InvalidArgs ter:NoConfig		<i>The VideoAnalyticsConfiguration does not exist.</i>
env:Sender ter:InvalidArgs ter:NameAlreadyExistent		<i>The same analytics module name exists already in the configuration.</i>
enc:Receiver ter:Action ter:TooManyModules		<i>There is not enough space in the device to add the analytics modules to the configuration.</i>
env:Receiver ter:Action ter:ConfigurationConflict		<i>The device cannot create the analytics modules without creating a conflicting configuration.</i>
env:Sender ter:InvalidArgVal ter:InvalidModule		<i>The suggested module configuration is not valid on the device.</i>

17.3.3.4 ModifyAnalytics Modules

The following operation modifies multiple Analytics Modules (see Table 241). If all analytics modules cannot be modified as requested, the device respond with a fault message.

Table 241 – ModifyAnalyticsModules command

ModifyAnalyticsModules		Request-response
Message name	Description	
ModifyAnalyticsModulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken for which the listed analytics modules should be modified.</i> tt:ReferenceToken ConfigurationToken [1][1] tt:Config AnalyticsModule [1][unbounded]	
ModifyAnalyticsModulesResponse	The response is an empty message.	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:NoConfig	<i>The VideoAnalyticsConfiguration does not exist.</i>	
env:Sender ter:InvalidArgs ter:NameNotExistent	<i>The analytics module with the requested name does not exist.</i>	
enc:Receiver ter:Action ter:TooManyModules	<i>There is not enough space in the device to add the analytics modules to the configuration.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The device cannot modify the analytics modules without creating a conflicting configuration.</i>	
env:Sender ter:InvalidArgVal ter:InvalidModule	<i>The suggested module configuration is not valid on the device.</i>	

17.3.3.5 DeleteAnalytics Modules

The following operation deletes multiple Analytics Modules (see Table 242). If all analytics modules cannot be deleted as requested, the device responds with a fault message.

Table 242 – DeleteAnalyticsModules command

DeleteAnalyticsModules		Request-response
Message name	Description	
DeleteAnalyticsModulesRequest	<i>The request message specifies the VideoAnalyticsConfigurationToken from which the listed Analytics Modules should be removed.</i> tt:ReferenceToken ConfigurationToken [1][1] xs:string AnalyticsModuleName [1][unbounded]	
DeleteAnalyticsModulesResponse	The response is an empty message.	
Fault codes	Description	
env:Sender ter:InvalidArgs ter:NoConfig	<i>The VideoAnalyticsConfiguration does not exist.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>The device cannot delete the analytics modules without creating a conflicting configuration.</i>	
env:Sender ter:InvalidArgs ter:NameNotExistent	<i>The analytics module with the requested name does not exist.</i>	

17.4 Service-specific fault codes

Table 243 below lists the analytics service-specific fault codes. Each command can also generate a generic fault. Refer to Table 6.

The specific faults are defined as subcode of a generic fault, see 5.11.2.1. The parent generic subcode is the *subcode* at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 243 – The analytics-specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Receiver	ter:Action	No more space available.	There is not enough space in the device to add the rules to the configuration.
	ter:TooManyRules		
env:Receiver	ter:Action	No more space available.	There is not enough space in the device to add the analytics modules to the configuration.
	ter:TooManyModules		
env:Receiver	ter:Action	Conflict when using new settings	The new settings result in an inconsistent configuration.
	ter:ConfigurationConflict		
env:Sender	ter:InvalidArgVal	No configuration such	The requested VideoAnalyticsConfiguration does not exist.
	ter:NoConfig		
env:Sender	ter:InvalidArgVal	The rule is invalid.	The suggested rule configuration is not valid.
	ter:InvalidRule		
env:Sender	ter:InvalidArgVal	The module is invalid	The suggested analytics module configuration is not valid on the device.
	ter:InvalidModule		
env:Sender	ter:InvalidArgVal	The rule exists	The same rule name exists already in the configuration.
	ter:RuleAlreadyExistent		
env:Sender	ter:InvalidArgs	The rule does not exist	The rule name or names do not exist.
	ter:RuleNotExist		
env:Sender	ter:InvalidArgs	The name exists	The same analytics module name exists already in the configuration.
	ter:NameAlreadyExistent		
env:Sender	ter:InvalidArgs	The name does not exist	The analytics module with the requested name does not exist.
	ter:NameNotExist		

18 Analytics device

The analytics device service has to be used for stand alone analytics devices which perform evaluation processes on media streams or metadata enhanced media streams. It may be used for other entities as well. Evaluations may involve more than one media stream or metadata enhanced media stream at a time.

The analytics device service receives media streams or metadata enhanced media streams from live-generating or storing devices. It could comprise decoder capabilities if analysis is being performed on uncompressed data. A metadata enhanced media stream describes any stream containing media data and assigned metadata.

The Analytics Device Service is being used by clients to configure properties and functionality of a stand alone analytics device or other analytic operations on an entity providing this service.

Backchannel capabilities are not provided by stand alone analytics devices.

The Analytics Device Service relies on the receiver service for receiving the data from other devices through receiver objects identified by ReceiverTokens. Mechanisms have to be provided to assign different tracks in the received RTSP stream to the appropriate AnalyticsEngine.

Changes of e.g. camera parameters while analysis is being performed may influence results of the analysis. Therefore, input parameter changes have to be reflected in the AnalyticsEngineInput structure.

18.1 Overview

The central element in the configuration of an Analytics Device Service is the AnalyticsEngineControl. It comprises necessary tokens and descriptions for the service as well as the possibility of activation/deactivation for the particular AnalyticsEngineControl.

An AnalyticsEngine could be either a single algorithm or a complete application, e.g. lost baggage. Several parameter sets (VideoAnalyticsConfiguration) can exist in parallel for an AnalyticsEngine to allow for switching between e.g. day and night configurations. Additionally, a structure is provided (AnalyticsEngineInputInfo) to describe input configuration requirements for the particular AnalyticsEngine.

In order to enable adaptation of the AnalyticsEngine to different input data the description of the input being feed into the AnalyticsEngine has to be provided in the AnalyticsEngineInput element.

All structures have to exist at least once after boot of the NVA entity and could be filled in with default values where appropriate.

18.2 Analytics engine input

The AnalyticsEngineInput structure describes the video and metadata input provided to a particular AnalyticsEngine. If more than one input source is being used there has to be an AnalyticsEngineInput element for each of the sources.

SourceIdentification: identifies the source the input is coming from (e.g. identification of the camera cluster, the particular camera and the profile being used)

VideoSource: information about the video source, in particular about the compression parameters being used

MetadataInput: describes the source metadata provisioning to be used for analysis

18.2.1 GetAnalyticsEngineInputs

This operation lists all available analytics engine inputs for the device. The Analytics Device Service shall support the listing of available analytics engine inputs through the GetAnalyticsEngineInputs command (see Table 244).

Table 244 – GetAnalyticsEngineInputs command

GetAnalyticsEngineInputs		Request-Response
Message name	Description	
GetAnalyticsEngineInputsRequest	<i>This is an empty message.</i>	
GetAnalyticsEngineInputsResponse	<i>Contains a list of structures describing available AnalyticsEngineInputs.</i> tt:AnalyticsEngineInput Configuration [1][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

18.2.2 GetAnalyticsEngineInput

The GetAnalyticsEngineInput command fetches the input configuration if the analytics engine input configuration token is known. An Analytics Device Service shall support the listing of an analytics engine input configuration through the GetAnalyticsEngineInput command (see Table 245).

Table 245 – GetAnalyticsEngineInput command

GetAnalyticsEngineInput		Request-Response
Message name	Description	
GetAnalyticsEngineInputRequest	<i>Contains the token of an existing analytics engine input configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsEngineInputResponse	<i>Contains the requested analytics engine input configuration.</i> tt:AnalyticsEngineInput Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

18.2.3 SetAnalyticsEngineInput

This command changes the analytics engine input configuration. An Analytics Device Service shall support the modification of its analytics engine input configuration through this command (see Table 246).

Table 246 – SetAnalyticsEngineInput command

SetAnalyticsEngineInput		Request-Response
Message name	Description	
SetAnalyticsEngineInput - Request	<i>The Configuration shall be the new configuration.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AnalyticsEngineInput Configuration[1][1] xs:boolean ForcePersistence [1][1]	
SetAnalyticsEngineInputResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>The configuration is not possible to set</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

18.2.4 CreateAnalyticsEngineInputs

This command generates analytics engine input configurations. An Analytics Device Service shall support the generation of analytics engine input configurations through this command (see Table 247).

Table 247 – CreateAnalyticsEngineInputs command

CreateAnalyticsEngineInputs		Request-Response
Message name	Description	
CreateAnalyticsEngineInputsRequest	<i>The Configuration shall be the new configuration.</i> <i>The ForcePersistence element determines if the configuration shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AnalyticsEngineInput Configuration[1][unbounded] xs:boolean ForcePersistence [1][unbounded]	
CreateAnalyticsEngineInputsResponse	<i>Contains the configurations including generated tokens.</i> tt:AnalyticsEngineInput Configuration[1][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>The configurations are not possible to set</i>	
env:Receiver ter:Action ter:MaxAnalyticsEngineInput	<i>The maximum number of supported AnalyticsEngineInput objects has been reached.</i>	

18.2.5 DeleteAnalyticsEngineInputs

This command deletes analytics engine input configurations. An Analytics Device Service shall support the deletion of analytics engine input configurations through this command (see Table 248).

Table 248 – DeleteAnalyticsEngineInputs command

DeleteAnalyticsEngineInputs		Request-Response
Message name	Description	
DeleteAnalyticsEngineInputsRequest	<i>Contains ConfigurationTokens identifying the AnalyticsEngineInputs to be deleted.</i> tt:ReferenceToken ConfigurationToken [1][unbounded]	
DeleteAnalyticsEngineInputsResponse	<i>This message is empty</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineInput	<i>The requested AnalyticsEngineInput indicated with ConfigurationToken does not exist.</i>	
env:Sender ter:Action ter:CannotDeleteEngineInput	<i>It is not possible to delete a specified AnalyticsEngineInput.</i>	

18.3 Video analytics configuration

18.3.1 GetVideoAnalyticsConfiguration

The GetVideoAnalyticsConfiguration command fetches the video analytics configuration if the video analytics configuration token is known. An Analytics Device Service shall support the listing of video analytics configuration through the GetVideoAnalyticsConfiguration command (see Table 249).

All suitable video analytics configuration token can be found within available AnalyticsEngine configurations.

Table 249 – GetVideoAnalyticsConfiguration command

GetVideoAnalyticsConfiguration		Request-Response
Message name	Description	
GetVideoAnalyticsConfigurationRequest	<i>Contains the token of an existing video analytics configuration.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoAnalyticsConfigurationResponse	<i>Contains the requested video analytics configuration.</i> tt:VideoAnalyticsConfiguration Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

18.3.2 SetVideoAnalyticsConfiguration

This command changes the video analytics configuration. An Analytics Device Service shall support the modification of its analytics engine configuration through this command (see Table 250). If the SetVideoAnalyticsConfiguration command is being received by the Analytics Device Service the changes shall be applied also to the affected configuration if it is in active use.

Table 250 – SetVideoAnalyticsConfiguration command

SetVideoAnalyticsConfiguration		Request-Response
Message name	Description	
SetVideoAnalyticsConfiguration Request	–	<i>The Configuration shall be the new configuration.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:VideoAnalyticsConfiguration Configuration[1][1] xs:boolean ForcePersistence [1][1]
SetVideoAnalyticsConfigurationResponse		<i>This message is empty</i>
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>The configuration is not possible to set</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

18.4 Analytics engines

The structure returned by the commands defined herein contains a list of available VideoAnalyticsConfiguration for the particular AnalyticsEngine together with appropriate AnalyticsEngineInputInfo elements for each VideoAnalyticsConfiguration.

VideoAnalyticsConfiguration: description of configuration possibilities of the analytics engine

AnalyticsEngineInputInfo: information about input requirements of the analytics engine

18.4.1 GetAnalyticsEngines

This operation lists all available analytics engines for the device. The Analytics Device Service shall support the listing of available analytics engines through the GetAnalyticsEngines command (see Table 251).

Table 251 – GetAnalyticsEngines command

GetAnalyticsEngines		Request-Response
Message name	Description	
GetAnalyticsEnginesRequest	<i>This is an empty message.</i>	
GetAnalyticsEnginesResponse	<i>Contains a list of structures describing available AnalyticsEngines.</i> tt:AnalyticsEngine Configuration [1][unbounded]	
Fault codes	Description	
	<i>No command specific faults!</i>	

18.4.2 GetAnalyticsEngine

The GetAnalyticsEngine command fetches the analytics engine if the analytics engine token is known. An Analytics Device Service shall support the listing of an analytics engine configuration through the GetAnalyticsEngine command (see Table 252).

Table 252 – GetAnalyticsEngine command

GetAnalyticsEngine		Request-Response
Message name	Description	
GetAnalyticsEngineRequest	<i>Contains the token of an existing analytics engine.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsEngineResponse	<i>Contains the requested AnalyticsEngine configuration.</i> tt:AnalyticsEngine Configuration [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	

18.5 Analytics engine control

The AnalyticsEngineControl structure shall be used to exercise control through the commands defined in the following.

Name: friendly description

EngineToken: Token of the analytics engine (AnalyticsEngine) being controlled

EngineConfigToken: Token of the analytics engine configuration (VideoAnalyticsConfiguration) in effect

InputToken: Tokens of the input (AnalyticsEngineInput) configuration applied

ReceiverToken: Tokens of the receiver providing media input data. The order of ReceiverToken shall exactly match the order of InputToken.

Multicast: parameter for multicast used to configure and control multicast of the metadata stream

Subscription: Description of Topics the controlled engine is reacting on

Mode: indicating the actual status for the controlled analysis (shall be either "Idle" or "Active")

18.5.1 GetAnalyticsEngineControls

This operation lists all available analytics engine controls for the device. The Analytics Device Service shall support the listing of available analytics engine controls through the GetAnalyticsEngineControls command (see Table 253).

Table 253 – GetAnalyticsEngineControls command

GetAnalyticsEngineControls	Request-Response
Message name	Description
GetAnalyticsEngineControlsRequest	<i>This is an empty message.</i>
GetAnalyticsEngineControlsResponse	<i>Contains a list of structures describing available AnalyticsEngineControls.</i> tt:AnalyticsEngineControl AnalyticsEngineControls [1][unbounded]
Fault codes	Description
	<i>No command specific faults!</i>

18.5.2 GetAnalyticsEngineControl

The GetAnalyticsEngineControl command fetches the analytics engine control if the analytics engine control token is known. An Analytics Device Service shall support the listing of analytics engine control configuration through the GetAnalyticsEngineControl command (see Table 254).

Table 254 – GetAnalyticsEngineControl command

GetAnalyticsEngineControl		Request-Response
Message name	Description	
GetAnalyticsEngineControlRequest	<i>Contains the token of an existing AnalyticsEngineControl.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsEngineControlResponse	<i>Contains the requested AnalyticsEngineControl configuration.</i> tt:AnalyticsEngineControl Configuration [1][1]	
Fault codes	Description	
env:Sender	<i>The requested configuration indicated with ConfigurationToken does not exist.</i>	
ter:InvalidArgVal		
ter:NoConfig		

18.5.3 SetAnalyticsEngineControl

This command changes the AnalyticsEngineControl configuration. An Analytics Device Service shall support the modification of its analytics engine control configuration through this command (see Table 255).

Table 255 – SetAnalyticsEngineControl command

SetAnalyticsEngineControl		Request-Response
Message name	Description	
SetAnalyticsEngineControlRequest	<i>The Configuration shall be the new configuration.</i> <i>The ForcePersistence element determines if the configuration changes shall be stored and remain after reboot. If true, changes shall be persistent. If false, changes MAY revert to previous values after reboot.</i> tt:AnalyticsEngineControl Configuration[1][1] xs:boolean ForcePersistence [1][1]	
SetAnalyticsEngineControlResponse	This message is empty	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	The configuration is not possible to set	
env:Sender ter:InvalidArgVal ter:NoConfig	The requested configuration indicated with ConfigurationToken does not exist.	

18.5.4 CreateAnalyticsEngineControl

CreateAnalyticsEngineControl shall create a new control object. Mode shall be set to "idle". To change the mode to "active" the SetAnalyticsEngineControl command can be used. An Analytics Device Service shall support the creation of control objects through this command (see Table 256).

Table 256 – CreateAnalyticsEngineControl command

CreateAnalyticsEngineControl		Request-Response
Message name	Description	
CreateAnalyticsEngineControlRequest	<i>The Configuration shall be the new configuration.</i> tt:AnalyticsEngineControl Configuration[1][1]	
CreateAnalyticsEngineControlResponse	<i>Contains the configuration including the generated token.</i> tt:AnalyticsEngineControl Configuration[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:AnalyticsEngineControlExists	<i>An AnalyticsEngineControl with the token ConfigurationToken already exists.</i>	
env:Receiver ter:Action ter:MaxAnalyticsEngineControl	<i>The maximum number of supported AnalyticsEngineControl objects has been reached.</i>	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>The configuration is not possible to set</i>	

18.5.5 DeleteAnalyticsEngineControl

DeleteAnalyticsEngineControl shall delete a control object. An Analytics Device Service shall support the deletion of control objects through this command (see Table 257).

Table 257 – DeleteAnalyticsEngineControl command

DeleteAnalyticsEngineControl		Request-Response
Message name	Description	
DeleteAnalyticsEngineControlRequest	<i>Contains the ConfigurationToken of the AnalyticsEngineControl to be deleted.</i> tt:ReferenceToken ConfigurationToken [1][1]	
DeleteAnalyticsEngineControlResponse	<i>This message is empty.</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	<i>The requested AnalyticsEngineControl indicated with ConfigurationToken does not exist.</i>	
env:Sender ter:Action ter:CannotDeleteControl	<i>It is not possible to delete the specified AnalyticsEngineControl.</i>	

18.6 GetAnalyticsState

GetAnalyticsState returns status information of the referenced AnalyticsEngineControl object (see Table 258). The structure AnalyticsStateInformation is expandable. The expansion shall be used to convey additional state information about substructures. E.g. an AnalyticsEngine is composed of different analytics algorithms for which state information should be provided. The state element of AnalyticsStateInformation always holds an aggregated state of all substructures.

An Analytics Device Service shall support state information provisioning through this command.

ConfigurationToken shall be the identification of the AnalyticsEngineControl for which the state information is requested

State shall hold the aggregated state over all substructures of AnalyticsStateInformation. If state equals “Error”, the Error may be filled in with an implementation defined value. A device shall apply the following rules to compute aggregate state:

Idle	The state of all substructures is "Idle"
PartiallyActive	At least one of the substructures has state "Active", all other substructures have state "Idle".
Active	The state of all substructures is "Active"
Error	At least one of the substructures has state "Error"

Error, if present, shall hold an implementation defined string value that describes the error.

Table 258 – GetAnalyticsState

GetAnalyticsState		Request-Response
Message name	Description	
GetAnalyticsStateRequest	Contains the ConfigurationToken of the AnalyticsEngineControl for which to get the state. tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsStateResponse	The State shall hold the state of the AnalyticsEngineControl. tt:AnalyticsStateInformation State[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	The ConfigurationToken does not reference an existing AnalyticsEngineControl.	

18.7 Output streaming configuration

18.7.1 General

An Analytics Device Service provides a real-time streaming interface as specified in the clause "Real time streaming" by acting as an RTSP server. Instead of the token identifying the profile being used in a Media Profile, the token identifying the AnalyticsEngineControl will be used on an Analytics Device Service.

18.7.2 Request stream URI

This operation requests a URI that can be used to initiate a live stream using RTSP as the control protocol. The URI is valid only as it is specified in the response or until the Analytics Engine Control is reconfigured. The Analytics Device Service shall support the retrieval of a stream URI for a specific analytics engine control through the GetAnalyticsDeviceStreamUri command (see Table 259).

Table 259 – GetAnalyticsDeviceStreamUri command

GetAnalyticsDeviceStreamUri		Request-Response
Message name	Description	
GetAnalyticsDeviceStreamUriRequest	<i>The StreamSetup element contains two parts. StreamType defines if a unicast or multicast media stream is requested. Transport specifies a chain of transport protocols defining the tunneling of the media stream over different network protocols.</i> <i>The AnalyticsEngineControlToken element shall indicate the analytics engine control to use.</i> tt:StreamSetup StreamSetup [1][1] tt:ReferenceToken AnalyticsEngineControlToken [1][1]	
GetAnalyticsDeviceStreamUriResponse	<i>Contains the Uri to be used for requesting the media stream</i> xs:anyURI Uri [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	<i>The requested configuration indicated with AnalyticsEngineControlToken does not exist.</i>	
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	<i>Specification of StreamType or Transport part in StreamSetup is not supported.</i>	
env:Sender ter:OperationProhibited ter:StreamConflict	<i>Specification of StreamType or Transport part in StreamSetup causes conflict with other streams.</i>	

19 Recording control

19.1 General

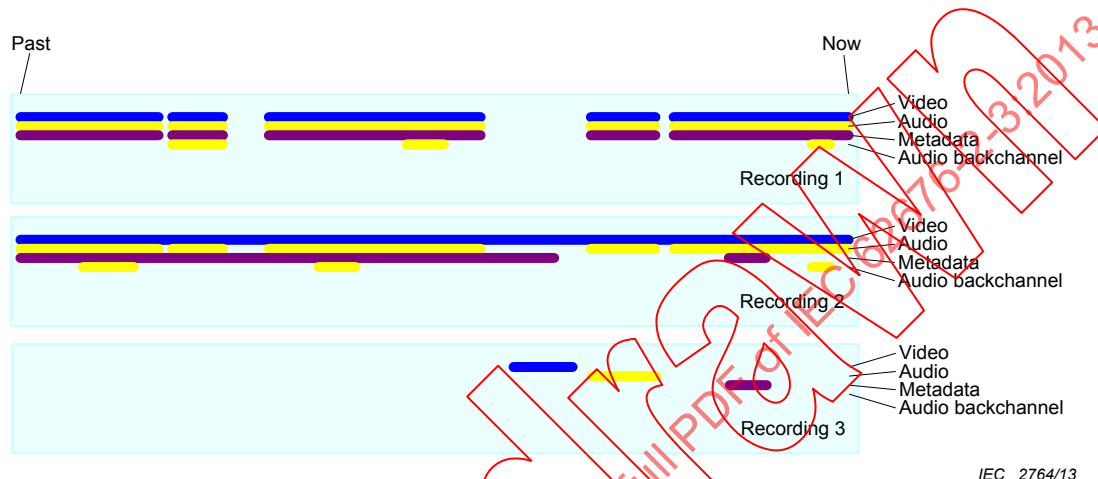
The recording service enables a client to manage recordings, and to configure the transfer of data from data sources to recordings. Managing recordings includes creation and deletion of recordings and tracks, as well as locking and unlocking ranges of recordings and deletion of recorded data.

Recording jobs transfer data from a recording source to a recording. A recording source can be a receiver object created with the receiver service, or it can be a media profile that

encodes data on a local device. The media profile could be used as a source on a camera with embedded storage.

The term *recording* is used in this standard to denote a container for a set of audio, video and metadata tracks. A recording could hold any number of tracks. A track is viewed as an infinite timeline that holds data at certain times.

Figure 24 shows three recordings, each with a video, a metadata and two audio tracks. Here second audio track is used for storing the audio backchannel.



IEC 2764/13

Figure 24 – Example of recordings and tracks

At a minimum, a recording shall be capable of holding three tracks, one for audio, one for video and one for metadata. Some implementations of the recording service may support multiple tracks of each type.

To save data to a recording, a client first creates a recording and ensures that the recording has the necessary tracks. Then the client creates a recording job that pulls data from one or more sources and stores the data to the tracks in the recording.

Clients may set up multiple recording jobs that all record into the same recording. If multiple recording jobs are active, the device uses a priority scheme to select between the tracks defined in the recording jobs. Clients may change the mode of recording jobs at any time, thereby providing means to implement features like alarm recording or manual recording.

The recording job relies on the receiver service for receiving the data from other devices through receiver objects identified by ReceiverTokens

For the cases where a client uses a receiver object with a single recording job, the recording service can auto create and auto delete receiver objects. Autocreation is signalled with the AutoCreateReceiver flag in the recording job configuration structure. Receiver objects created this way shall be automatically deleted when no recording job uses them anymore. A receiver object that is automatically created shall have all its fields set to empty values. The client should configure the receiver object after it has created the recording job.

The ONVIF view of recordings is a logical one which is independent of the way recordings are physically stored on disk. For instance, some camera implementations realise alarm recording by creating a distinct file on a FAT file system for each alarm that occurs. Although each file could be represented as a different ONVIF recording, the intent of the model in this standard is that all these files are aggregated into a single recording. By searching for the

“DataPresent” event with the FindEvents method of the search service, a client can locate the times at which video started to be recorded and where video stopped being recorded.

If Metadata is recorded, the metadata can also hold all the events generated by the data source (see Clause 15 on event handling and 11.10 for the MetadataConfiguration object). In addition, a device also conceptually record ONVIF defined historical events (see Recording Event Descriptions in the search service), this includes information like start and end of a recorded data range. A device may also conceptually record vendor specific historical events. Events generated by the device are not inserted in existing metadata tracks of recordings. The FindEvents method in the search service can find all the recorded events. Many device implementations will automatically delete the oldest recorded data from storage in order to free up space for new recordings. Locks provide a mechanism to allow a user to select ranges of data. A range of data that is locked does not get deleted automatically. Support for locks is reserved for future editions of the standard.

19.2 General requirements

All the objects created within the recording service shall be persistent – i.e. they shall survive a power cycle. Likewise, all the configuration data in the objects shall be persistent.

19.3 Data structures

19.3.1 RecordingConfiguration

The RecordingConfiguration structure shall be used to configure recordings through CreateRecordings and Get/SetRecordingConfiguration.

MaximumRetentionTime specifies the maximum time that data in the any track within the recording shall be stored. The device shall delete any data older than the maximum retention time. Such data shall not be accessible anymore. If the MaximumRetentionPeriod is set to 0, the device shall not limit the retention time of stored data, except by resource constraints. Whatever the value of MaximumRetentionTime, the device may automatically delete recordings to free up storage space for new recordings.

None of the other fields defined in this structure shall be used by the device. Instead, it simply stores this information, and it shall return it through the *GetRecordingConfiguration* and *GetRecordingInformation* (see 20.5) methods.

19.3.2 TrackConfiguration

The TrackConfiguration structure shall be used to configure tracks using CreateTrack and Get/SetTrackConfiguration

The TrackConfiguration contains the following fields:

The **TrackType** defines the data type of the track. It shall be equal to the strings “Video”, “Audio” or “Metadata”. The track shall only be able to hold data of that type.

None of the other fields defined in this structure shall be used by the device. Instead, it simply stores this information, and it shall return it through the *GetTrackConfiguration* and *GetRecordingInformation* (see 20.5) methods.

19.3.3 RecordingJobConfiguration

The RecordingJobConfiguration structure shall hold the configuration for a recording job. Figure 25 shows the RecordingJobConfiguration as an UML diagram.

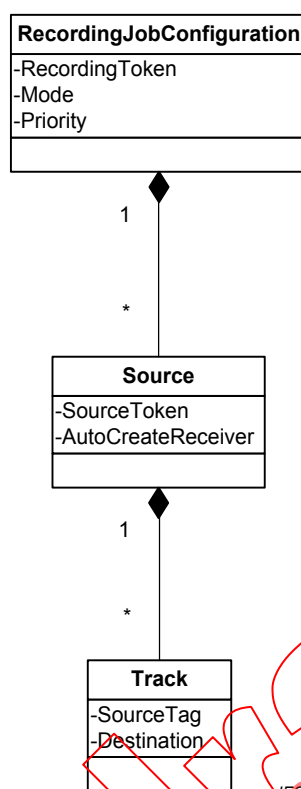


Figure 25 – Diagram of the RecordingJobConfiguration elements

The RecordingJobConfiguration holds the following fields:

RecordingToken: Identifies the recording to which this job shall store the received data.

Mode: The mode of the job. If it is idle, nothing shall happen. If it is active, the device shall try to obtain data from the receivers. A client shall use GetRecordingJobState to determine if data transfer is really taking place. The only valid values for Mode shall be “Idle” and “Active”.

Priority: This shall be a positive number. If there are multiple recording jobs that store data to the same track, the device will only store the data for the recording job with the highest priority. The priority is specified per recording job, but the device shall determine the priority of each track individually. If there are two recording jobs with the same priority, the device shall record the data corresponding to the recording job that was activated the latest.

The value 0 indicates the lowest priority. Higher values shall indicate a higher priority.

SourceToken: This field shall be a reference to the source of the data. The type of the source is determined by the attribute Type in the SourceToken structure. If Type is <http://www.onvif.org/ver10/schema/Receiver>, the token is a ReceiverReference. In this case the device shall receive the data over the network. If Type is <http://www.onvif.org/ver10/schema/Profile>, the token identifies a media profile, instructing the device to obtain data from a profile that exists on the local device.

If the **SourceToken** is omitted, **AutoCreateReceiver** shall be true.

AutoCreateReceiver: If this field is TRUE, and if the **SourceToken** is omitted, the device shall create a receiver object (through the receiver service) and assign the ReceiverReference to the **SourceToken** field. When retrieving the RecordingJobConfiguration from the device, the **AutoCreateReceiver** field shall never be present.

SourceTag: If the received RTSP stream contains multiple tracks of the same type, the **SourceTag** differentiates between those Tracks.

Destination: The destination is the tracktoken of the track to which the device shall store the received data.

19.4 CreateRecording

CreateRecording shall create a new recording (see Table 260). The new recording shall be created with one video, one audio and one metadata track.

This method is optional. It shall be available if the Recording/DynamicRecordings capability is TRUE.

Table 260 – CreateRecording command

CreateRecording	
Message name	Description
CreateRecordingRequest	Contains the initial configuration for the recording tt:RecordingConfiguration RecordingConfiguration [1][1]
CreateRecordingResponse	Returns the reference to the created recording tt:RecordingReference RecordingToken [1][1]
Fault codes	Description
env:Receiver ter:Action ter:MaxRecordings	The device cannot create a new recording because it already has the maximum number of recordings that it supports.
env:Sender ter:InvalidArgVal ter:BadConfiguration	The RecordConfiguration is invalid.
env:Receiver ter:ActionNotSupported ter:NotImplemented	This optional method is not implemented

When successfully completed, CreateRecording shall have created three tracks with the following configurations:

TrackToken	TrackType
VIDEO001	Video
AUDIO001	Audio
META001	Metadata

All TrackConfigurations shall have the MaximumRetentionTime set to 0 (unlimited), and the Description set to the empty string.

19.5 DeleteRecording

DeleteRecording shall delete a recording object (see Table 261). Whenever a recording is deleted, the device shall delete all the tracks that are part of the recording, and it shall delete all the Recording Jobs that record into the recording. For each deleted recording job, the device shall also delete all the receiver objects associated with the recording job that are automatically created using the AutoCreateReceiver field of the recording job configuration structure and are not used in any other recording job.

This method is optional. It shall be available if the Recording/DynamicRecordings capability is TRUE.

Table 261 – DeleteRecording command

DeleteRecording	
Message name	Description
DeleteRecordingRequest	Identifies the recording that shall be deleted tt:RecordingReference RecordingToken [1][1]
DeleteRecordingResponse	This message shall be empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecording	The RecordingToken does not reference an existing recording
env:Receiver ter: ActionNotSupported ter:NotImplemented	The device cannot delete recordings
env:Receiver ter:Action ter:CannotDelete	This specific recording cannot be deleted

19.6 GetRecordings

GetRecordings shall return a description of all the recordings in the device (see Table 262). This description shall include a list of all the tracks for each recording.

Table 262 – GetRecordings command

GetRecordings	
Message name	Description
GetRecordingsRequest	<i>This shall be an empty message</i>
GetRecordingsResponse	<i>The RecordingItem identifies a recording and its current configuration</i> tt:GetRecordingsResponseItem RecordingItem[0][unbounded]
Fault codes	Description
No command specific faults	

IECNORM.COM: Click to view the full PDF of IEC 62676-2-3:2013

19.7 SetRecordingConfiguration

SetRecordingConfiguration shall change the configuration of a recording (see Table 263).

Table 263 – SetRecordingConfiguration command

SetRecordingConfiguration	
Message name	Description
SetRecordingConfigurationRequest	The RecordingToken shall identify the recording that shall be changed. The RecordingConfiguration shall be the new configuration for that recording tt:RecordingReference RecordingToken[1][1] tt:RecordingConfiguration RecordingConfiguration[1][1]
SetRecordingConfigurationResponse	This message shall be empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:BadConfiguration	The configuration is invalid.
env:Sender ter:InvalidArgVal ter:NoRecording	The RecordingToken does not reference an existing recording

19.8 GetRecordingConfiguration

GetRecordingConfiguration shall retrieve the recording configuration for a recording (see Table 264).

Table 264 – GetRecordingConfiguration command

GetRecordingConfiguration	
Message name	Description
GetRecordingConfigurationRequest	The RecordingToken shall identify the recording for which the configuration shall be retrieved. tt:RecordingReference RecordingToken[1][1]
GetRecordingConfigurationResponse	The RecordingConfiguration shall be the current configuration for the specified recording tt:RecordingConfiguration RecordingConfiguration[1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecording	The <i>RecordingToken</i> does not reference an existing recording

19.9 CreateTrack

This method shall create a new track within a recording (see Table 265).

This method is optional. It shall be available if the Recording/DynamicTracks capability is TRUE.

Table 265 – CreateTrack command

CreateTrack	
Message name	Description
CreateTrackRequest	The RecordingToken shall identify the recording to which a track shall be added. The TrackConfiguration shall provide the configuration for the new track. tt:RecordingReference RecordingToken[1][1] tt:TrackConfiguration TrackConfiguration[1][1]
CreateTrackResponse	The TrackToken shall identify the newly created track. The TrackToken shall be unique within the recording to which the new track belongs. tt:TrackReference TrackToken[1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecording	The RecordingToken does not reference an existing recording
env:Receiver ter:Action ter:MaxTracks	The new track cannot be created because the maximum number of tracks that the device supports for this recording has been reached.
env:Sender ter:InvalidArgVal ter:BadConfiguration	The TrackConfiguration is invalid.
env:Receiver ter:ActionNotSupported ter:NotImplemented	This optional method is not implemented

A TrackToken in itself does not uniquely identify a specific track. Tracks within different recordings may have the same TrackToken.

19.10 DeleteTrack

DeleteTrack shall remove a track from a recording (see Table 266). All the data in the track shall be deleted. This method is optional. It shall be available if the Recording/DynamicTracks capability is TRUE.

Table 266 – DeleteTrack command

DeleteTrack	
Message name	Description
DeleteTrackRequest	The RecordingToken shall identify the recording from which to delete the track. The TrackToken identifies the track to delete. tt:RecordingReference RecordingToken[1][1] tt:TrackReference TrackToken[1][1]
DeleteTrackResponse	<i>This message shall be empty.</i>
Fault codes	Description
env:Receiver ter:ActionNotSupported ter:NotImplemented	<i>The device does not implement the DeleteTrack method.</i>
env:Sender ter:InvalidArgVal ter:NoTrack	<i>The TrackToken does not reference an existing track of the recording.</i>
env:Sender ter:InvalidArgVal ter:NoRecording	<i>The RecordingToken does not reference an existing recording</i>
env:Receiver ter:Action ter:CannotDelete	<i>This specific track cannot be deleted</i>

19.11 GetTrackConfiguration

GetTrackConfiguration shall retrieve the configuration for a specific track (see Table 267).

Table 267 – GetTrackConfiguration command

GetTrackConfiguration	
Message name	Description
GetTrackConfigurationRequest	The RecordingToken and TrackToken shall identify the recording from which to get the track configuration. tt:RecordingReference RecordingToken[1][1] tt:TrackReference TrackToken[1][1]
GetTrackConfigurationResponse	tt:TrackConfiguration TrackConfiguration [1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoTrack	The TrackToken does not reference an existing track of the recording.
env:Sender ter:InvalidArgVal ter:NoRecording	The RecordingToken does not reference an existing recording

19.12 SetTrackConfiguration

SetTrackConfiguration shall change the configuration of a track (see Table 268).

Table 268 – SetTrackConfiguration command

SetTrackConfiguration	
Message name	Description
SetTrackConfigurationRequest	The RecordingToken and TrackToken shall identify the track for which to set the track configuration. The TrackConfiguration is the new configuration for the track. tt:RecordingReference RecordingToken[1][1] tt:TrackReference TrackToken[1][1] tt:TrackConfiguration TrackConfiguration[1][1]
SetTrackConfigurationResponse	This message shall be empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoTrack	The TrackToken does not reference an existing track of the recording.
env:Sender ter:InvalidArgVal ter:NoRecording	The RecordingToken does not reference an existing recording
env:Sender ter:InvalidArgVal ter:BadConfiguration	The contents of the configuration object are invalid.

19.13 CreateRecordingJob

CreateRecordingJob shall create a new recording job (see Table 269).

Table 269 – CreateRecordingJob command

CreateRecordingJob	
Message name	Description
CreateRecordingJobRequest	JobConfiguration shall hold the configuration for the new recording job. tt:RecordingJobConfiguration JobConfiguration[1][1]
CreateRecordingJobResponse	The JobToken shall identify the created recording job. The JobConfiguration structure shall be the configuration as it is used by the device. This may be different from the JobConfiguration passed to CreateRecordingJob. tt:RecordingJobReference JobToken[1][1] tt:RecordingJobConfiguration JobConfiguration[1][1]
Fault codes	Description
env:Receiver ter:Action ter:MaxRecordingJobs	The maximum number of recording jobs that the device can handle has been reached.
env:Sender ter:InvalidArgVal ter:BadConfiguration	The contents of the JobConfiguration are invalid.
env:Receiver ter:Action ter:MaxReceivers	If the AutoCreateReceivers flag is TRUE, this error can be returned if the receiver service cannot create a new receiver.

The **JobConfiguration** returned from CreateRecordingJob shall be identical to the **JobConfiguration** passed into CreateRecordingJob, except for the ReceiverToken and the AutoCreateReceiver. In the returned structure, the ReceiverToken shall be present and valid and the AutoCreateReceiver field shall be omitted.

19.14 DeleteRecordingJob

DeleteRecordingJob removes a recording job (see Table 270). It shall also implicitly delete all the receiver objects associated with the recording job that are automatically created using the AutoCreateReceiver field of the recording job configuration structure and are not used in any other recording job.

Table 270 – DeleteRecordingJob command

DeleteRecordingJob	
Message name	Description
DeleteRecordingJobRequest	The JobToken shall identify the recording job that shall be deleted. tt:RecordingJobReference JobToken[1][1]
DeleteRecordingJobResponse	The message shall be empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	The JobToken does not reference an existing job.

19.15 GetRecordingJobs

GetRecordingJobs shall return a list of all the recording jobs in the device (see Table 271).

Table 271 – GetRecordingJobs command

GetRecordingJobs	
Message name	Description
GetRecordingJobsRequest	This message shall be empty.
GetRecordingJobsResponse	The JobItem identifies a job in the device and holds its current configuration. tt:GetRecordingJobsResponseItem JobItem[0][unbounded]
Fault codes	Description
No command specific faults	

19.16 SetRecordingJobConfiguration

SetRecordingJobConfiguration shall change the configuration for a recording job (see Table 272).

Table 272 – SetRecordingJobConfiguration command

SetRecordingJobConfiguration	
Message name	Description
SetRecordingJobConfigurationRequest	The <i>JobConfiguration</i> returned from <i>SetRecordingJobConfiguration</i> shall be identical to the <i>JobConfiguration</i> passed into <i>SetRecordingJobConfiguration</i>, except for the <i>ReceiverToken</i> and the <i>AutoCreateReceiver</i>. In the returned structure, the <i>ReceiverToken</i> shall be present and valid and the <i>AutoCreateReceiver</i> field shall be omitted. tt:RecordingJobReference JobToken[1][1] tt:RecordingJobConfiguration JobConfiguration[1][1]
SetRecordingJobConfigurationResponse	The JobConfiguration structure shall be the configuration as it is used by the device. This may be different from the <i>JobConfiguration</i> passed to <i>CreateRecordingJob</i>. tt:RecordingJobConfiguration JobConfiguration[1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	The <i>JobToken</i> does not reference an existing job
env:Sender ter:InvalidArgVal ter:BadConfiguration	The contents of the <i>JobConfiguration</i> are invalid.
env:Receiver ter:Action ter:MaxReceivers	If the <i>AutoCreateReceivers</i> flag is <i>TRUE</i> , this error can be returned if the receiver service cannot create a new receiver.

SetRecordingJobConfiguration shall implicitly delete any receiver objects that were created automatically if they are no longer used as a result of changing the recording job configuration.

19.17 GetRecordingJobConfiguration

GetRecordingJobConfiguration shall return the current configuration for a recording job (see Table 273).

Table 273 – GetRecordingJobConfiguration command

GetRecordingJobConfiguration	
Message name	Description
GetRecordingJobConfigurationRequest	The JobToken shall identify the recording job for which to retrieve the configuration. tt:RecordingJobReference JobToken[1][1]
GetRecordingJobConfigurationResponse	The JobConfiguration shall hold the current configuration of the recording job. tt:RecordingJobConfiguration JobConfiguration[1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	The JobToken does not reference an existing job

19.18 SetRecordingJobMode

SetRecordingJobMode shall change the mode of the recording job (see Table 274). Using this method shall be equivalent to retrieving the recording job configuration, and writing it back with a different mode.

Table 274 – SetRecordingJobMode command

SetRecordingJobMode	
Message name	Description
SetRecordingJobModeRequest	The JobToken shall identify the recording job for which to change the recording mode. The mode shall be the new mode for the recording job. tt:RecordingJobReference JobToken[1][1] tt:RecordingJobMode Mode[1][1]
SetRecordingJobModeResponse	This message shall be empty.
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	The JobToken does not reference an existing job
env:Sender ter:InvalidArgVal ter:BadMode	The Mode is invalid.

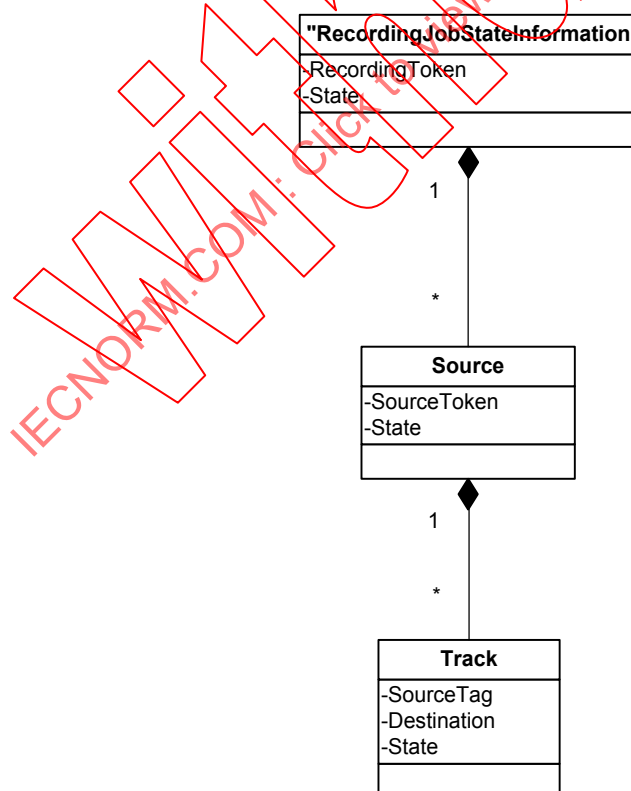
19.19 GetRecordingJobState

GetRecordingJobState returns the state of a recording job (see Table 275). It includes an aggregated state, and state for each track of the recording job.

Table 275 – GetRecordingJobState command

GetRecordingJobState	
Message name	Description
GetRecordingJobState Request	The JobToken shall identify the recording job for which to get the state. tt:RecordingJobReference JobToken[1][1]
GetRecordingJobState Response	The State shall hold the state of the recording job. tt:RecordingJobStateInformation State[1][1]
Fault codes	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	The JobToken does not reference an existing job

The UML representation of the RecordingJobStateInformation structure is shown in Figure 26.



IEC 2766/13

Figure 26 – Diagram of the RecordingJobStateInformation elements

RecordingToken shall be the identification of the recording that the recording job records to.

State (as part of RecordingJobStateInformation) shall hold the aggregated state over the whole RecordingJobInformation structure.

SourceToken shall identify the data source of the recording job.

State (as part of RecordingJobStateSource) shall hold the aggregated state over all substructures of RecordingJobStateSource.

SourceTag shall identify the track of the data source that provides the data.

Destination shall indicate the destination track

State (as part of RecordingJobTrackState) shall provide the job state of the track. The valid values of state shall be "Idle", "Active" and "Error". If state equals "Error", the Error field may be filled in with an implementation defined value.

Error, if present, shall hold an implementation defined string value that describes the error. The string should be in the English language.

A device shall apply the following rules to compute aggregate state

Idle	All state values in sub-nodes are "Idle"
PartiallyActive	The state of some sub-nodes are "active" and some sub-nodes are "idle"
Active	The state of all sub-nodes is "Active"
Error	At least one of the sub-nodes has state "Error"

19.20 Events

The recording service shall dispatch events through the event service. It shall be capable of generating the events listed in this subclause whenever the condition that fires the event occurs.

Some of these events are similar to the automatically generated events that can be searched for by the FindEvents method in the search service. See 20.17.

19.20.1 Recording job state changes

If the a state field of the RecordingJobStateInformation structure changes, the device shall send the event:

```

Topic: tns1:RecordingConfig/JobState
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingJobToken"
Type="tt:RecordingJobReference"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItemDescription Name="State" Type="xs:String"/>
    <tt:ElementItemDescription Name="Information"
Type="tt:RecordingJobStateInformation"/>
  </tt>Data>
</tt:MessageDescription>

```

19.20.2 Configuration changes

If the configuration of a recording is changed, the device shall send the event:

```
Topic: tns1:RecordingConfig/RecordingConfiguration
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItemDescription Name="Configuration"
Type="tt:RecordingConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

If the configuration of a track is changed, the device shall send the event:

```
Topic: tns1:RecordingConfig/TrackConfiguration
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItemDescription Name="Configuration"
Type="tt:TrackConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

If the configuration of a recording job is changed, the device shall send the event:

```
Topic: tns1:RecordingConfig/RecordingJobConfiguration
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingJobToken"
Type="tt:RecordingJobReference"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItemDescription Name="Configuration"
Type="tt:RecordingJobConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

19.20.3 Data deletion

Whenever data is deleted, the device shall send the event:

```
Topic: tns1:RecordingConfig/DeleteTrackData
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItemDescription Name="StartTime" Type="xsDateTime"/>
    <tt:SimpleItemDescription Name="EndTime" Type="xsDateTime"/>
  </tt>Data>
</tt:MessageDescription>
```

19.20.4 Recording and track creation and deletion

Whenever a recording is created, the device shall send the event:

```
Topic: tns1:RecordingConfig/CreateRecording
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

Whenever a recording is deleted, the device shall send the event:

```
Topic: tns1:RecordingConfig/DeleteRecording
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

Whenever a track is created, the device shall send the event:

```
Topic: tns1:RecordingConfig/CreateTrack
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

Whenever a track is deleted, the device shall send the event:

```
Topic: tns1:RecordingConfig/DeleteTrack
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

19.21 Examples

19.21.1 Example 1: Setup recording of a single camera

There are two steps involved. The first step is to configure the NVS

```
; Create recording (this implicitly creates an A, V and M track)
```

```
RecordToken = CreateRecording(RecordConfiguration)
```

; The tracktokens are predefined. We don't have to find them on the device

```
TrackToken1 = "VIDEO001"
```

```
TrackToken2 = "AUDIO001"
```

```
TrackToken3 = "META001"
```

```
; Create a recording job, assume that we set mode to idle, auto create receiver
```

```
JobToken, ActualJobConfig = CreateRecordingJob(JobConfiguration)
```

```
; Configure the receiver
```

```
ConfigureReceiver(ActualJobConfiguration.ReceiverToken, ReceiverConfiguration)
```

This completes the configuration step.

Finally, to really start recording, some entity calls

```
; Activate the recording job
```

```
SetRecordingJobMode(JobToken, Active)
```

to make the job active. This will cause the NVS to set up an RTSP connection with the device.

Therefore, to start and stop recording, all that is needed is to call SetRecordingJobMode on pre-configured recording jobs. And since the embedded configuration objects are persistent, the configuration cycle only needs to be done once.

19.21.2 Example 2: Record multiple streams from one camera to a single recording

This example is very similar to example 1. The jobconfiguration will hold references to two receiver objects. Each receiver object is configured to receive from the same device, but from a different stream.

```
; Create recording (this implicitly creates an A, V and M track)
```

```
RecordToken = CreateRecording(RecordConfiguration)
```

```
; The tracktokens are predefined. We don't have to find them on the device
```

```
TrackToken1 = "VIDEO001"
```

```
TrackToken2 = "AUDIO001"
```

```
TrackToken3 = "META001"
```

```

; Create three additional tracks

TrackToken4 = CreateTrack(RecordToken, AudioConfig)

TrackToken5 = CreateTrack(RecordToken, VideoConfig)

TrackToken6 = CreateTrack(RecordToken, MetadataConfig)


; Create a recording job, assume that we set mode to idle, auto create
two receivers

JobToken, ActualJobConfiguration = CreateRecordingJob(JobConfiguration)


; Configure the receivers

ConfigureReceiver(ActualJobConfiguration.ReceiverToken[1],

Receiver1Configuration)

ConfigureReceiver(ActualJobConfiguration.ReceiverToken[2],

Receiver2Configuration)

```

To really start recording, some entity calls

```

; Activate the recording job

SetRecordingJobMode(JobToken, Active)

```

20 Recording search

20.1 General

The search service provides a number of operations for finding data of interest within a set of recordings. The most common way of doing this would be to search for events that are either included in the metadata track of a recording, or are otherwise associated with a recording in the device (see Recording Events below).

GetRecordingSummary returns a summary for all recording, and can be used to provide the scale of a timeline.

GetRecordingInformation returns information about a single recording, such as start time and current status.

GetMediaAttributes returns the media attributes of a recording at a specific point in time.

The actual search is done by coupled find and result operations. Each find operation initiates a search session. The client can then acquire the results from the search session in increments, or all at once, depending on implementation and the scale of the search. There are four pairs of search operations for Recordings, Recording Events, PTZ Positions and Metadata.

GetSearchState returns the state of a search session.

EndSearch ends a search session, halting search and returning any blocking result operations.

20.2 Concepts

20.2.1 Search direction

Search is performed from a start point on the time line, towards an end point. If the end point is prior to the start point, search will be performed backwards. This can be useful if only the newest matching event is of interest, or if it is otherwise convenient to get the results in newest to oldest order.

If no end point is specified, the search will always be performed forward in time from the start point.

20.2.2 Recording event

Describes a discrete event related to the recording. It is represented as a notification message, but this does not necessarily mean it has been recorded as a notification. Recording events can either be notifications included in a recorded metadata track, it can be created by the recording device as a result of an internal event or mechanism, or it can be inserted by a client using a WebService request or a metadata stream. However the recording event has been created and associated with a particular recording, this specification makes no implications on how it is stored internally on a device, only how it should be represented in the interface.

However created, recording events are always treated as notifications in regards to search filters and results returned. Each recording event has a notification topic as defined in 15.7 of the core specification. Predefined recording events are described in 20.17.

To communicate the original state of property events, virtual start state events can be returned in a search result containing the value of one or more properties at the start point of the search interval. Such start state events are virtual events in the sense that they are created on the fly by the server, rather than being collected from recorded data. If the client indicates that such events are desired by setting the appropriate flag, virtual events matching the topics defined in the search filter should be returned for any recording in the search scope.

20.2.3 Search session

A search session is started asynchronously by a Find-operation and is identified by a search token unique for that session. Results are returned in increments using GetResult-operations referring to the session created by the Find-operation. The search can be terminated in three ways:

- KeepAlive time expires – if no request from a client has been made that refers to a particular session within the specified time interval, it will terminate;
- A GetResult method returns the last data for the search session by setting the search state in its result to “Completed”;

- EndSearch – the client explicitly ends a session.

Ending a session will cancel an ongoing search, immediately return and make further requests to the same session result in an error message. A device shall not reuse search token immediately as it would confuse clients unaware that a session had ended.

20.2.4 Search scope

The scope contains a number of optional elements, together limiting the set of data to look into when performing searches.

20.2.4.1 Included data

Optionally, the client can define sources and recordings to search in by specifying lists of tokens for each type. If several types are given, the union of the specified tokens shall be used. If there are no sources or recordings tokens specified, all recordings shall be included. The scope is further refined by the Recording Information Filter. However, if recordings are specified the filters will only be applied to that subset of recordings.

20.2.4.2 Recording information filter

Rather than specifying a list of recording tokens, the recordings can be filtered by an XPath filter operating on the RecordingInformation structure. This allows the client to filter on all elements present in the RecordingInformation structure, using comparisons according to the XPath dialect defined in 20.18. If a recording information filter is supplied, only recordings matching the filter shall be part of the scope.

Example of a filter that includes only recordings containing audio in the search scope:

```
boolean(//Tracks[TrackType = "Audio"])
```

20.2.5 Search filters

Search filters are specific for the type of search operation. See FindEvents, FindPTZPosition, FindMetadata respectively. They all act on the recordings defined by the scope.

20.3 Data structures

20.3.1 RecordingInformation structure

RecordingInformation contains information about a recording, the tracks it consists of and the source.

- RecordingToken – a unique identifier of the recording;
- EarliestRecording – the date and time of the oldest data in the recording;
- LatestRecording – the date and time of the newest data in the recording;
- Content – informative description of content;
- RecordingStatus – current status of recording, can be any of: Initiated, Recording, Stopped, Removing, Removed;
- RecordingSourceInformation – a structure containing information about the source of the recording;
- TrackInformation – a list of track information structures.

20.3.2 RecordingSourceInformation structure

Contains information about the source of a recording.

- SourceId – an identifier for the source chosen by the client that creates the recording. This identifier is opaque to the NVS. Clients may use any type of URI for this field;
- Name – informative name of the source;
- Location – informative description of the location of the source;
- Description – informative description of the source;
- Address – informative URI of the source.

20.3.3 TrackInformation structure

Contains information about a single track in a recording.

- TrackToken – an identifier of the track. The TrackToken is unique between all TrackTokens used within a recording;
- TrackType – identifies the type of track (video, audio or metadata);
- Description – informative description of the track;
- DataFrom – The date and time of the oldest recorded data in the track;
- DataTo – The date and time of the newest recorded data in the track.

20.3.4 SearchState enumeration

The search state can be one of the following

- Queued – meaning that the search has not yet begun;
- Searching – meaning that the search is under way, and new results can be produced;
- Completed – meaning that the search is completed and no new results will be produced.

20.3.5 MediaAttributes structure

The MediaAttributes contains information about the media tracks of a particular recording for a particular time frame. The time frame can be a single point in time, in which case the *From* and *Until* elements are identical.

- RecordingToken – a reference to the recording that this structure concerns;
- From – a point in time from when the attributes are valid for the recording;
- Until – a point in time until when the specified attributes are valid for the recording;
- VideoAttributes – a set of video attributes, describing the data of a recorded video track;
- AudioAttributes – a set of audio attributes, describing the data of a recorded audio track;
- MetadataAttributes – a set of attributes, describing the possible metadata content of a recorded metadata track.

20.3.6 FindEventResult structure

- RecordingToken – identifying the recording containing the found event;
- TrackToken – identifying the track containing the found event;
- Time – the date and time of the found event;
- Event – the event message found;
- StartStateEvent – if true, indicates the event represents the start state of one or more properties in the recording.

20.3.7 FindPTZPositionResult structure

- RecordingToken – identifying the recording containing the matching position;
- TrackToken – identifying the track containing the matching position;
- Time – the date and time of the matching position;
- Position – the matching PTZ vector.

20.3.8 PTZPositionFilter structure

Contains the necessary elements to define what PTZ positions to search for. The PTZ vectors shall be in the same coordinate space as the PTZ coordinates stored in the recording.

- MinPosition – the lower boundary of the PTZ volume to look for;
- MaxPosition – the upper boundary of the PTZ volume to look for;
- EnterOrExit – if true, search for when entering or exiting the specified PTZ volume.

20.3.9 MetadataFilter structure

Contains an XPath expression to be applied to the MetadataStream structure.

Example of an expression searching for objects overlapping the lower right quadrant of the scene:

```
boolean(//Object/Appearance/Shape/BoundingBox[@right > "0.5"])
and
boolean(//Object/Appearance/Shape/BoundingBox[@bottom > "0.5"])
```

20.3.10 FindMetadataResult structure

- RecordingToken – identifying the recording containing the matching metadata;
- TrackToken – identifying the track containing the matching metadata;
- Time – the date and time of the matching metadata.

20.4 GetRecordingSummary

GetRecordingSummary is used to get a summary description of all recorded data (see Table 276). This operation is mandatory to support for a device implementing the recording search service.

Table 276 – GetRecordingSummary command

GetRecordingSummary		Request-Response
Message name	Description	
GetRecordingSummaryRequest	This shall be the empty message	
GetRecordingSummaryResponse	Returns a structure containing: <i>DataFrom</i> specifying the first time when there is recorded data on the device; <i>DataUntil</i> specifying the last point in time where there is data recorded on the device; the estimated total number of recordings and tracks. tt:RecordingSummary summary[1][1]	
Fault codes	Description	
	No command specific error codes	

20.5 GetRecordingInformation

Returns information about a single *Recording* specified by a *RecordingToken* (see Table 277). This operation is mandatory to support for a device implementing the recording search service.

Table 277 – GetRecordingInformation command

GetRecordingInformation		Request-Response
Message name	Description	
GetRecordingInformationRequest	<i>Request description</i> tt:ReferenceToken RecordingToken [1][1]	
GetRecordingInformationResponse	<i>Response description</i> tt:RecordingInformation RecordingInformation [1][1]	
Fault codes	Description	
env:Sender ter: InvalidArgVal ter: InvalidToken	<i>The RecordingToken is not valid.</i>	

20.6 GetMediaAttributes

Returns a set of media attributes for all tracks of the specified recordings at a specified point in time (see Table 278). Clients using this operation shall be able to use it as a non blocking operation. A device shall set the starttime and endtime of the MediaAttributes structure to equal values if calculating this range would causes this operation to block. See

MediaAttributes structure for more information. This operation is mandatory to support for a device implementing the recording search service.

Table 278 – GetMediaAttributes command

GetMediaAttributes		Request-Response
Message name	Description	
GetMediaAttributesRequest	<i>RecordingTokens</i> is a list of references to the recordings to query. <i>Time</i> is the point in time where for which information is requested. tt:ReferenceToken RecordingTokens [0][unbounded] xs:dateTime Time[1][1]	
GetMediaAttributesResponse	Contains a <i>MediaAttributes</i> structure for each <i>RecordingToken</i> specified in the request. tt:MediaAttributes MediaAttributes [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidToken	<i>The RecordingToken is not valid.</i>	

20.7 FindRecordings

FindRecordings starts a search session, looking for recordings that matches the scope (see 20.2.4) defined in the request (see Table 279). Results from the search are acquired using the GetRecordingSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- the entire time range from *StartPoint* to *EndPoint* has been searched through;
- the total number of matches has been found, defined by the *MaxMatches* parameter;
- the session has been ended by a client EndSearch request;
- the session has been ended because *KeepAliveTime* since the last request related to this session has expired.

The order of the results is undefined, to allow the device to return results in any order they are found. This operation is mandatory to support for a device implementing the recording search service.

Table 279 – FindRecordings command

FindRecordings		Request-Response
Message name	Description	
FindRecordingsRequest	<i>Scope</i> defines the dataset to consider for this search. The search ends after <i>MaxMatches</i>. <i>KeepAliveTime</i> is the session timeout after each request concerning this session. tt:SearchScopeScope [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]	
FindRecordingsResponse	Returns the <i>SearchToken</i> identifying the search session created by this request. tt:JobToken SearchToken [1][1]	
Fault codes	Description	
env:Receiver ter:Action ter:ResourceProblem	<i>Device is unable to create a new search session.</i>	

20.8 GetRecordingSearchResults

GetRecordingSearchResults acquires the results from a recording search session previously initiated by a FindRecordings operation (see Table 280). The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results.

GetRecordingSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified;
- *MinResults* results are available for the response if *MinResults* is specified;
- *WaitTime* has expired;
- Search is completed or stopped.

This operation is mandatory to support for a device implementing the recording search service.

Table 280 – GetRecordingSearchResults command

GetRecordingSearchResults		Request-Response
Message name	Description	
GetRecordingSearchResultsRequest	<i>SearchToken</i> specifies the search session. <i>MinResults</i> specifies the minimum number of results that should be returned. If the total number of results is lower than <i>MinResults</i> in a completed search, all results should be returned. <i>MaxResults</i> specifies the maximum number of results to return. <i>WaitTime</i> defines the maximum time to block, waiting for results. tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]	
GetRecordingSearchResultsResponse	Returns a structure containing the current <i>SearchState</i> and a list of <i>RecordingInformation</i> structures. tt:FindRecordingResultList ResultList[1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidToken	<i>The search token is invalid.</i>	

20.9 FindEvents

FindEvents starts a search session, looking for recording events (see 20.2.2) in the *scope* (see 20.2.4) that matches the search filter defined in the request (see Table 281). Results from the search are acquired using the GetEventSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- the entire time range from *StartPoint* to *EndPoint* has been searched through;
- the total number of matches has been found, defined by the *MaxMatches* parameter;
- the session has been ended by a client EndSearch request;
- the session has been ended because *KeepAliveTime* since the last request related to this session has expired.

Results shall be ordered by time, ascending in case of forward search, or descending in case of backward search. This operation is mandatory to support for a device implementing the recording search service.

Table 281 – FindEvents command

FindEvents		Request-Response
Message name	Description	
FindEventsRequest	<i>StartPoint</i> is the point of time where the search will start. <i>EndPoint</i> is the point of time where the search will stop. This can be a time before the <i>StartPoint</i>, in which case the search is performed backwards in time. If <i>EndPoint</i> is omitted, search will go forward from the <i>StartPoint</i>. <i>Scope</i> defines the dataset to consider for this search. <i>SearchFilter</i> contains the topic and message filter needed to define what events to search for. By setting the <i>IncludeStartState</i> to true, the client indicates that virtual events at the time of <i>StartPoint</i> should be returned to represent the state in the recording. The search ends after <i>MaxMatches</i>. <i>KeepAliveTime</i> is the session timeout after each request concerning this session. xs:dateTime StartPoint [1][1] xs:dateTime EndPoint [0][1] tt:SearchScope Scope [1][1] tt:EventFilter SearchFilter [1][1] xs:boolean IncludeStartState [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]	
FindEventsResponse	Returns the <i>SearchToken</i> identifying the search session created by this request. tt:JobToken SearchToken [1][1]	
Fault codes	Description	
env:Receiver ter:Action ter:ResourceProblem	<i>Device is unable to create a new search session.</i>	

20.10 GetEventSearchResults

GetEventSearchResults acquires the results from a recording event search session previously initiated by a FindEvents operation (see Table 282). The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results.

GetEventSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified;
- *MinResults* results are available for the response if *MinResults* is specified;
- *WaitTime* has expired;
- Search is completed or stopped.

This operation is mandatory to support for a device implementing the recording search service.

Table 282 – GetEventSearchResults command

GetEventSearchResults		Request-Response
Message name	Description	
GetEventSearchResultsRequest	<i>SearchToken</i> specifies the search session. <i>MinResults</i> specifies the minimum number of results that should be returned. <i>MaxResults</i> specifies the maximum number of results to return. <i>WaitTime</i> defines the maximum time to block, waiting for results. tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]	
GetEventSearchResultsResponse	Returns a structure containing the current <i>SearchState</i> and a list of <i>FindEventResult</i> structures. tt:SearchState SearchState [1][1] tt:FindEventResult FindEventResult [0][unbounded]	
Fault codes	Description	
env:Sender	The search token is invalid.	
ter:InvalidArgVal		
ter:InvalidToken		

20.11 FindPTZPosition

FindPTZPosition starts a search session, looking for ptz positions in the *scope* (see 20.2.4) that matches the search filter defined in the request (see Table 283). Results from the search are acquired using the GetPTZPositionSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- the entire time range from *StartPoint* to *EndPoint* has been searched through;
- the total number of matches has been found, defined by the *MaxMatches* parameter;
- the session has been ended by a client EndSearch request;
- the session has been ended because *KeepAliveTime* since the last request related to this session has expired.

This operation is mandatory to support whenever CanContainPTZ is true for any metadata track in any recording on the device.

Table 283 – FindPTZPosition command

FindPTZPosition		Request-Response
Message name	Description	
FindPTZPositionRequest	<i>StartPoint</i> is the point of time where the search will start. <i>EndPoint</i> is the point of time where the search will stop. This can be a time before the <i>StartPoint</i>, in which case the search is performed backwards in time. If <i>EndPoint</i> is omitted, search will go forward from the <i>StartPoint</i>. <i>Scope</i> defines the dataset to consider for this search. <i>SearchFilter</i> contains the search criteria needed to define the PTZ position to search for. The search ends after <i>MaxMatches</i>. <i>KeepAliveTime</i> is the session timeout after each request concerning this session. xs:dateTime StartPoint [1][1] xs:dateTime EndPoint [0][1] tt:SearchScopeScope [1][1] tt:PTZPositionFilter SearchFilter [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]	
FindPTZPositionResponse	Returns the <i>SearchToken</i> identifying the search session created by this request. tt:JobToken SearchToken [1][1]	
Fault codes	Description	
env:Receiver ter:Action ter:ResourceProblem	Device is unable to create a new search session.	
env:Receiver ter:ActionNotSupported ter:NotImplemented	This optional method is not implemented	

20.12 GetPTZPositionSearchResults

GetPTZPositionSearchResults acquires the results from a PTZ position search session previously initiated by a FindPTZPosition operation (see Table 284). The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results.

GetPTZPositionSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified;
- *MinResults* results are available for the response if *MinResults* is specified;
- *WaitTime* has expired;
- Search is completed or stopped.

This operation is mandatory to support whenever CanContainPTZ is true for any metadata track in any recording on the device.

Table 284 – GetPTZPositionSearchResults command

GetPTZPositionSearchResults		Request-Response
Message name	Description	
GetPTZPositionSearchResultsRequest	<i>SearchToken</i> specifies the search session. <i>MinResults</i> specifies the minimum number of results that should be returned. <i>MaxResults</i> specifies the maximum number of results to return. <i>WaitTime</i> defines the maximum time to block, waiting for results. tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]	
GetPTZPositionSearchResultsResponse	Returns a structure containing the current <i>SearchState</i> and a list of <i>FindPTZPositionResult</i> structures. tt:SearchState SearchState [1][1] tt:FindPTZPositionResult FindPTZPositionResult [0][unbounded]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidToken	<i>The search token is invalid.</i>	

20.13 FindMetadata

FindMetadata starts a search session, looking for metadata in the scope (see 20.2.4) that matches the search filter defined in the request (see Table 285). Results from the search are acquired using the GetMetadataSearchResults request, specifying the search token returned from this request.

The device shall continue searching until one of the following occurs:

- the entire time range from *StartPoint* to *EndPoint* has been searched through;
- the total number of matches has been found, defined by the *MaxMatches* parameter;
- the session has been ended by a client EndSearch request;
- the session has been ended because *KeepAliveTime* since the last request related to this session has expired.

This operation is mandatory to support if the MetadataSearch capability is set to true in the SearchCapabilities structure returned by the GetCapabilities command in the Device service.

Table 285 – FindMetadata command

FindMetadata		Request-Response
Message name	Description	
FindMetadataRequest	<i>StartPoint</i> is the point of time where the search will start. <i>EndPoint</i> is the point of time where the search will stop. This can be a time before the <i>StartPoint</i>, in which case the search is performed backwards in time. If <i>EndPoint</i> is omitted, search will go forward from the <i>StartPoint</i>. <i>Scope</i> defines the dataset to consider for this search. <i>SearchFilter</i> contains the search criteria needed to define the metadata to search for. The search ends after <i>MaxMatches</i>. <i>KeepAliveTime</i> is the session timeout after each request concerning this session. xs:dateTime StartPoint [1][1] xs:dateTime EndPoint [0][1] tt:SearchScope Scope [1][1] tt:MetadataFilter SearchFilter [1][1] xs:int MaxMatches [0][1] xs:duration KeepAliveTime [1][1]	
FindMetadataResponse	Returns the <i>SearchToken</i> identifying the search session created by this request. tt:JobToken SearchToken [1][1]	
Fault codes	Description	
env:Receiver ter:Action ter:ResourceProblem	<i>Device is unable to create a new search session.</i>	

20.14 GetMetadataSearchResults

GetMetadataSearchResults acquires the results from a recording search session previously initiated by a FindMetadata operation (see Table 286). The response shall not include results already returned in previous requests for the same session. If *MaxResults* is specified, the response shall not contain more than *MaxResults* results.

GetMetadataSearchResults shall block until:

- *MaxResults* results are available for the response if *MaxResults* is specified;
- *MinResults* results are available for the response if *MinResults* is specified;
- *WaitTime* has expired;
- Search is completed or stopped.

This operation is mandatory to support if the MetaDataSearch capability is set to true in the SearchCapabilities structure return by the GetCapabilities command in the Device service.

Table 286 – GetMetadataSearchResults command

GetMetadataSearchResults		Request-Response
Message name	Description	
GetMetadataSearchResults Request	<i>SearchToken</i> specifies the search session. <i>MinResults</i> specifies the minimum number of results that should be returned. <i>MaxResults</i> specifies the maximum number of results to return. <i>WaitTime</i> defines the maximum time to block, waiting for results. tt:JobToken SearchToken [1][1] xs:int MinResults [0][1] xs:int MaxResults [0][1] xs:duration WaitTime [0][1]	
GetMetadataSearchResults Response	Returns a structure containing the current <i>SearchState</i> and a list of <i>FindMetadataResult</i> structures. tt:SearchState SearchState [1][1] tt:FindMetadataResult FindMetadataResult [0][unbounded]	
Fault codes	Description	
env:Sender	<i>The search token is invalid.</i>	
ter:InvalidArgVal		
ter:InvalidToken		

20.15 GetSearchState

GetSearchState returns the current state of the specified search session (see Table 287). Queued, Searching or Completed. This operation is mandatory to support for a device implementing the recording search service.

Table 287 – GetSearchState command

GetSearchState		Request-Response
Message name	Description	
GetSearchStateRequest	<i>SearchToken</i> specifies the search session. tt:JobToken SearchToken [1][1]	
GetSearchStateResponse	Returns the current state of the search session. tt:SearchState State [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidToken	<i>The search token is invalid.</i>	

20.16 EndSearch

EndSearch stops an ongoing search session, causing any blocking result request to return and the *SearchToken* to become invalid (see Table 288). If the search was interrupted before completion, the point in time that the search had reached shall be returned. If the search had not yet begun, the *StartPoint* shall be returned. If the search was completed the original *EndPoint* supplied by the Find operation shall be returned. This operation is mandatory to support for a device implementing the recording search service.

Table 288 – EndSearch command

EndSearch		Request-Response
Message name	Description	
EndSearchRequest	<i>SearchToken</i> specifies the search session. tt:JobToken SearchToken [1][1]	
EndSearchResponse	Returns the point in time where the search was at when ended. xs:dateTime EndPoint [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:InvalidToken	<i>The search token is invalid.</i>	

20.17 Recording Event Descriptions

A device shall generate the following events with the corresponding event message descriptions. A device supporting the recording search service shall record these notification messages so that clients can use FindEvents to search for these messages. All recording events that are generated by the device and inserted into the recording history shall have a root topic of tns1:RecordingHistory.

Topic: tns1:RecordingHistory/Recording/State

```
<tt:MessageDescription IsProperty="true">

  <tt:Source>

    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:ReferenceToken"/>

  </tt:Source>

  <tt>Data>

    <tt:SimpleItemDescription Name="IsRecording" Type="tt:boolean"/>

  </tt>Data>

</tt:MessageDescription>
```

This message is sent whenever a client starts or stops recording for a specific recording. At start recording, IsRecording shall be set to true. At stop recording, IsRecording shall be set to false.

Topic: tns1:RecordingHistory/Track/State

```
<tt:MessageDescription IsProperty="true">

  <tt:Source>

    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:ReferenceToken"/>

    <tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>

  </tt:Source>

  <tt>Data>

    <tt:SimpleItemDescription Name="IsDataPresent" Type="tt:boolean"/>

  </tt>Data>

</tt:MessageDescription>
```

This message signals when the data for a track is present. When the data becomes present, a message with `IsDataPresent` set to `TRUE` shall be sent. When the data becomes unavailable, The message with `IsDataPresent` set to `FALSE` shall be sent.

An NVS MAY generate the following events. If the NVS supports these events, it shall always automatically records these notification messages so that clients can always use `FindEvent` for these messages.

Topic: `tns1:RecordingHistory/Track/VideoParameters`

```
<tt:MessageDescription IsProperty="true">

  <tt:Source>

    <tt:SimpleItemDescription                                Name="Recording"
Type="tt:ReferenceToken"/>

    <tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>

  </tt:Source>

  <tt>Data>

    <tt:SimpleItemDescription                                Name="VideoEncoding"
Type="tt:VideoEncoding"/>

    <tt:SimpleItemDescription Name="VideoWidth" Type="xs:int"/>

    <tt:SimpleItemDescription Name="VideoHeight" Type="xs:int"/>

    <tt:SimpleItemDescription                                Name="tt:RateControl"
Type="VideoRateControl"/>

  </tt>Data>

</tt:MessageDescription>
```

Topic: `tns1:RecordingHistory/Track/AudioParameters`

```
<tt:MessageDescription IsProperty="true">

  <tt:Source>

    <tt:SimpleItemDescription                                Name="Recording"
Type="tt:ReferenceToken"/>

    <tt:SimpleItemDescription Name="Track" Type="tt:ReferenceToken"/>

  </tt:Source>
```



```

<tt:Data>

    <tt:SimpleItemDescription                                Name="AudioEncoding"
Type="tt:AudioEncoding"/>

    <tt:SimpleItemDescription Name="AudioSampleRate" Type="xs:int"/>

    <tt:SimpleItemDescription Name="AudioBitrate" Type="xs:int"/>

</tt:Data>

</tt:MessageDescription>

```

The NVS shall send either message (depending on the track data type) whenever any of these properties change.

20.18 XPath dialect

This subclause defines the XPATH dialect that a device that realises the search service shall implement to parse the XPath strings that are passed to the methods of the search service.

```

Dialect=http://www.onvif.org/ver10/tse/searchFilter

[1] Expression ::= BoolExpr | Expression 'and' Expression
                | Expression 'or' Expression | '(' Expression ')' |
                'not' '(' Expression ')'
[2] BoolExpr ::= 'boolean' '(' PathExpr ')' | 'contains' '(' ElementPath
                ',' 'String' ')'
[3] PathExpr ::= '//SimpleItem' NodeTest | '//ElementItem' NodeTest |
                ElementTest
[4] NodeTest ::= '[' AttrExpr ']'
[5] AttrExpr ::= NameComp | ValueComp | AttrExpr 'and' AttrExpr |
                AttrExpr 'or' AttrExpr | 'not' '(' AttrExpr ')'
[6] NameComp ::= NameAttr '=' 'String'
[7] ValueComp ::= ValueAttr Operator 'String'
[8] Operator ::= '=' | '!=' | '<' | '<=' | '>' | '>='
[9] NameAttr ::= '@Name'
[10] ValueAttr ::= '@Value'
[11] ElementTest ::= '/' ElementPath '[' NodeComp ']'
[12] ElementPath ::= ElementName ElementName*
[13] ElementName ::= '/' String
[14] NodeComp ::= NodeName Operator 'String'
[15] NodeName ::= '@' String | String

```

Example of an XPath expression used to find recordings from the basement where there is at least one track containing video:

```

boolean(//Source[Location = "Basement"]) and
boolean(//Tracks[TrackType = "Video"])

```

21 Replay control

This clause describes the use of RTSP for replaying recorded streams, and defines a service for mapping replay endpoints to URI for use in RTSP.

21.1 Use of RTSP

The replay protocol is based on RTSP [RFC 2326]. However because RTSP does not directly support many of the features required by VSS applications, this standard defines several extensions to the protocol; these are detailed below.

This standard makes the following stipulations on the usage of RTSP:

1. RTP/RTSP/HTTP/TCP shall be supported by the server. This is the same transport protocol as a device that implements media streaming through the media service shall support, and the same requirements shall apply to replay streaming.
2. the server shall support the unicast RTP/UDP transport for streaming.
3. clients should use a TCP-based transport for replay, in order to achieve reliable delivery of media packets.
4. the server MAY elect not to send RTCP packets during replay. In typical usage RTCP packets are not required, because usually a reliable transport will be used, and because absolute time information is sent within the stream, making the timing information in RTCP sender reports redundant.

RTSP describe

The SDP returned by the RTSP describe command shall include the TrackReference for each track of the recording to allow a client to map the tracks presented in the SDP to tracks of the recording. The tag shall use the following format:

a:x-onvif-track:<TrackReference>

For example:

```
NVS - NVT:    DESCRIBE rtsp://192.168.0.1 RTSP/1.0
              CSeq: 1
              User-Agent: ONVIF Rtsp client
              Accept: application/sdp
```

```
NVT - NVS:    RTSP/1.0 200 OK
              CSeq: 1
              Content-Type: application/sdp
              Content-Length: xxx
```

v=0

```
o= 2890842807 IN IP4 192.168.0.1
m=video 0 RTP/AVP 26
a=control:rtsp://192.168.0.1/video
a=x-onvif-track:VIDEO001
m=audio 0 RTP/AVP 98
a=control:rtsp://192.168.0.1/audio
a=x-onvif-track:AUDIO001
```

21.2 RTP header extension

In order to allow clients to report a stable and accurate timestamp for each frame played back regardless of the direction of playback, it is necessary to associate an absolute timestamp with each packet, or each group of packets with the same RTP timestamp (e.g. a video frame). This is achieved using an RTP header extension containing an NTP timestamp and some additional information also useful for replay.

The replay mechanism uses the extension ID 0xABAC for the replay extension.

Table 289 shows the general form of an RTP packet containing this extension.

Table 289 – RTP packet layout

V=2	P	X=1	CC	M	PT	sequence number
timestamp						
synchronization source (SSRC) identifier						
0xABAC					length=3	
NTP timestamp...						
...NTP timestamp						
C	E	D	mbz		CSeq	padding
payload...						

The fields of this extension are as follows:

- NTP timestamp. An NTP [RFC 1305] timestamp indicating the absolute UTC time associated with the access unit;
- C: 1 bit. Indicates that this access unit is a synchronization point or “clean point”, e.g. the start of an intra-coded frame in the case of video streams;
- E: 1 bit. Indicates the end of a contiguous section of recording. The last access unit in each track before a recording gap, or at the end of available footage, shall have this bit set. When replaying in reverse, the E flag shall be set on the last frame at the end of the contiguous section of recording;
- D: 1 bit. Indicates that this access unit follows a discontinuity in transmission. It is primarily used during reverse replay; the first packet of each GOP has the D bit set since it does not chronologically follow the previous packet in the data stream (see 21.5);
- mbz: This field is reserved for future use and shall be zero;
- CSeq: 1 byte. This is the low-order byte of the CSeq value used in the RTSP PLAY command that was used to initiate transmission. When a client sends multiple, consecutive PLAY commands, this value may be used to determine where the data from each new PLAY command begins.

The replay header extension shall be present in the first packet of every access unit (e.g. video frame). It MAY NOT be present in subsequent packets of an access unit.

21.2.1 NTP timestamps

The NTP timestamps in the RTP extension header shall increase monotonically over successive packets within a single RTP stream. They should correspond to wallclock time as measured at the original transmitter of the stream, adjusted if necessary to preserve monotonicity.

21.2.2 Compatibility with the JPEG header extension

The replay header extension may co-exist with the header extension used by the JPEG RTP profile; this is necessary to allow replay of JPEG streams that use this extension. The JPEG extension is simply appended to the replay extension; its presence is indicated by an RTP header extension length field with a value greater than 3, and by the extension start codes of 0xFFD8 or 0xFFFF at the start of the fourth word of the extension content.

The following Table 290 illustrates a JPEG packet that uses both extensions:

Table 290 – RTP packet with JPEG header layout

V=2	P	X=1	CC	M	PT	sequence number
timestamp						
synchronization source (SSRC) identifier						
0xABAC						length=N+3
NTP timestamp...						
...NTP timestamp						
C	E	D	mbz	CSeq	padding	
0xFFD8						jpeglength=N
extension payload: sequence of additional JPEG marker segments padded with 0xFF to the total extension length						
payload...						

21.3 RTSP feature tag

The Replay Service uses the “onvif-replay” feature tag to indicate that it supports the RTSP extensions described in this standard. This allows clients to query the server's support for these extensions using the Require header as described in [RFC 2326], 12.3.1.

Example:

```

C->S:  SETUP rtsp://server.com/foo/bar/baz.rm
RTSP/1.0
      CSeq: 302
      Require: onvif-replay

S->C:  RTSP/1.0 551 Option not supported
      CSeq: 302
      Unsupported: onvif-replay

```

The Replay Server shall accept a SETUP command that includes a Require header containing the onvif-replay feature tag.

21.4 Initiating Playback

Playback is initiated by means of the RTSP PLAY method. For example:

```

PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Require: onvif-replay
Range: clock=20090615T114900.440Z-

```

Rate-Control: no

ONVIF devices MAY support reverse playback. Reverse playback is indicated using the Scale header field with a negative value. For example to play in reverse without no data loss a value of $-1,0$ would be used.

```
PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Require: onvif-replay
Range: clock=20090615T114900.440Z-
Rate-Control: no
Scale: -1.0
```

If a device supports reverse playback it shall accept a Scale header with a value of $-1,0$. A device MAY accept other values for the Scale parameter. Unless the Rate-Control header is set to “no” (see below), the Scale parameter is used in the manner described in [RFC 2326]. If Rate-Control is set to “no”, the Scale parameter, if it is present, shall be either $1,0$ or $-1,0$, to indicate forward or reverse playback respectively. If it is not present, forward playback is assumed.

21.4.1 Range header field

The Range field shall be expressed using absolute times only; the other formats defined by [RFC 2326] shall NOT be used by ONVIF replay clients. Servers may choose to support other formats also. Absolute times are expressed using the *utc-range* from [RFC 2326].

Either open or closed ranges may be used. In the case of a closed range, the range is increasing (end time later than start time) for forward playback and decreasing for reverse playback. The direction of the range shall correspond to the value of the Scale header.

In all cases, the first point of the range indicates the starting point for replay.

Examples:

```
PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Require: onvif-replay
Range: clock=20090615T114900.440Z-20090615T115000
Rate-Control: no
```

```
PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Require: onvif-replay
Range: clock=20090615T115000.440Z-20090615T114900
Rate-Control: no
Scale: -1.0
```

21.4.2 Rate-Control header field

This specification introduces the Rate-Control header field, which may be either “yes” or “no”. If the field is not present, “yes” is assumed, and the stream is delivered in real time using

standard RTP timing mechanisms. If this field is “no”, the stream is delivered as fast as possible, using only the flow control provided by the transport to limit the delivery rate.

The important difference between these two modes is that with “Rate-Control=yes”, the server is in control of the playback speed, whereas with “Rate-Control=no” the client is in control of the playback speed. Rate-controlled replay will typically only be used by non-ONVIF specific clients as they will not specify “Rate-Control=no”.

When replaying multiple tracks of a single recording, started by a single RTSP PLAY command and not using rate-control, the data from the tracks should be multiplexed in time in the same order as they were recorded.

21.4.3 Frames header field

The Frames header field may be used to reduce the number of frames that are transmitted, for example to lower bandwidth or processing load. Three modes are possible:

1. intra frames only. This is indicated using the value “intra”, optionally followed by a minimum interval between successive intra frames in the stream. The latter can be used to limit the number of frames received even in the presence of “I-frame storms” caused by many receivers requesting frequent I-frames.
2. intra frames and predicted frames only. This is indicated using the value “predicted”. This value can be used to eliminate B-frames if the stream includes them.
3. all frames. This is the default.

Examples:

To request intra frames only:

Frames: intra

To request intra frames with a minimum interval of 4 000 ms:

Frames: intra/4000

To request intra frames and predicted frames only:

Frames: predicted

To request all frames (note that it is not necessary to explicitly specify this mode but the example is included for completeness):

Frames: all

The interval argument used with the “intra” option refers to the recording timeline, not playback time; thus for any given interval the same frames are played regardless of playback speed. The interval argument shall NOT be present unless the Frames option is “intra”.

The server shall support the Frames header field. This does not preclude the use of the Scale header field as an alternative means of limiting the data rate. The implementation of the Scale header field may vary between different server implementations, as stated by [RFC 2326].

21.4.4 Synchronization points

The transmitted video stream shall begin at a synchronization point (see 11.18). The rules for choosing the starting frame are as follows:

- if the requested start time is within a section of recorded footage, the stream starts with the first clean point at or before the requested start time. This is the case regardless of playback direction;
- if the requested start time is within a gap in recorded footage and playback is being initiated in the forwards direction, the stream starts with the first clean point in the section following the requested start time;
- if the requested start time is within a gap in recorded footage and playback is being initiated in the reverse direction, the stream starts with the last clean point in the section preceding the requested start time.

21.5 Reverse replay

Reverse replay is initiated using the Scale header field with a negative value as described above.

21.5.1 Packet transmission order

The order in which video packets are transmitted during reverse replay is based on GOPs, where a GOP consists of a clean point followed by a sequence of non-cleanpoint packets.

During reverse playback, GOPs are sent in reverse order, but packets within a GOP are sent in forward order. The first packet of each GOP shall have the “discontinuity” bit set in its RTP extension header. The last packet of a GOP immediately following a gap (or the beginning of available footage) shall have the E bit set in its RTP extension header.

When transmitting only key frames, or when the codec is not motion-based (e.g. JPEG), a GOP is considered to consist of a single frame, but may still be composed of multiple packets. In this case the packets within each frame are again sent in forward order, while the frames themselves are sent in reverse order.

Audio and metadata streams MAY be transmitted in an order mirroring that of the video stream. Thus packets from these streams are sent in forward playback order until the occurrence of a packet (generally a video packet) with the D bit set in the extension header, at which point they jump back to a point before the discontinuity.

21.5.2 RTP sequence numbers

The RTP sequence numbers of packets transmitted during reverse playback shall increment monotonically *in the order of delivery*, not in the intended order of playback.

21.5.3 RTP timestamps

The use of RTP timestamps depends on the value of the Rate-Control header. If the value of this header is “no” (i.e. the client controls playback speed), the RTP timestamps are derived from the original sampling times of the recorded frames. If the Rate-Control header is not present or has the value “yes” (i.e. the server controls playback speed), the RTP timestamps correspond to playback timing as described in [RFC 2326] Annex B.

If Rate-Control is “no”, the RTP timestamps of packets transmitted during reverse playback shall be the same as they would be if those same packets were being transmitted in the forwards direction. Unlike the sequence numbers, the RTP timestamps correspond to the original recording order, not the delivery order. The server MAY use the same RTP timestamps that were originally received when the stream was recorded.

This means that successive RTP packets of a single GOP will always have increasing RTP timestamps (see transmission order above), but that the timestamp on index frames of successively received GOPs will decrease during reverse replay.

If Rate-Control is “yes”, the RTP timestamps of packets transmitted during reverse playback shall indicate the times at which each frame should be rendered at the client. Thus successive packets of a single GOP will have *decreasing* RTP timestamps (since the first one delivered should be played last), and the timestamps on index frames will *increase*. In this mode the interval between successive timestamps depends on the values of the Speed and Scale headers, as described in [RFC 2326] Annex B.

21.6 RTSP keepalive

When rate control is disabled and the RTP stream is tunneled through the RTSP connection (i.e. using the RTP/RTSP/TCP or RTP/RTSP/HTTP/TCP transports), the client shall not send SET_PARAMETER requests and the server shall not time out the connection in the absence of these requests. This is because the client may be unable to receive the responses to these requests, for example if replay is paused.

On the other hand, either the server or client may enable TCP keepalive on the connection in order to determine if the other endpoint has become unresponsive.

21.7 Currently recording footage

If the client commences playback from the current real world time or shortly before it, it can end up playing footage in real time as it is being recorded. In this event the server simply continues to send stream data to the client as it receives it.

Note that the E bit is not set on access units currently being recorded even though each access unit sent to the replay client will typically be the last one known to the server. If recording stops however, the E bit is set on the last access unit of the recording.

21.8 End of footage

If playback reaches a point after which there is no further data in one or more of the streams being sent, it stops transmitting data but does not enter the “paused” state. If the server resumes recording after this has happened, delivery will resume with the new data as it is received.

21.9 Go to time

As stated in [RFC 2326], 10.5, a PLAY command received when replay is already in progress will not take effect until the existing play operation has completed. This specification adds a new RTSP header, “Immediate”, which overrides this behaviour for the PLAY command that it is used with:

```
PLAY rtsp://192.168.0.1/path/to/recording RTSP/1.0
CSeq: 123
Session: 12345678
Require: onvif-replay
Range: clock=20090615T114900.440Z-
Rate-Control: no
Immediate: yes
```

If the server receives a PLAY command with the Immediate header set to “yes”, it will immediately start playing from the new location, cancelling any existing PLAY command. The first packet sent from the new location shall have the D (discontinuity) bit set in its RTP extension header.

21.10 Use of RTCP

A server is not required to send RTCP packets. If it does send them, the following rules apply:

If Rate Control is enabled (see 21.4.2), RTCP packets shall be constructed and transmitted as specified in [RFC 3550]. In particular, the NTP timestamp in a sender report indicates the current wallclock time, and is not related to the NTP timestamps embedded in the RTP extension headers in the data streams.

If Rate Control is not enabled, both the NTP timestamp and RTP timestamp in each sender report shall be set to zero.

21.11 Replay service commands

This subclause describes the web service commands provided by the Replay Service.

21.11.1 Request replay URI

GetReplayUri requests a URI that can be used to initiate playback of a recorded stream using RTSP as the control protocol. The URI is valid only as it is specified in the response. All implementations of the Replay Service shall support the GetReplayUri command (see Table 291).

Table 291 – GetReplayUri command

GetReplayUri		Request-Response
Message name	Description	
GetReplayUriRequest	<i>The StreamSetup element contains two parts. StreamType defines if a unicast or multicast media stream is requested. Transport specifies a chain of transport protocols defining the tunnelling of the media stream over different network protocols.</i> <i>The RecordingToken element indicates the recording to be streamed.</i> tt:StreamSetup StreamSetup [1][1] tt:ReferenceToken RecordingToken [1][1]	
GetReplayUriResponse	<i>Contains the Uri to be used for requesting the media stream.</i> xs:anyURI Uri [1][1]	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>The recording does not exist.</i>	
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	<i>Specification of StreamType or Transport part in StreamSetup is not supported.</i>	
env:Sender ter:OperationProhibited ter:StreamConflict	<i>Specification of StreamType or Transport part in StreamSetup causes conflict with other streams.</i>	

21.11.2 ReplayConfiguration

The ReplayConfiguration structure contains the configuration of the replay service (see Table 292). The fields in the ReplayConfiguration structure are:

SessionTimeout: An RTSP session has a keep-alive time. It shall be refreshed regularly to prevent the session from timing out. If the session times out, it shall be torn down. The session timeout for replay follows the same rules as applies for live streaming using the media service and as discussed in Clause 4.

21.11.3 SetReplayConfiguration

SetReplayConfiguration changes the configuration of the replay service. The replay service shall allow its configuration to be changed using this command.

Table 292 – SetReplayConfiguration command

SetReplayConfiguration		Request-Response
Message name	Description	
SetReplayConfigurationRequest	<i>The Configuration shall hold the new configuration for the replay service.</i> tt:ReplayConfiguration Configuration[1][1]	
SetReplayConfigurationResponse	<i>This shall be the empty message</i>	
Fault codes	Description	
env:Sender ter:InvalidArgVal ter:ConfigModify	<i>The values in the configuration cannot be set.</i>	

21.11.4 GetReplayConfiguration

GetReplayConfiguration returns the current configuration of the replay service. The replay service shall allow its configuration to be retrieved using this command (see Table 293).

Table 293 – GetReplayConfiguration command

GetReplayConfiguration		Request-Response
Message name	Description	
GetReplayConfigurationRequest	<i>This shall be an empty message.</i>	
GetReplayConfigurationResponse	<i>The Configuration shall holds the current configuration for the replay service.</i> tt:ReplayConfiguration Configuration[1][1]	
Fault codes	Description	
	<i>No command specific error codes.</i>	

21.11.5 Service specific fault codes

Table 294 lists the replay service-specific fault codes. In addition, each command can also generate a generic fault, see Table 6.

The specific faults are defined as sub code of a generic fault, see 5.11.2.1. The parent generic subcode is the *subcode* at the top of each row below and the specific fault *subcode* is at the bottom of the cell.

Table 294 – Replay service specific fault codes

Fault Code	Parent Subcode	Fault Reason	Description
	Subcode		
env:Sender	ter:InvalidArgVal	Profile token does not exist	The requested profile token ProfileToken does not exist.
	ter:NoProfile		
env:Sender	ter:InvalidArgVal	Invalid Stream setup	Specification of StreamType or Transport part in StreamSetup is not supported.
	ter:InvalidStreamSetup		
env:Sender	ter:OperationProhibited	Stream conflict	Specification of StreamType or Transport part in StreamSetup causes conflict with other streams.
	ter:StreamConflict		
env:Sender	ter:InvalidArgVal	Parameters cannot be set	The configuration parameters cannot be set.
	ter:ConfigModify		

22 Security

As is true for all network-oriented information technology, security is a most important subject for network video communication. The security threat depends on the application. While some applications are most vulnerable to network based attacks, other applications are not at all sensitive. The cost for implementing security countermeasures varies depending on the type of attacks intended to prevent. These facts imply that we cannot list general security requirements on the network video product or system, but can try to find a reasonable level of security requirements for devices conformant to this standard and to define basic security mechanism that allows building secure network video systems.

The current specification defines security mechanisms on two communication levels:

- transport level security,
- message level security.

This specification adopts port-based authentication mechanism as follows:

- IEEE 802.1X

22.1 Transport level security

Transport *level* security protects the data transfer between the client and the server. Transport Layer Security (TLS) is regarded as a mature standard for encrypted transport connections to provide a basic level of communication security. The TLS protocol allows the configuration of a mutually authenticated transport session as well as preserving the confidentiality and the integrity protected transport.

A device conformant to this specification should support TLS 1.0 [RFC 2246] and related specifications. The device should support TLS 1.1 [RFC 4346]. The device MAY support TLS 1.2 [RFC 5246].

A device should support TLS for protection of all of the ONVIF services it provides. A device also should support TLS for protection of media streams for the RTP/RTSP/HTTPS tunnel option as defined in Clause 11. This standard profiles a particular implementation of TLS and other relevant specifications that can be used with TLS.

A client should support TLS 1.0 [RFC 2246] and TLS 1.1 [RFC 4346]. The client MAY support TLS 1.2 [RFC 5246].

22.1.1 Supported cipher suites

A device that supports TLS shall support all of the following cipher suites [RFC 2246], [RFC 3268]:

- TLS_RSA_WITH_AES_128_CBC_SHA,
- TLS_RSA_WITH_NULL_SHA.

If a client supports TLS, then it shall support the following cipher suites:

- TLS_RSA_WITH_AES_128_CBC_SHA,
- TLS_RSA_WITH_NULL_SHA.

22.1.2 Server authentication

A device that supports TLS shall support server authentication using TLS. The device shall support processing of X.509 server certificates. The RSA key length shall be at least 1 024 bits.

A client should support server authentication using TLS.

This standard does not provide a full server certificate generation and Certificate Authority (CA) model. However, device management commands for device certificate retrieval and download are defined in 0.

The details of the server private key or keys secure bootstrapping mechanisms are *outside the scope* of the current standard. However, commands for *on board* key generation are defined in 0.

22.1.3 Client authentication

A device that supports TLS should support client authentication. Client authentication can be enabled/disabled with a device management command as described in 0

A device that supports TLS shall include the RSA certificate type (rsa_sign, for example) in the certificate request [RFC 2246] for client certificates, and shall support verification of the RSA client certificate and signature.

A client should support client authentication. If client authentication is supported, the client shall support RSA client certificate and signature and shall use an RSA key length of at least 1 024 bits.

The trusted CA bootstrapping mechanisms are *outside the scope* of the current standard. Future editions of the standard might define standardized bootstrapping mechanisms.

22.2 Message level security

TLS allows point-to-point confidentiality and integrity. Web Services, however, allow a more flexible communication pattern with intermediate nodes. In such situations TLS cannot provide end-to-end security. Furthermore, in order to implement user based access control on command level for Web Services, there is a need to verify the origin of each SOAP message. This can be provided through the WS-Security framework. ONVIF WS-security is profiled in 5.12.

22.3 IEEE 802.1X

IEEE 802.1X is an IEEE standard for port based network access control for the purpose of providing authentication and authorization of the devices attached to LAN ports. It makes use of the physical access characteristics of IEEE 802 LAN infrastructures in order to provide a means of authenticating and authorizing devices attached to a LAN port that has point-to-point connection characteristics, and of preventing access to that port in cases in which the authentication and authorization process fails.

This standard recommends the adoption of IEEE 802.1X for port based authentication for wireless networks. A device that supports IEEE 802.1X shall support EAP-PEAP/MSCHAPv2 type as a supported EAP method. The device MAY also support other EAP methods such as EAP-MD5, EAP-TLS and EAP-TTLS types.

This standard defines a set of commands to configure and manage the IEEE 802.1X configuration, please refer to 0.

Annex A (informative)

Notification topics

A.1 Media configuration topics

For the following entities of the Media Configuration, the video IP network interface

TopicNamespace provides the following topics:

```
tnsl:MediaConfiguration/Profile
tnsl:MediaConfiguration/VideoSourceConfiguration
tnsl:MediaConfiguration/AudioSourceConfiguration
tnsl:MediaConfiguration/VideoEncoderConfiguration
tnsl:MediaConfiguration/AudioEncoderConfiguration
tnsl:MediaConfiguration/VideoAnalyticsConfiguration
tnsl:MediaConfiguration/PTZConfiguration
tnsl:MediaConfiguration/MetaDataConfiguration
```

Each of these topics represents a property. A client subscribing to one of these topics will be notified about changes, creation and deletion of the corresponding entity.

The Message structures of the different topics are specified next using the MessageDescription Language introduced in 15.5.4.

A.1.1 Profile

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="ProfileToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItem Name="Config"
      Type="tt:Profile"/>
  </tt>Data>
</tt:MessageDescription>
```

A.1.2 VideoSourceConfiguration

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="VideoSourceConfigurationToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItem Name="Config"
      Type="tt:VideoSourceConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

A.1.3 AudioSourceConfiguration

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="AudioSourceConfigurationToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItem Name="Config"
      Type="tt:AudioSourceConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

A.1.4 VideoEncoderConfiguration

```
<tt:MessageDescription iIsProperty="true">
```

```
<tt:Source>
  <tt:SimpleItem Name="VideoEncoderConfigurationToken"
    Type="tt:ReferenceToken"/>
</tt:Source>
<tt:Data>
  <tt:ElementItem Name="Config"
    Type="tt:VideoEncoderConfiguration"/>
</tt:Data>
</tt:MessageDescription>
```

A.1.5 AudioEncoderConfiguration

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="AudioEncoderConfigurationToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:ElementItem Name="Config"
      Type="tt:AudioEncoderConfiguration"/>
  </tt:Data>
</tt:MessageDescription>
```

A.1.6 VideoAnalyticsConfiguration

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="VideoAnalyticsConfigurationToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:ElementItem Name="Config"
      Type="tt:VideoAnalyticsConfiguration"/>
  </tt:Data>
</tt:MessageDescription>
```

A.1.7 PTZConfiguration

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="PTZConfigurationToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:ElementItem Name="Config"
      Type="tt:PTZConfiguration"/>
  </tt:Data>
</tt:MessageDescription>
```

A.1.8 MetaDataConfiguration

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="MetaDataConfigurationToken"
      Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:ElementItem Name="Config"
      Type="tt:MetaDataConfiguration"/>
  </tt:Data>
</tt:MessageDescription>
```

A.1.9 Device management topics

The Device Topic contains the following Sub-topics defined in the video IP network interface TopicNamespace:

```
tns1:Device/Trigger/Relay
tns1:Device/OperationMode/ShutdownInitiated
tns1:Device/OperationMode/UploadInitiated
```

```

tns1:Device/HardwareFailure/FanFailure
tns1:Device/HardwareFailure/PowerSupplyFailure
tns1:Device/HardwareFailure/StorageFailure
tns1:Device/HardwareFailure/TemperatureCritical

```

Only the Relay defines a message payload. The other topics reply with an empty message.

A.1.10 Relay

```

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItem Name="RelayToken" Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItem Name="LogicalState" Type="tt:RelayLogicalState"/>
  </tt>Data>
</tt:MessageDescription>

```

A.1.11 PTZ Controller Topics

The PTZ service specifies handling of PTZ presets. Since the move operations are non-blocking, an NVC is not informed when the PTZ preset has been reached. Therefore, the following events are introduced which inform subscribers about the status of preset movements.

```

tns1:PTZController/PTZPresets/Invoked
tns1:PTZController/PTZPresets/Reached
tns1:PTZController/PTZPresets/Aborted
tns1:PTZController/PTZPresets/Left

```

The typical sequence of events is that first a NVC requests a certain Preset. When the NVT accepts this request, it will send out an Invoked event. The Invoked event has to follow either a Reached event or an Aborted event. The former is used when dome was able to reach the invoked preset position, the latter in any other case. A Reached event has to follow a Left event, as soon as the dome moves away from the preset position.

The Message structure of these events is given by the following Message Description (see Clause 12):

```

<tt:MessageDescription>
  <tt:Source>
    <tt:SimpleItem Name="PTZConfigurationToken"
                  Type="tt:ReferenceToken"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItem Name="PresetToken" Type="tt:ReferenceToken"/>
    <tt:SimpleItem Name="PresetName" Type="tt:Name"/>
  </tt>Data>
</tt:MessageDescription>

```

A.2 (Void)

Annex B (informative)

Scene descriptions

B.1 Colour Descriptor

A Colour Descriptor is defined as an optional element of the Appearance Node of an Object Node. The Colour Descriptor is defined by a list of Colour Clusters, each consisting of a Colour Value, an optional weight and an optional covariance matrix. The Colour Descriptor does not specify, how the Colour Clusters are created. They can represent bins of a colour histogram or the result of a clustering algorithm.

Colours are represented by three-dimensional vectors. Additionally, the colourspace of each colour vector can be specified by a colourspace attribute. If the colourspace attribute is missing, the YCbCr colourspace is assumed. It refers to the 'sRGB' gamut with the RGB to YCbCr transformation as of ISO/IEC 10918-1, a.k.a. JPEG. The Colourspace URI for the YCbCr colourspace is www.onvif.org/ver10/colourspace/YCbCr.

```
<xs:complexType name="ColorDescriptor">
  <xs:sequence>
    <xs:element name="ColorCluster" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Color" type="tt:ColorType"/>
          <xs:element name="Weight" type="xs:float" minOccurs="0"/>
          <xs:element name="Covariance" type="tt:ColorCovariance"
minOccurs="0"/>
          ...
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Color">
  <xs:attribute name="X" type="xs:float" use="required"/>
  <xs:attribute name="Y" type="xs:float" use="required"/>
  <xs:attribute name="Z" type="xs:float" use="required" />
  <xs:attribute name="Colorspace" type="xs:anyURI"/>
</xs:complexType>

<xs:complexType name="ColorCovariance">
  <xs:attribute name="XX" type="xs:float" use="required"/>
  <xs:attribute name="YY" type="xs:float" use="required"/>
  <xs:attribute name="ZZ" type="xs:float" use="required" />
  <xs:attribute name="XY" type="xs:float"/>
  <xs:attribute name="XZ" type="xs:float"/>
  <xs:attribute name="YZ" type="xs:float" />
  <xs:attribute name="Colorspace" type="xs:anyURI"/>
</xs:complexType>
```

B.2 Class descriptor

A Class Descriptor is defined as an optional element of the Appearance Node of an Object Node. The Class Descriptor is defined by a list of object classes together with a likelihood that the corresponding object belongs to this class. The sum of the likelihoods shall not exceed 1.

```
<xs:simpleType name="ClassType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Animal"/>
    <xs:enumeration value="Face"/>
```

```
<xs:enumeration value="Human"/>
<xs:enumeration value="Vehicle"/>
<xs:enumeration value="Other"/>
</xs:restriction>
</xs:simpleType>

<xs:complexType name="ClassDescriptor">
  <xs:sequence>
    <xs:element name="ClassCandidate" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Type" type="tt:ClassType"/>
          <xs:element name="Likelihood" type="xs:float"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

Withdrawing
IECNORM.COM: Click to view the full PDF of IEC 62676-2-3:2013

Annex C (normative)

Video IP network interface XML schemata

C.1 Video analytics service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:tns="http://www.onvif.org/ver20/analytics/wsdl"
  targetNamespace="http://www.onvif.org/ver20/analytics/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tns="http://www.onvif.org/ver10/schema"
      targetNamespace="http://www.onvif.org/ver20/analytics/wsdl"
      elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
        schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/>
    </xs:schema>
  </wsdl:types>

  <!-- Message Request/Responses elements --><!--=====
  <xs:element name="GetSupportedRules">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tns:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetSupportedRulesResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="SupportedRules" type="tns:SupportedRules"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element><!--=====
  <xs:element name="CreateRules">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tns:ReferenceToken"/>
        <xs:element name="Rule" type="tns:Config" minOccurs="1"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="CreateRulesResponse">
    <xs:complexType/>
  </xs:element><!--=====
  <xs:element name="DeleteRules">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tns:ReferenceToken"/>
        <xs:element name="RuleName" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="DeleteRulesResponse">
    <xs:complexType/>
  </xs:element><!--=====

```

```

<xs:element name="ModifyRules">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
      <xs:element name="Rule" type="tt:Config" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ModifyRulesResponse">
  <xs:complexType/>
</xs:element><!--=====-->
<xs:element name="GetRules">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetRulesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Rule" type="tt:Config" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetSupportedAnalyticsModules">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetSupportedAnalyticsModulesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SupportedAnalyticsModules"
type="tt:SupportedAnalyticsModules"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="CreateAnalyticsModules">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
      <xs:element name="AnalyticsModule" type="tt:Config"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="CreateAnalyticsModulesResponse">
  <xs:complexType/>
</xs:element><!--=====-->
<xs:element name="DeleteAnalyticsModules">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
      <xs:element name="AnalyticsModuleName" type="xs:string"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

```

```

</xs:element>
<xs:element name="DeleteAnalyticsModulesResponse">
  <xs:complexType/>
</xs:element><!--=====-->
<xs:element name="ModifyAnalyticsModules">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
      <xs:element name="AnalyticsModule" type="tt:Config"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ModifyAnalyticsModulesResponse">
  <xs:complexType/>
</xs:element><!--=====-->
<xs:element name="GetAnalyticsModules">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsModulesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AnalyticsModule" type="tt:Config" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
</xs:schema>
</wsdl:types>
<wsdl:message name="GetSupportedRulesRequest">
  <wsdl:part name="parameters" element="tan:GetSupportedRules"/>
</wsdl:message>
<wsdl:message name="GetSupportedRulesResponse">
  <wsdl:part name="parameters" element="tan:GetSupportedRulesResponse"/>
</wsdl:message>
<wsdl:message name="CreateRulesRequest">
  <wsdl:part name="parameters" element="tan:CreateRules"/>
</wsdl:message>
<wsdl:message name="CreateRulesResponse">
  <wsdl:part name="parameters" element="tan:CreateRulesResponse"/>
</wsdl:message>
<wsdl:message name="DeleteRulesRequest">
  <wsdl:part name="parameters" element="tan:DeleteRules"/>
</wsdl:message>
<wsdl:message name="DeleteRulesResponse">
  <wsdl:part name="parameters" element="tan:DeleteRulesResponse"/>
</wsdl:message>
<wsdl:message name="GetRulesRequest">
  <wsdl:part name="parameters" element="tan:GetRules"/>
</wsdl:message>
<wsdl:message name="GetRulesResponse">
  <wsdl:part name="parameters" element="tan:GetRulesResponse"/>
</wsdl:message>
<wsdl:message name="GetSupportedAnalyticsModulesResponse">
  <wsdl:part name="parameters"
element="tan:GetSupportedAnalyticsModulesResponse"/>
</wsdl:message>
<wsdl:message name="GetSupportedAnalyticsModulesRequest">

```

```

    <wsdl:part name="parameters" element="tan:GetSupportedAnalyticsModules"/>
  </wsdl:message>
  <wsdl:message name="CreateAnalyticsModulesRequest">
    <wsdl:part name="parameters" element="tan:CreateAnalyticsModules"/>
  </wsdl:message>
  <wsdl:message name="CreateAnalyticsModulesResponse">
    <wsdl:part name="parameters" element="tan:CreateAnalyticsModulesResponse"/>
  </wsdl:message>
  <wsdl:message name="DeleteAnalyticsModulesRequest">
    <wsdl:part name="parameters" element="tan:DeleteAnalyticsModules"/>
  </wsdl:message>
  <wsdl:message name="DeleteAnalyticsModulesResponse">
    <wsdl:part name="parameters" element="tan:DeleteAnalyticsModulesResponse"/>
  </wsdl:message>
  <wsdl:message name="GetAnalyticsModulesRequest">
    <wsdl:part name="parameters" element="tan:GetAnalyticsModules"/>
  </wsdl:message>
  <wsdl:message name="GetAnalyticsModulesResponse">
    <wsdl:part name="parameters" element="tan:GetAnalyticsModulesResponse"/>
  </wsdl:message>
  <wsdl:message name="ModifyRulesRequest">
    <wsdl:part name="parameters" element="tan:ModifyRules"/>
  </wsdl:message>
  <wsdl:message name="ModifyRulesResponse">
    <wsdl:part name="parameters" element="tan:ModifyRulesResponse"/>
  </wsdl:message>
  <wsdl:message name="ModifyAnalyticsModulesRequest">
    <wsdl:part name="parameters" element="tan:ModifyAnalyticsModules"/>
  </wsdl:message>
  <wsdl:message name="ModifyAnalyticsModulesResponse">
    <wsdl:part name="parameters" element="tan:ModifyAnalyticsModulesResponse"/>
  </wsdl:message>
  <wsdl:portType name="RuleEnginePort">
    <wsdl:operation name="GetSupportedRules">
      <wsdl:input message="tan:GetSupportedRulesRequest"/>
      <wsdl:output message="tan:GetSupportedRulesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreateRules">
      <wsdl:input message="tan:CreateRulesRequest"/>
      <wsdl:output message="tan:CreateRulesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="DeleteRules">
      <wsdl:input message="tan:DeleteRulesRequest"/>
      <wsdl:output message="tan:DeleteRulesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetRules">
      <wsdl:input message="tan:GetRulesRequest"/>
      <wsdl:output message="tan:GetRulesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="ModifyRules">
      <wsdl:input message="tan:ModifyRulesRequest"/>
      <wsdl:output message="tan:ModifyRulesResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:portType name="AnalyticsEnginePort">
    <wsdl:operation name="GetSupportedAnalyticsModules">
      <wsdl:input message="tan:GetSupportedAnalyticsModulesRequest"/>
      <wsdl:output message="tan:GetSupportedAnalyticsModulesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreateAnalyticsModules">
      <wsdl:input message="tan:CreateAnalyticsModulesRequest"/>
      <wsdl:output message="tan:CreateAnalyticsModulesResponse"/>
    </wsdl:operation>
  </wsdl:portType>

```



```

</wsdl:operation>
<wsdl:operation name="DeleteAnalyticsModules">
  <wsdl:input message="tan:DeleteAnalyticsModulesRequest"/>
  <wsdl:output message="tan:DeleteAnalyticsModulesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAnalyticsModules">
  <wsdl:input message="tan:GetAnalyticsModulesRequest"/>
  <wsdl:output message="tan:GetAnalyticsModulesResponse"/>
</wsdl:operation>
<wsdl:operation name="ModifyAnalyticsModules">
  <wsdl:input message="tan:ModifyAnalyticsModulesRequest"/>
  <wsdl:output message="tan:ModifyAnalyticsModulesResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="RuleEngineBinding" type="tan:RuleEnginePort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetSupportedRules">
    <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/GetSupportedRules"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="CreateRules">
    <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/CreateRules"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="DeleteRules">
    <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/DeleteRules"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetRules">
    <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/GetRules"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="ModifyRules">
    <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/ModifyRules"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
  </wsdl:operation>

```

```

        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
    </wsdl:binding>
    <wsdl:binding name="AnalyticsEngineBinding" type="tan:AnalyticsEnginePort">
      <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
      <wsdl:operation name="GetSupportedAnalyticsModules">
        <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/GetSupportedAnalyticsModules"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="CreateAnalyticsModules">
        <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/CreateAnalyticsModules"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="DeleteAnalyticsModules">
        <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/DeleteAnalyticsModules"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="GetAnalyticsModules">
        <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/GetAnalyticsModules"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="ModifyAnalyticsModules">
        <soap:operation
soapAction="http://www.onvif.org/ver20/analytics/wsdl/ModifyAnalyticsModules"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
    </wsdl:binding>
  </wsdl:definitions>

```


C.2 Video analytics device WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:tad="http://www.onvif.org/ver10/analyticsdevice/wsdl"
  targetNamespace="http://www.onvif.org/ver10/analyticsdevice/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
      targetNamespace="http://www.onvif.org/ver10/analyticsdevice/wsdl"
      elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
        schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/>
      <!--=====-->
      <xs:element name="DeleteAnalyticsEngineControl">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="DeleteAnalyticsEngineControlResponse">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element>
      <xs:element name="CreateAnalyticsEngineInputs">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Configuration" type="tt:AnalyticsEngineInput" minOccurs="1"
maxOccurs="unbounded"/>
            <xs:element name="ForcePersistence" type="xs:boolean" minOccurs="1"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="CreateAnalyticsEngineInputsResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Configuration" type="tt:AnalyticsEngineInput" minOccurs="1"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="CreateAnalyticsEngineControl">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Configuration" type="tt:AnalyticsEngineControl"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="CreateAnalyticsEngineControlResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Configuration" type="tt:AnalyticsEngineInput" minOccurs="1"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SetAnalyticsEngineControl">
        <xs:complexType>

```

```

    <xs:sequence>
      <xs:element name="Configuration" type="tt:AnalyticsEngineControl"/>
      <xs:element name="ForcePersistence" type="xs:boolean"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetAnalyticsEngineControlResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEngineControl">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEngineControlResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configuration" type="tt:AnalyticsEngineControl"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEngineControls">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEngineControlsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AnalyticsEngineControls" type="tt:AnalyticsEngineControl"
minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEngine">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEngineResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configuration" type="tt:AnalyticsEngine"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEngines">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetAnalyticsEnginesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configuration" type="tt:AnalyticsEngine" minOccurs="1"
maxOccurs="unbounded"/>

```

```

        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetVideoAnalyticsConfiguration">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Configuration" type="tt:VideoAnalyticsConfiguration"/>
          <xs:element name="ForcePersistence" type="xs:boolean"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetVideoAnalyticsConfigurationResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetAnalyticsEngineInput">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Configuration" type="tt:AnalyticsEngineInput"/>
          <xs:element name="ForcePersistence" type="xs:boolean"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetAnalyticsEngineInputResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAnalyticsEngineInput">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAnalyticsEngineInputResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Configuration" type="tt:AnalyticsEngineInput"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAnalyticsEngineInputs">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAnalyticsEngineInputsResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Configuration" type="tt:AnalyticsEngineInput" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAnalyticsDeviceStreamUri">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="StreamSetup" type="tt:StreamSetup"/>
          <xs:element name="AnalyticsEngineControlToken" type="tt:ReferenceToken"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>

```

```

    </xs:complexType>
  </xs:element>
  <xs:element name="GetAnalyticsDeviceStreamUriResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Uri" type="xs:anyURI"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetVideoAnalyticsConfiguration">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetVideoAnalyticsConfigurationResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Configuration" type="tt:VideoAnalyticsConfiguration"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="DeleteAnalyticsEngineInputs">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="DeleteAnalyticsEngineInputsResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetAnalyticsState">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="AnalyticsEngineControlToken" type="tt:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetAnalyticsStateResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="State" type="tt:AnalyticsStateInformation"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema><!--=====-->
</wsdl:types>
<wsdl:message name="DeleteAnalyticsEngineControlRequest">
  <wsdl:part name="parameters" element="tad:DeleteAnalyticsEngineControl"/>
</wsdl:message>
<wsdl:message name="DeleteAnalyticsEngineControlResponse">
  <wsdl:part name="parameters" element="tad:DeleteAnalyticsEngineControlResponse"/>
</wsdl:message>
<wsdl:message name="CreateAnalyticsEngineControlRequest">
  <wsdl:part name="parameters" element="tad:CreateAnalyticsEngineControl"/>
</wsdl:message>
<wsdl:message name="CreateAnalyticsEngineControlResponse">

```

```

    <wsdl:part name="parameters" element="tad:CreateAnalyticsEngineControlResponse"/>
</wsdl:message>
<wsdl:message name="SetAnalyticsEngineControlRequest">
    <wsdl:part name="parameters" element="tad:SetAnalyticsEngineControl"/>
</wsdl:message>
<wsdl:message name="SetAnalyticsEngineControlResponse">
    <wsdl:part name="parameters" element="tad:SetAnalyticsEngineControlResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineControlRequest">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineControl"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineControlResponse">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineControlResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineControlsRequest">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineControls"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineControlsResponse">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineControlsResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineRequest">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngine"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineResponse">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEnginesRequest">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngines"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEnginesResponse">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEnginesResponse"/>
</wsdl:message>
<wsdl:message name="SetVideoAnalyticsConfigurationRequest">
    <wsdl:part name="parameters" element="tad:SetVideoAnalyticsConfiguration"/>
</wsdl:message>
<wsdl:message name="SetVideoAnalyticsConfigurationResponse">
    <wsdl:part name="parameters" element="tad:SetVideoAnalyticsConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetAnalyticsEngineInputRequest">
    <wsdl:part name="parameters" element="tad:SetAnalyticsEngineInput"/>
</wsdl:message>
<wsdl:message name="SetAnalyticsEngineInputResponse">
    <wsdl:part name="parameters" element="tad:SetAnalyticsEngineInputResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineInputRequest">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineInput"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineInputResponse">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineInputResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineInputsRequest">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineInputs"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsEngineInputsResponse">
    <wsdl:part name="parameters" element="tad:GetAnalyticsEngineInputsResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsDeviceStreamUriRequest">
    <wsdl:part name="parameters" element="tad:GetAnalyticsDeviceStreamUri"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsDeviceStreamUriResponse">
    <wsdl:part name="parameters" element="tad:GetAnalyticsDeviceStreamUriResponse"/>
</wsdl:message>

```



```

<wsdl:message name="GetVideoAnalyticsConfigurationRequest">
  <wsdl:part name="parameters" element="tad:GetVideoAnalyticsConfiguration"/>
</wsdl:message>
<wsdl:message name="GetVideoAnalyticsConfigurationResponse">
  <wsdl:part name="parameters" element="tad:GetVideoAnalyticsConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="CreateAnalyticsEngineInputsRequest">
  <wsdl:part name="parameters" element="tad:CreateAnalyticsEngineInputs"/>
</wsdl:message>
<wsdl:message name="CreateAnalyticsEngineInputsResponse">
  <wsdl:part name="parameters" element="tad:CreateAnalyticsEngineInputsResponse"/>
</wsdl:message>
<wsdl:message name="DeleteAnalyticsEngineInputsRequest">
  <wsdl:part name="parameters" element="tad:DeleteAnalyticsEngineInputs"/>
</wsdl:message>
<wsdl:message name="DeleteAnalyticsEngineInputsResponse">
  <wsdl:part name="parameters" element="tad:DeleteAnalyticsEngineInputsResponse"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsStateRequest">
  <wsdl:part name="parameters" element="tad:GetAnalyticsState"/>
</wsdl:message>
<wsdl:message name="GetAnalyticsStateResponse">
  <wsdl:part name="parameters" element="tad:GetAnalyticsStateResponse"/>
</wsdl:message>
<wsdl:portType name="AnalyticsDevicePort">
  <wsdl:operation name="DeleteAnalyticsEngineControl">
    <wsdl:input message="tad:DeleteAnalyticsEngineControlRequest"/>
    <wsdl:output message="tad:DeleteAnalyticsEngineControlResponse"/>
  </wsdl:operation>
  <wsdl:operation name="CreateAnalyticsEngineControl">
    <wsdl:input message="tad:CreateAnalyticsEngineControlRequest"/>
    <wsdl:output message="tad:CreateAnalyticsEngineControlResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SetAnalyticsEngineControl">
    <wsdl:input message="tad:SetAnalyticsEngineControlRequest"/>
    <wsdl:output message="tad:SetAnalyticsEngineControlResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAnalyticsEngineControl">
    <wsdl:input message="tad:GetAnalyticsEngineControlRequest"/>
    <wsdl:output message="tad:GetAnalyticsEngineControlResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAnalyticsEngineControls">
    <wsdl:input message="tad:GetAnalyticsEngineControlsRequest"/>
    <wsdl:output message="tad:GetAnalyticsEngineControlsResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAnalyticsEngine">
    <wsdl:input message="tad:GetAnalyticsEngineRequest"/>
    <wsdl:output message="tad:GetAnalyticsEngineResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAnalyticsEngines">
    <wsdl:input message="tad:GetAnalyticsEnginesRequest"/>
    <wsdl:output message="tad:GetAnalyticsEnginesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SetVideoAnalyticsConfiguration">
    <wsdl:input message="tad:SetVideoAnalyticsConfigurationRequest"/>
    <wsdl:output message="tad:SetVideoAnalyticsConfigurationResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SetAnalyticsEngineInput">
    <wsdl:input message="tad:SetAnalyticsEngineInputRequest"/>
    <wsdl:output message="tad:SetAnalyticsEngineInputResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAnalyticsEngineInput">

```

```

        <wsdl:input message="tad:GetAnalyticsEngineInputRequest"/>
        <wsdl:output message="tad:GetAnalyticsEngineInputResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetAnalyticsEngineInputs">
        <wsdl:input message="tad:GetAnalyticsEngineInputsRequest"/>
        <wsdl:output message="tad:GetAnalyticsEngineInputsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetAnalyticsDeviceStreamUri">
        <wsdl:input message="tad:GetAnalyticsDeviceStreamUriRequest"/>
        <wsdl:output message="tad:GetAnalyticsDeviceStreamUriResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetVideoAnalyticsConfiguration">
        <wsdl:input message="tad:GetVideoAnalyticsConfigurationRequest"/>
        <wsdl:output message="tad:GetVideoAnalyticsConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreateAnalyticsEngineInputs">
        <wsdl:input message="tad:CreateAnalyticsEngineInputsRequest"/>
        <wsdl:output message="tad:CreateAnalyticsEngineInputsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="DeleteAnalyticsEngineInputs">
        <wsdl:input message="tad:DeleteAnalyticsEngineInputsRequest"/>
        <wsdl:output message="tad:DeleteAnalyticsEngineInputsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetAnalyticsState">
        <wsdl:input message="tad:GetAnalyticsStateRequest"/>
        <wsdl:output message="tad:GetAnalyticsStateResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="AnalyticsDeviceBinding" type="tad:AnalyticsDevicePort">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="DeleteAnalyticsEngineControl">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/DeleteAnalyticsEngineControl"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateAnalyticsEngineControl">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/CreateAnalyticsEngineControl"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetAnalyticsEngineControl">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/SetAnalyticsEngineControl"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAnalyticsEngineControl">
        <soap:operation

```

```

soapAction="http://www.onvif.org/ver10/analyticsdevice/wsd/GetAnalyticsEngineControl"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAnalyticsEngineControls">
  <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsd/GetAnalyticsEngineControls"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAnalyticsEngine">
  <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsd/GetAnalyticsEngine"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAnalyticsEngines">
  <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsd/GetAnalyticsEngines"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetVideoAnalyticsConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsd/SetVideoAnalyticsConfiguration"
/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetAnalyticsEngineInput">
  <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsd/SetAnalyticsEngineInput"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAnalyticsEngineInput">
  <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsd/GetAnalyticsEngineInput"/>

```



```

        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAnalyticsEngineInputs">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/GetAnalyticsEngineInputs"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAnalyticsDeviceStreamUri">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/GetAnalyticsDeviceStreamUri"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetVideoAnalyticsConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/GetVideoAnalyticsConfiguration"
/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateAnalyticsEngineInputs">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/CreateAnalyticsEngineInputs"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DeleteAnalyticsEngineInputs">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/DeleteAnalyticsEngineInputs"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAnalyticsState">
        <soap:operation
soapAction="http://www.onvif.org/ver10/analyticsdevice/wsdl/GetAnalyticsState"/>
        <wsdl:input>

```

```

        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

C.3 Device IO service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:trt="http://www.onvif.org/ver10/media/wsdl"
    xmlns:tds="http://www.onvif.org/ver10/device/wsdl"
    xmlns:tmd="http://www.onvif.org/ver10/deviceIO/wsdl"
    targetNamespace="http://www.onvif.org/ver10/deviceIO/wsdl">
    <wsdl:import namespace="http://www.onvif.org/ver10/media/wsdl"
        location="http://www.onvif.org/onvif/ver10/media/wsdl/media.wsdl"/>
    <wsdl:import namespace="http://www.onvif.org/ver10/device/wsdl"
        location="http://www.onvif.org/onvif/ver10/device/wsdl/device.wsdl"/>
    <wsdl:import namespace="http://www.onvif.org/ver10/device/wsdl/devicegmt.wsdl"
        location="http://www.onvif.org/onvif/ver10/device/wsdl/devicegmt.wsdl"/>
    <wsdl:types>
        <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
            targetNamespace="http://www.onvif.org/ver10/deviceIO/wsdl"
            elementFormDefault="qualified">
            <xs:import namespace="http://www.onvif.org/ver10/schema"
                schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!--
=====
            <xs:element name="GetVideoOutputs">
                <xs:complexType>
                    <xs:sequence/>
                </xs:complexType>
            </xs:element>
            <xs:element name="GetVideoOutputsResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="VideoOutputs" type="tt:VideoOutput" minOccurs="0"
maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element><!--=====
            <xs:element name="GetAudioSourceConfiguration">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="AudioSourceToken" type="tt:ReferenceToken"/>
                        <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="GetAudioSourceConfigurationResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="AudioSourceConfiguration"
type="tt:AudioSourceConfiguration"/>
                        <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>

```

```

</xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioOutputConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioOutputToken" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioOutputConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioOutputConfiguration"
type="tt:AudioOutputConfiguration"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetVideoSourceConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetVideoSourceConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoSourceConfiguration"
type="tt:VideoSourceConfiguration"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetVideoOutputConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoOutputToken" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetVideoOutputConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoOutputConfiguration"
type="tt:VideoOutputConfiguration"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetAudioSourceConfiguration">
  <xs:complexType>
    <xs:sequence>

```

```

        <xs:element name="Configuration" type="tt:AudioSourceConfiguration"/>
        <xs:element name="ForcePersistence" type="xs:boolean"/>
        <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="SetAudioSourceConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<!--=====-->
<xs:element name="SetAudioOutputConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:AudioOutputConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
            <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetAudioOutputConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<!--=====-->
<xs:element name="SetVideoSourceConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:VideoSourceConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
            <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetVideoSourceConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<!--=====-->
<xs:element name="SetVideoOutputConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:VideoOutputConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
            <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetVideoOutputConfigurationResponse">

```

```

<xs:complexType>
  <xs:sequence>
    <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
  </xs:sequence>
</xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetVideoSourceConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetVideoSourceConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoSourceConfigurationOptions"
type="tt:VideoSourceConfigurationOptions"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetVideoOutputConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoOutputToken" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetVideoOutputConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoOutputConfigurationOptions"
type="tt:VideoOutputConfigurationOptions"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioSourceConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioSourceToken" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioSourceConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioSourceOptions"
type="tt:AudioSourceConfigurationOptions"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```



```

</xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioOutputConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioOutputToken" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioOutputConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioOutputOptions"
type="tt:AudioOutputConfigurationOptions"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRelayOutputSettings">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RelayOutput" type="tt:RelayOutput"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRelayOutputSettingsResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
</xs:schema>
</wsdl:types>
<wsdl:message name="GetVideoOutputsRequest">
  <wsdl:part name="parameters" element="tmd:GetVideoOutputs"/>
</wsdl:message>
<wsdl:message name="GetVideoOutputsResponse">
  <wsdl:part name="parameters" element="tmd:GetVideoOutputsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputsRequest">
  <wsdl:part name="parameters" element="trt:GetAudioOutputs"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputsResponse">
  <wsdl:part name="parameters" element="trt:GetAudioOutputsResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoSourcesRequest">
  <wsdl:part name="parameters" element="trt:GetVideoSources"/>
</wsdl:message>
<wsdl:message name="GetVideoSourcesResponse">
  <wsdl:part name="parameters" element="trt:GetVideoSourcesResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioSourcesRequest">
  <wsdl:part name="parameters" element="trt:GetAudioSources"/>
</wsdl:message>
<wsdl:message name="GetAudioSourcesResponse">
  <wsdl:part name="parameters" element="trt:GetAudioSourcesResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:GetVideoSourceConfiguration"/>
</wsdl:message>

```

```

<wsdl:message name="GetVideoSourceConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:GetVideoSourceConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoOutputConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:GetVideoOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="GetVideoOutputConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:GetVideoOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:GetAudioSourceConfiguration"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:GetAudioSourceConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:GetAudioOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:GetAudioOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetVideoSourceConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:SetVideoSourceConfiguration"/>
</wsdl:message>
<wsdl:message name="SetVideoSourceConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:SetVideoSourceConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetVideoOutputConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:SetVideoOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="SetVideoOutputConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:SetVideoOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetAudioSourceConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:SetAudioSourceConfiguration"/>
</wsdl:message>
<wsdl:message name="SetAudioSourceConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:SetAudioSourceConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetAudioOutputConfigurationRequest">
  <wsdl:part name="parameters" element="tmd:SetAudioOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="SetAudioOutputConfigurationResponse">
  <wsdl:part name="parameters" element="tmd:SetAudioOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationOptionsRequest">
  <wsdl:part name="parameters" element="tmd:GetVideoSourceConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationOptionsResponse">
  <wsdl:part name="parameters"
element="tmd:GetVideoSourceConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoOutputConfigurationOptionsRequest">
  <wsdl:part name="parameters" element="tmd:GetVideoOutputConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetVideoOutputConfigurationOptionsResponse">
  <wsdl:part name="parameters"
element="tmd:GetVideoOutputConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationOptionsRequest">
  <wsdl:part name="parameters" element="tmd:GetAudioSourceConfigurationOptions"/>
</wsdl:message>

```

```

<wsdl:message name="GetAudioSourceConfigurationOptionsResponse">
  <wsdl:part name="parameters"
element="tmd:GetAudioSourceConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationOptionsRequest">
  <wsdl:part name="parameters" element="tmd:GetAudioOutputConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationOptionsResponse">
  <wsdl:part name="parameters"
element="tmd:GetAudioOutputConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetRelayOutputsRequest">
  <wsdl:part name="parameters" element="tds:GetRelayOutputs"/>
</wsdl:message>
<wsdl:message name="GetRelayOutputsResponse">
  <wsdl:part name="parameters" element="tds:GetRelayOutputsResponse"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputSettingsRequest">
  <wsdl:part name="parameters" element="tmd:SetRelayOutputSettings"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputSettingsResponse">
  <wsdl:part name="parameters" element="tmd:SetRelayOutputSettingsResponse"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputStateRequest">
  <wsdl:part name="parameters" element="tds:SetRelayOutputState"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputStateResponse">
  <wsdl:part name="parameters" element="tds:SetRelayOutputStateResponse"/>
</wsdl:message>
<wsdl:portType name="DeviceIOPort">
  <wsdl:operation name="GetAudioSources">
    <wsdl:input message="tmd:GetAudioSourcesRequest"/>
    <wsdl:output message="tmd:GetAudioSourcesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAudioOutputs">
    <wsdl:input message="tmd:GetAudioOutputsRequest"/>
    <wsdl:output message="tmd:GetAudioOutputsResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetVideoSources">
    <wsdl:input message="tmd:GetVideoSourcesRequest"/>
    <wsdl:output message="tmd:GetVideoSourcesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetVideoOutputs">
    <wsdl:input message="tmd:GetVideoOutputsRequest"/>
    <wsdl:output message="tmd:GetVideoOutputsResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetVideoSourceConfiguration">
    <wsdl:input message="tmd:GetVideoSourceConfigurationRequest"/>
    <wsdl:output message="tmd:GetVideoSourceConfigurationResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetVideoOutputConfiguration">
    <wsdl:input message="tmd:GetVideoOutputConfigurationRequest"/>
    <wsdl:output message="tmd:GetVideoOutputConfigurationResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAudioSourceConfiguration">
    <wsdl:input message="tmd:GetAudioSourceConfigurationRequest"/>
    <wsdl:output message="tmd:GetAudioSourceConfigurationResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAudioOutputConfiguration">
    <wsdl:input message="tmd:GetAudioOutputConfigurationRequest"/>
    <wsdl:output message="tmd:GetAudioOutputConfigurationResponse"/>
  </wsdl:operation>

```



```

<wsdl:operation name="SetVideoSourceConfiguration">
  <wsdl:input message="tmd:SetVideoSourceConfigurationRequest"/>
  <wsdl:output message="tmd:SetVideoSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetVideoOutputConfiguration">
  <wsdl:input message="tmd:SetVideoOutputConfigurationRequest"/>
  <wsdl:output message="tmd:SetVideoOutputConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetAudioSourceConfiguration">
  <wsdl:input message="tmd:SetAudioSourceConfigurationRequest"/>
  <wsdl:output message="tmd:SetAudioSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetAudioOutputConfiguration">
  <wsdl:input message="tmd:SetAudioOutputConfigurationRequest"/>
  <wsdl:output message="tmd:SetAudioOutputConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoSourceConfigurationOptions">
  <wsdl:input message="tmd:GetVideoSourceConfigurationOptionsRequest"/>
  <wsdl:output message="tmd:GetVideoSourceConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoOutputConfigurationOptions">
  <wsdl:input message="tmd:GetVideoOutputConfigurationOptionsRequest"/>
  <wsdl:output message="tmd:GetVideoOutputConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioSourceConfigurationOptions">
  <wsdl:input message="tmd:GetAudioSourceConfigurationOptionsRequest"/>
  <wsdl:output message="tmd:GetAudioSourceConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioOutputConfigurationOptions">
  <wsdl:input message="tmd:GetAudioOutputConfigurationOptionsRequest"/>
  <wsdl:output message="tmd:GetAudioOutputConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetRelayOutputs">
  <wsdl:input message="tmd:GetRelayOutputsRequest"/>
  <wsdl:output message="tmd:GetRelayOutputsResponse"/>
</wsdl:operation>
<wsdl:operation name="SetRelayOutputSettings">
  <wsdl:input message="tmd:SetRelayOutputSettingsRequest"/>
  <wsdl:output message="tmd:SetRelayOutputSettingsResponse"/>
</wsdl:operation>
<wsdl:operation name="SetRelayOutputState">
  <wsdl:input message="tmd:SetRelayOutputStateRequest"/>
  <wsdl:output message="tmd:SetRelayOutputStateResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="DeviceIOBinding" type="tmd:DeviceIOPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetAudioSources">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetAudioSources"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetAudioOutputs">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetAudioOutputs"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
  </wsdl:operation>

```

```

        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetVideoSources">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetVideoSources"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetVideoOutputs">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetVideoOutputs"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetVideoSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetVideoSourceConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetVideoOutputConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetVideoOutputConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetAudioSourceConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioOutputConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetAudioOutputConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>

```

```

        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="SetVideoSourceConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/SetVideoSourceConfiguration"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetVideoOutputConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/SetVideoOutputConfiguration"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetAudioSourceConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/SetAudioSourceConfiguration"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetAudioOutputConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/SetAudioOutputConfiguration"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="GetVideoSourceConfigurationOptions">
    <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetVideoSourceConfigurationOptions"/
>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetVideoOutputConfigurationOptions">
    <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetVideoOutputConfigurationOptions"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>

```

```

        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="GetAudioSourceConfigurationOptions">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetAudioSourceConfigurationOptions"/
>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="GetAudioOutputConfigurationOptions">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetAudioOutputConfigurationOptions"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="GetRelayOutputs">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/GetRelayOutputs"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="SetRelayOutputSettings">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/SetRelayOutputSettings"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="SetRelayOutputState">
        <soap:operation
soapAction="http://www.onvif.org/ver10/deviceio/wsdl/SetRelayOutputState"/>
        <wsdl:input>
          <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
    </wsdl:binding>
  </wsdl:definitions>

```

C.4 Device management service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
  <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

```

```

xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tds="http://www.onvif.org/ver10/device/wsdl"
targetNamespace="http://www.onvif.org/ver10/device/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
targetNamespace="http://www.onvif.org/ver10/device/wsdl"
elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/>
      <!-- Message Request/Responses elements --><!--
      =====>
      <xs:element name="GetDeviceInformation">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetDeviceInformationResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Manufacturer" type="xs:string"/>
            <xs:element name="Model" type="xs:string"/>
            <xs:element name="FirmwareVersion" type="xs:string"/>
            <xs:element name="SerialNumber" type="xs:string"/>
            <xs:element name="HardwareId" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====-->
      <xs:element name="SetSystemDateAndTime">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="DateTimeType" type="tt:SetDateTimeType"/>
            <xs:element name="DaylightSavings" type="xs:boolean"/>
            <xs:element name="TimeZone" type="tt:TimeZone" minOccurs="0"/>
            <xs:element name="UTCDateTime" type="tt:DateTime" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SetSystemDateAndTimeResponse">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element><!--=====-->
      <xs:element name="GetSystemDateAndTime">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetSystemDateAndTimeResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="SystemDateAndTime" type="tt:SystemDateTime"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====-->
      <xs:element name="SetSystemFactoryDefault">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="FactoryDefault" type="tt:FactoryDefaultType"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

```

```

<xs:element name="SetSystemFactoryDefaultResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="UpgradeSystemFirmware">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Firmware" type="tt:AttachmentData"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="UpgradeSystemFirmwareResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Message" type="xs:string" minOccurs="0"
maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SystemReboot">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="SystemRebootResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Message" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="RestoreSystem">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="BackupFiles" type="tt:BackupFile" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RestoreSystemResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetSystemBackup">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetSystemBackupResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="BackupFiles" type="tt:BackupFile" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetSystemSupportInformation">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>

```



```

</xs:element>
<xs:element name="GetSystemSupportInformationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SupportInformation" type="tt:SupportInformation"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetSystemLog">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="LogType" type="tt:SystemLogType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetSystemLogResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SystemLog" type="tt:SystemLog"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetScopes">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetScopesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Scopes" type="tt:Scope" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetScopes">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Scopes" type="xs:anyURI" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetScopesResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="AddScopes">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ScopeItem" type="xs:anyURI" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddScopesResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="RemoveScopes">

```

```

    <xs:complexType>
      <xs:sequence>
        <xs:element name="ScopeItem" type="xs:anyURI" minOccurs="1"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="RemoveScopesResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ScopeItem" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetDiscoveryMode">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetDiscoveryModeResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DiscoveryMode" type="tt:DiscoveryMode"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetDiscoveryMode">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DiscoveryMode" type="tt:DiscoveryMode"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetDiscoveryModeResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetRemoteDiscoveryMode">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetRemoteDiscoveryModeResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="RemoteDiscoveryMode" type="tt:DiscoveryMode"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetRemoteDiscoveryMode">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="RemoteDiscoveryMode" type="tt:DiscoveryMode"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetRemoteDiscoveryModeResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>

```



```

</xs:element><!--=====-->
<xs:element name="GetDPAddresses">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetDPAddressesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="DPAAddress" type="tt:NetworkHost" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetDPAddresses">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="DPAAddress" type="tt:NetworkHost" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetDPAddressesResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetEndpointReference">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetEndpointReferenceResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="GUID" type="xs:string"/>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetRemoteUser">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRemoteUserResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RemoteUser" type="tt:RemoteUser" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetRemoteUser">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RemoteUser" type="tt:RemoteUser" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRemoteUserResponse">
  <xs:complexType>

```

```

        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetUsers">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetUsersResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="User" type="tt:User" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="CreateUsers">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="User" type="tt:User" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="CreateUsersResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="DeleteUsers">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Username" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="DeleteUsersResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="SetUser">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="User" type="tt:User" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetUserResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetWsdlUrl">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetWsdlUrlResponse">
      <xs:complexType>

```

```

        <xs:sequence>
            <xs:element name="WsdlUrl" type="xs:anyURI"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCapabilities">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Category" type="tt:CapabilityCategory" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetCapabilitiesResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Capabilities" type="tt:Capabilities"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetHostname">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetHostnameResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="HostnameInformation" type="tt:HostnameInformation"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetHostname">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Name" type="xs:token"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetHostnameResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetDNS">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetDNSResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="DNSInformation" type="tt:DNSInformation"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetDNS">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="FromDHCP" type="xs:boolean"/>
            <xs:element name="SearchDomain" type="xs:token" minOccurs="0"
maxOccurs="unbounded"/>

```

```

        <xs:element name="DNSManual" type="tt:IPAddress" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="SetDNSResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetNTP">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetNTPResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="NTPInformation" type="tt:NTPInformation"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetNTP">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="FromDHCP" type="xs:boolean"/>
            <xs:element name="NTPManual" type="tt:NetworkHost" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetNTPResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetDynamicDNS">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetDynamicDNSResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="DynamicDNSInformation"
type="tt:DynamicDNSInformation"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetDynamicDNS">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Type" type="tt:DynamicDNSType"/>
            <xs:element name="Name" type="tt:DNSName" minOccurs="0"/>
            <xs:element name="TTL" type="xs:duration" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetDynamicDNSResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>

```

```

</xs:element><!--=====-->
<xs:element name="GetNetworkInterfaces">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetNetworkInterfacesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="NetworkInterfaces" type="tt:NetworkInterface"
minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetNetworkInterfaces">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="InterfaceToken" type="tt:ReferenceToken"/>
      <xs:element name="NetworkInterface"
type="tt:NetworkInterfaceSetConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetNetworkInterfacesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RebootNeeded" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetNetworkProtocols">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetNetworkProtocolsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="NetworkProtocols" type="tt:NetworkProtocol"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetNetworkProtocols">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="NetworkProtocols" type="tt:NetworkProtocol"
minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetNetworkProtocolsResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetNetworkDefaultGateway">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>

```

```

<xs:element name="GetNetworkDefaultGatewayResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="NetworkGateway" type="tt:NetworkGateway"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetNetworkDefaultGateway">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="IPv4Address" type="tt:IPv4Address" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="IPv6Address" type="tt:IPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetNetworkDefaultGatewayResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetZeroConfiguration">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetZeroConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ZeroConfiguration"
type="tt:NetworkZeroConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetZeroConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="InterfaceToken" type="tt:ReferenceToken"/>
      <xs:element name="Enabled" type="xs:boolean"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetZeroConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetIPAddressFilter">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetIPAddressFilterResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="IPAddressFilter" type="tt:IPAddressFilter"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetIPAddressFilter">
  <xs:complexType>

```

```

        <xs:sequence>
          <xs:element name="IPAddressFilter" type="tt:IPAddressFilter"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetIPAddressFilterResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="AddIPAddressFilter">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="IPAddressFilter" type="tt:IPAddressFilter"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="AddIPAddressFilterResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="RemoveIPAddressFilter">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="IPAddressFilter" type="tt:IPAddressFilter"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="RemoveIPAddressFilterResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetAccessPolicy">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAccessPolicyResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="PolicyFile" type="tt:BinaryData"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="SetAccessPolicy">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="PolicyFile" type="tt:BinaryData"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetAccessPolicyResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="CreateCertificate">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="CertificateID" type="xs:token" minOccurs="0"/>

```



```

        <xs:element name="Subject" type="xs:string" minOccurs="0"/>
        <xs:element name="ValidNotBefore" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="ValidNotAfter" type="xs:dateTime" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="CreateCertificateResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="NvtCertificate" type="tt:Certificate"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<!--=====-->
<xs:element name="GetCertificates">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetCertificatesResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="NvtCertificate" type="tt:Certificate" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<!--=====-->
<xs:element name="GetCertificatesStatus">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetCertificatesStatusResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="CertificateStatus" type="tt:CertificateStatus"
minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<!--=====-->
<xs:element name="SetCertificatesStatus">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="CertificateStatus" type="tt:CertificateStatus"
minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetCertificatesStatusResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<!--=====-->
<xs:element name="DeleteCertificates">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="CertificateID" type="xs:token" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="DeleteCertificatesResponse">
    <xs:complexType>

```



```

        </xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetPkcs10Request">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="CertificateID" type="xs:token"/>
            <xs:element name="Subject" type="xs:string" minOccurs="0"/>
            <xs:element name="Attributes" type="tt:BinaryData" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetPkcs10RequestResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Pkcs10Request" type="tt:BinaryData"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="LoadCertificates">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="NVTCertificate" type="tt:Certificate" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="LoadCertificatesResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetClientCertificateMode">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetClientCertificateModeResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Enabled" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetClientCertificateMode">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Enabled" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetClientCertificateModeResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCACertificates">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetCACertificatesResponse">

```

```

<xs:complexType>
  <xs:sequence>
    <xs:element name="CACertificate" type="tt:Certificate" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
</xs:element><!--=====-->
<xs:element name="LoadCertificateWithPrivateKey">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CertificateWithPrivateKey"
type="tt:CertificateWithPrivateKey" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="LoadCertificateWithPrivateKeyResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCertificateInformation">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CertificateID" type="xs:token" minOccurs="1"
maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetCertificateInformationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CertificateInformation" type="tt:CertificateInformation"
minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="LoadCACertificates">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CACertificate" type="tt:Certificate" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="LoadCACertificatesResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="CreateDot1XConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Dot1XConfiguration" type="tt:Dot1XConfiguration"
minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="CreateDot1XConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->

```

```

<xs:element name="SetDot1XConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Dot1XConfiguration" type="tt:Dot1XConfiguration"
minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetDot1XConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetDot1XConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Dot1XConfigurationToken" type="tt:ReferenceToken"
minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetDot1XConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Dot1XConfiguration" type="tt:Dot1XConfiguration"
minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetDot1XConfigurations">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetDot1XConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Dot1XConfiguration" type="tt:Dot1XConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="DeleteDot1XConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Dot1XConfigurationToken" type="tt:ReferenceToken"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="DeleteDot1XConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetRelayOutputs">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRelayOutputsResponse">
  <xs:complexType>

```

```

        <xs:sequence>
          <xs:element name="RelayOutputs" type="tt:RelayOutput" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="SetRelayOutputSettings">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="RelayOutputToken" type="tt:ReferenceToken"/>
          <xs:element name="Properties" type="tt:RelayOutputSettings"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetRelayOutputSettingsResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="SetRelayOutputState">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="RelayOutputToken" type="tt:ReferenceToken"
minOccurs="1" maxOccurs="1"/>
          <xs:element name="LogicalState" type="tt:RelayLogicalState" minOccurs="1"
maxOccurs="1"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetRelayOutputStateResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="SendAuxiliaryCommand">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="AuxiliaryCommand" type="tt:AuxiliaryData"
minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SendAuxiliaryCommandResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="AuxiliaryCommandResponse" type="tt:AuxiliaryData"
minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetDot11Capabilities">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetDot11CapabilitiesResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Capabilities" type="tt:Dot11Capabilities"/>

```

```

        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetDot11Status">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="InterfaceToken" type="tt:ReferenceToken"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetDot11StatusResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Status" type="tt:Dot11Status"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="ScanAvailableDot11Networks">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="InterfaceToken" type="tt:ReferenceToken"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="ScanAvailableDot11NetworksResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Networks" type="tt:Dot11AvailableNetworks"
minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetSystemUri">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetSystemUriResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="SystemLogUri" type="tt:SystemLogUriList"
minOccurs="0" maxOccurs="1"/>
          <xs:element name="SupportInfoUri" type="xs:anyURI" minOccurs="0"
maxOccurs="1"/>
          <xs:element name="SystemBackupUri" type="xs:anyURI" minOccurs="0"
maxOccurs="1"/>
          <xs:element name="Extension" minOccurs="0">
            <xs:complexType>
              <xs:sequence>
                <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="StartFirmwareUpgrade">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>

```

```

<xs:element name="StartFirmwareUpgradeResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="UploadUri" type="xs:anyURI"/>
      <xs:element name="UploadDelay" type="xs:duration"/>
      <xs:element name="ExpectedDownTime" type="xs:duration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="StartSystemRestore">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="StartSystemRestoreResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="UploadUri" type="xs:anyURI"/>
      <xs:element name="ExpectedDownTime" type="xs:duration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
</wsdl:types>
<wsdl:message name="GetDeviceInformationRequest">
  <wsdl:part name="parameters" element="tds:GetDeviceInformation"/>
</wsdl:message>
<wsdl:message name="GetDeviceInformationResponse">
  <wsdl:part name="parameters" element="tds:GetDeviceInformationResponse"/>
</wsdl:message>
<wsdl:message name="SetSystemDateAndTimeRequest">
  <wsdl:part name="parameters" element="tds:SetSystemDateAndTime"/>
</wsdl:message>
<wsdl:message name="SetSystemDateAndTimeResponse">
  <wsdl:part name="parameters" element="tds:SetSystemDateAndTimeResponse"/>
</wsdl:message>
<wsdl:message name="GetSystemDateAndTimeRequest">
  <wsdl:part name="parameters" element="tds:GetSystemDateAndTime"/>
</wsdl:message>
<wsdl:message name="GetSystemDateAndTimeResponse">
  <wsdl:part name="parameters" element="tds:GetSystemDateAndTimeResponse"/>
</wsdl:message>
<wsdl:message name="SetSystemFactoryDefaultRequest">
  <wsdl:part name="parameters" element="tds:SetSystemFactoryDefault"/>
</wsdl:message>
<wsdl:message name="SetSystemFactoryDefaultResponse">
  <wsdl:part name="parameters" element="tds:SetSystemFactoryDefaultResponse"/>
</wsdl:message>
<wsdl:message name="UpgradeSystemFirmwareRequest">
  <wsdl:part name="parameters" element="tds:UpgradeSystemFirmware"/>
</wsdl:message>
<wsdl:message name="UpgradeSystemFirmwareResponse">
  <wsdl:part name="parameters" element="tds:UpgradeSystemFirmwareResponse"/>
</wsdl:message>
<wsdl:message name="SystemRebootRequest">
  <wsdl:part name="parameters" element="tds:SystemReboot"/>
</wsdl:message>
<wsdl:message name="SystemRebootResponse">
  <wsdl:part name="parameters" element="tds:SystemRebootResponse"/>
</wsdl:message>
<wsdl:message name="GetSystemBackupRequest">
  <wsdl:part name="parameters" element="tds:GetSystemBackup"/>

```



```

</wsdl:message>
<wsdl:message name="GetSystemBackupResponse">
  <wsdl:part name="parameters" element="tds:GetSystemBackupResponse"/>
</wsdl:message>
<wsdl:message name="RestoreSystemRequest">
  <wsdl:part name="parameters" element="tds:RestoreSystem"/>
</wsdl:message>
<wsdl:message name="RestoreSystemResponse">
  <wsdl:part name="parameters" element="tds:RestoreSystemResponse"/>
</wsdl:message>
<wsdl:message name="GetSystemSupportInformationRequest">
  <wsdl:part name="parameters" element="tds:GetSystemSupportInformation"/>
</wsdl:message>
<wsdl:message name="GetSystemSupportInformationResponse">
  <wsdl:part name="parameters"
element="tds:GetSystemSupportInformationResponse"/>
</wsdl:message>
<wsdl:message name="GetSystemLogRequest">
  <wsdl:part name="parameters" element="tds:GetSystemLog"/>
</wsdl:message>
<wsdl:message name="GetSystemLogResponse">
  <wsdl:part name="parameters" element="tds:GetSystemLogResponse"/>
</wsdl:message>
<wsdl:message name="GetScopesRequest">
  <wsdl:part name="parameters" element="tds:GetScopes"/>
</wsdl:message>
<wsdl:message name="GetScopesResponse">
  <wsdl:part name="parameters" element="tds:GetScopesResponse"/>
</wsdl:message>
<wsdl:message name="SetScopesRequest">
  <wsdl:part name="parameters" element="tds:SetScopes"/>
</wsdl:message>
<wsdl:message name="SetScopesResponse">
  <wsdl:part name="parameters" element="tds:SetScopesResponse"/>
</wsdl:message>
<wsdl:message name="AddScopesRequest">
  <wsdl:part name="parameters" element="tds:AddScopes"/>
</wsdl:message>
<wsdl:message name="AddScopesResponse">
  <wsdl:part name="parameters" element="tds:AddScopesResponse"/>
</wsdl:message>
<wsdl:message name="RemoveScopesRequest">
  <wsdl:part name="parameters" element="tds:RemoveScopes"/>
</wsdl:message>
<wsdl:message name="RemoveScopesResponse">
  <wsdl:part name="parameters" element="tds:RemoveScopesResponse"/>
</wsdl:message>
<wsdl:message name="GetDiscoveryModeRequest">
  <wsdl:part name="parameters" element="tds:GetDiscoveryMode"/>
</wsdl:message>
<wsdl:message name="GetDiscoveryModeResponse">
  <wsdl:part name="parameters" element="tds:GetDiscoveryModeResponse"/>
</wsdl:message>
<wsdl:message name="SetDiscoveryModeRequest">
  <wsdl:part name="parameters" element="tds:SetDiscoveryMode"/>
</wsdl:message>
<wsdl:message name="SetDiscoveryModeResponse">
  <wsdl:part name="parameters" element="tds:SetDiscoveryModeResponse"/>
</wsdl:message>
<wsdl:message name="GetRemoteDiscoveryModeRequest">
  <wsdl:part name="parameters" element="tds:GetRemoteDiscoveryMode"/>
</wsdl:message>

```

```

<wsdl:message name="GetRemoteDiscoveryModeResponse">
  <wsdl:part name="parameters" element="tds:GetRemoteDiscoveryModeResponse"/>
</wsdl:message>
<wsdl:message name="SetRemoteDiscoveryModeRequest">
  <wsdl:part name="parameters" element="tds:SetRemoteDiscoveryMode"/>
</wsdl:message>
<wsdl:message name="SetRemoteDiscoveryModeResponse">
  <wsdl:part name="parameters" element="tds:SetRemoteDiscoveryModeResponse"/>
</wsdl:message>
<wsdl:message name="GetDPAddressesRequest">
  <wsdl:part name="parameters" element="tds:GetDPAddresses"/>
</wsdl:message>
<wsdl:message name="GetDPAddressesResponse">
  <wsdl:part name="parameters" element="tds:GetDPAddressesResponse"/>
</wsdl:message>
<wsdl:message name="SetDPAddressesRequest">
  <wsdl:part name="parameters" element="tds:SetDPAddresses"/>
</wsdl:message>
<wsdl:message name="SetDPAddressesResponse">
  <wsdl:part name="parameters" element="tds:SetDPAddressesResponse"/>
</wsdl:message>
<wsdl:message name="GetEndpointReferenceRequest">
  <wsdl:part name="parameters" element="tds:GetEndpointReference"/>
</wsdl:message>
<wsdl:message name="GetEndpointReferenceResponse">
  <wsdl:part name="parameters" element="tds:GetEndpointReferenceResponse"/>
</wsdl:message>
<wsdl:message name="GetRemoteUserRequest">
  <wsdl:part name="parameters" element="tds:GetRemoteUser"/>
</wsdl:message>
<wsdl:message name="GetRemoteUserResponse">
  <wsdl:part name="parameters" element="tds:GetRemoteUserResponse"/>
</wsdl:message>
<wsdl:message name="SetRemoteUserRequest">
  <wsdl:part name="parameters" element="tds:SetRemoteUser"/>
</wsdl:message>
<wsdl:message name="SetRemoteUserResponse">
  <wsdl:part name="parameters" element="tds:SetRemoteUserResponse"/>
</wsdl:message>
<wsdl:message name="GetUsersRequest">
  <wsdl:part name="parameters" element="tds:GetUsers"/>
</wsdl:message>
<wsdl:message name="GetUsersResponse">
  <wsdl:part name="parameters" element="tds:GetUsersResponse"/>
</wsdl:message>
<wsdl:message name="CreateUsersRequest">
  <wsdl:part name="parameters" element="tds>CreateUsers"/>
</wsdl:message>
<wsdl:message name="CreateUsersResponse">
  <wsdl:part name="parameters" element="tds>CreateUsersResponse"/>
</wsdl:message>
<wsdl:message name="DeleteUsersRequest">
  <wsdl:part name="parameters" element="tds>DeleteUsers"/>
</wsdl:message>
<wsdl:message name="DeleteUsersResponse">
  <wsdl:part name="parameters" element="tds>DeleteUsersResponse"/>
</wsdl:message>
<wsdl:message name="SetUserRequest">
  <wsdl:part name="parameters" element="tds:SetUser"/>
</wsdl:message>
<wsdl:message name="SetUserResponse">
  <wsdl:part name="parameters" element="tds:SetUserResponse"/>

```



```

</wsdl:message>
<wsdl:message name="GetWsdUrlRequest">
  <wsdl:part name="parameters" element="tds:GetWsdUrl"/>
</wsdl:message>
<wsdl:message name="GetWsdUrlResponse">
  <wsdl:part name="parameters" element="tds:GetWsdUrlResponse"/>
</wsdl:message>
<wsdl:message name="GetCapabilitiesRequest">
  <wsdl:part name="parameters" element="tds:GetCapabilities"/>
</wsdl:message>
<wsdl:message name="GetCapabilitiesResponse">
  <wsdl:part name="parameters" element="tds:GetCapabilitiesResponse"/>
</wsdl:message>
<wsdl:message name="GetHostnameRequest">
  <wsdl:part name="parameters" element="tds:GetHostname"/>
</wsdl:message>
<wsdl:message name="GetHostnameResponse">
  <wsdl:part name="parameters" element="tds:GetHostnameResponse"/>
</wsdl:message>
<wsdl:message name="SetHostnameRequest">
  <wsdl:part name="parameters" element="tds:SetHostname"/>
</wsdl:message>
<wsdl:message name="SetHostnameResponse">
  <wsdl:part name="parameters" element="tds:SetHostnameResponse"/>
</wsdl:message>
<wsdl:message name="GetDNSRequest">
  <wsdl:part name="parameters" element="tds:GetDNS"/>
</wsdl:message>
<wsdl:message name="GetDNSResponse">
  <wsdl:part name="parameters" element="tds:GetDNSResponse"/>
</wsdl:message>
<wsdl:message name="SetDNSRequest">
  <wsdl:part name="parameters" element="tds:SetDNS"/>
</wsdl:message>
<wsdl:message name="SetDNSResponse">
  <wsdl:part name="parameters" element="tds:SetDNSResponse"/>
</wsdl:message>
<wsdl:message name="GetNTPRequest">
  <wsdl:part name="parameters" element="tds:GetNTP"/>
</wsdl:message>
<wsdl:message name="GetNTPResponse">
  <wsdl:part name="parameters" element="tds:GetNTPResponse"/>
</wsdl:message>
<wsdl:message name="SetNTPRequest">
  <wsdl:part name="parameters" element="tds:SetNTP"/>
</wsdl:message>
<wsdl:message name="SetNTPResponse">
  <wsdl:part name="parameters" element="tds:SetNTPResponse"/>
</wsdl:message>
<wsdl:message name="GetDynamicDNSRequest">
  <wsdl:part name="parameters" element="tds:GetDynamicDNS"/>
</wsdl:message>
<wsdl:message name="GetDynamicDNSResponse">
  <wsdl:part name="parameters" element="tds:GetDynamicDNSResponse"/>
</wsdl:message>
<wsdl:message name="SetDynamicDNSRequest">
  <wsdl:part name="parameters" element="tds:SetDynamicDNS"/>
</wsdl:message>
<wsdl:message name="SetDynamicDNSResponse">
  <wsdl:part name="parameters" element="tds:SetDynamicDNSResponse"/>
</wsdl:message>
<wsdl:message name="GetNetworkInterfacesRequest">

```

```

    <wsdl:part name="parameters" element="tds:GetNetworkInterfaces"/>
  </wsdl:message>
  <wsdl:message name="GetNetworkInterfacesResponse">
    <wsdl:part name="parameters" element="tds:GetNetworkInterfacesResponse"/>
  </wsdl:message>
  <wsdl:message name="SetNetworkInterfacesRequest">
    <wsdl:part name="parameters" element="tds:SetNetworkInterfaces"/>
  </wsdl:message>
  <wsdl:message name="SetNetworkInterfacesResponse">
    <wsdl:part name="parameters" element="tds:SetNetworkInterfacesResponse"/>
  </wsdl:message>
  <wsdl:message name="GetNetworkProtocolsRequest">
    <wsdl:part name="parameters" element="tds:GetNetworkProtocols"/>
  </wsdl:message>
  <wsdl:message name="GetNetworkProtocolsResponse">
    <wsdl:part name="parameters" element="tds:GetNetworkProtocolsResponse"/>
  </wsdl:message>
  <wsdl:message name="SetNetworkProtocolsRequest">
    <wsdl:part name="parameters" element="tds:SetNetworkProtocols"/>
  </wsdl:message>
  <wsdl:message name="SetNetworkProtocolsResponse">
    <wsdl:part name="parameters" element="tds:SetNetworkProtocolsResponse"/>
  </wsdl:message>
  <wsdl:message name="GetNetworkDefaultGatewayRequest">
    <wsdl:part name="parameters" element="tds:GetNetworkDefaultGateway"/>
  </wsdl:message>
  <wsdl:message name="GetNetworkDefaultGatewayResponse">
    <wsdl:part name="parameters"
element="tds:GetNetworkDefaultGatewayResponse"/>
  </wsdl:message>
  <wsdl:message name="SetNetworkDefaultGatewayRequest">
    <wsdl:part name="parameters" element="tds:SetNetworkDefaultGateway"/>
  </wsdl:message>
  <wsdl:message name="SetNetworkDefaultGatewayResponse">
    <wsdl:part name="parameters"
element="tds:SetNetworkDefaultGatewayResponse"/>
  </wsdl:message>
  <wsdl:message name="GetZeroConfigurationRequest">
    <wsdl:part name="parameters" element="tds:GetZeroConfiguration"/>
  </wsdl:message>
  <wsdl:message name="GetZeroConfigurationResponse">
    <wsdl:part name="parameters" element="tds:GetZeroConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="SetZeroConfigurationRequest">
    <wsdl:part name="parameters" element="tds:SetZeroConfiguration"/>
  </wsdl:message>
  <wsdl:message name="SetZeroConfigurationResponse">
    <wsdl:part name="parameters" element="tds:SetZeroConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="GetIPAddressFilterRequest">
    <wsdl:part name="parameters" element="tds:GetIPAddressFilter"/>
  </wsdl:message>
  <wsdl:message name="GetIPAddressFilterResponse">
    <wsdl:part name="parameters" element="tds:GetIPAddressFilterResponse"/>
  </wsdl:message>
  <wsdl:message name="SetIPAddressFilterRequest">
    <wsdl:part name="parameters" element="tds:SetIPAddressFilter"/>
  </wsdl:message>
  <wsdl:message name="SetIPAddressFilterResponse">
    <wsdl:part name="parameters" element="tds:SetIPAddressFilterResponse"/>
  </wsdl:message>
  <wsdl:message name="AddIPAddressFilterRequest">

```

```

    <wsdl:part name="parameters" element="tds:AddIPAddressFilter"/>
</wsdl:message>
<wsdl:message name="AddIPAddressFilterResponse">
    <wsdl:part name="parameters" element="tds:AddIPAddressFilterResponse"/>
</wsdl:message>
<wsdl:message name="RemoveIPAddressFilterRequest">
    <wsdl:part name="parameters" element="tds:RemoveIPAddressFilter"/>
</wsdl:message>
<wsdl:message name="RemoveIPAddressFilterResponse">
    <wsdl:part name="parameters" element="tds:RemoveIPAddressFilterResponse"/>
</wsdl:message>
<wsdl:message name="GetAccessPolicyRequest">
    <wsdl:part name="parameters" element="tds:GetAccessPolicy"/>
</wsdl:message>
<wsdl:message name="GetAccessPolicyResponse">
    <wsdl:part name="parameters" element="tds:GetAccessPolicyResponse"/>
</wsdl:message>
<wsdl:message name="SetAccessPolicyRequest">
    <wsdl:part name="parameters" element="tds:SetAccessPolicy"/>
</wsdl:message>
<wsdl:message name="SetAccessPolicyResponse">
    <wsdl:part name="parameters" element="tds:SetAccessPolicyResponse"/>
</wsdl:message>
<wsdl:message name="CreateCertificateRequest">
    <wsdl:part name="parameters" element="tds:CreateCertificate"/>
</wsdl:message>
<wsdl:message name="CreateCertificateResponse">
    <wsdl:part name="parameters" element="tds:CreateCertificateResponse"/>
</wsdl:message>
<wsdl:message name="GetCertificatesRequest">
    <wsdl:part name="parameters" element="tds:GetCertificates"/>
</wsdl:message>
<wsdl:message name="GetCertificatesResponse">
    <wsdl:part name="parameters" element="tds:GetCertificatesResponse"/>
</wsdl:message>
<wsdl:message name="GetCertificatesStatusRequest">
    <wsdl:part name="parameters" element="tds:GetCertificatesStatus"/>
</wsdl:message>
<wsdl:message name="GetCertificatesStatusResponse">
    <wsdl:part name="parameters" element="tds:GetCertificatesStatusResponse"/>
</wsdl:message>
<wsdl:message name="SetCertificatesStatusRequest">
    <wsdl:part name="parameters" element="tds:SetCertificatesStatus"/>
</wsdl:message>
<wsdl:message name="SetCertificatesStatusResponse">
    <wsdl:part name="parameters" element="tds:SetCertificatesStatusResponse"/>
</wsdl:message>
<wsdl:message name="DeleteCertificatesRequest">
    <wsdl:part name="parameters" element="tds>DeleteCertificates"/>
</wsdl:message>
<wsdl:message name="DeleteCertificatesResponse">
    <wsdl:part name="parameters" element="tds>DeleteCertificatesResponse"/>
</wsdl:message>
<wsdl:message name="GetPkcs10RequestRequest">
    <wsdl:part name="parameters" element="tds:GetPkcs10Request"/>
</wsdl:message>
<wsdl:message name="GetPkcs10RequestResponse">
    <wsdl:part name="parameters" element="tds:GetPkcs10RequestResponse"/>
</wsdl:message>
<wsdl:message name="LoadCertificatesRequest">
    <wsdl:part name="parameters" element="tds:LoadCertificates"/>
</wsdl:message>

```

```

<wsdl:message name="LoadCertificatesResponse">
  <wsdl:part name="parameters" element="tds:LoadCertificatesResponse"/>
</wsdl:message>
<wsdl:message name="GetClientCertificateModeRequest">
  <wsdl:part name="parameters" element="tds:GetClientCertificateMode"/>
</wsdl:message>
<wsdl:message name="GetClientCertificateModeResponse">
  <wsdl:part name="parameters" element="tds:GetClientCertificateModeResponse"/>
</wsdl:message>
<wsdl:message name="SetClientCertificateModeRequest">
  <wsdl:part name="parameters" element="tds:SetClientCertificateMode"/>
</wsdl:message>
<wsdl:message name="SetClientCertificateModeResponse">
  <wsdl:part name="parameters" element="tds:SetClientCertificateModeResponse"/>
</wsdl:message>
<wsdl:message name="GetRelayOutputsRequest">
  <wsdl:part name="parameters" element="tds:GetRelayOutputs"/>
</wsdl:message>
<wsdl:message name="GetRelayOutputsResponse">
  <wsdl:part name="parameters" element="tds:GetRelayOutputsResponse"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputSettingsRequest">
  <wsdl:part name="parameters" element="tds:SetRelayOutputSettings"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputSettingsResponse">
  <wsdl:part name="parameters" element="tds:SetRelayOutputSettingsResponse"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputStateRequest">
  <wsdl:part name="parameters" element="tds:SetRelayOutputState"/>
</wsdl:message>
<wsdl:message name="SetRelayOutputStateResponse">
  <wsdl:part name="parameters" element="tds:SetRelayOutputStateResponse"/>
</wsdl:message>
<wsdl:message name="SendAuxiliaryCommandRequest">
  <wsdl:part name="parameters" element="tds:SendAuxiliaryCommand"/>
</wsdl:message>
<wsdl:message name="SendAuxiliaryCommandResponse">
  <wsdl:part name="parameters" element="tds:SendAuxiliaryCommandResponse"/>
</wsdl:message>
<wsdl:message name="GetCACertificatesRequest">
  <wsdl:part name="parameters" element="tds:GetCACertificates"/>
</wsdl:message>
<wsdl:message name="GetCACertificatesResponse">
  <wsdl:part name="parameters" element="tds:GetCACertificatesResponse"/>
</wsdl:message>
<wsdl:message name="LoadCertificateWithPrivateKeyRequest">
  <wsdl:part name="parameters" element="tds:LoadCertificateWithPrivateKey"/>
</wsdl:message>
<wsdl:message name="LoadCertificateWithPrivateKeyResponse">
  <wsdl:part name="parameters"
element="tds:LoadCertificateWithPrivateKeyResponse"/>
</wsdl:message>
<wsdl:message name="GetCertificateInformationRequest">
  <wsdl:part name="parameters" element="tds:GetCertificateInformation"/>
</wsdl:message>
<wsdl:message name="GetCertificateInformationResponse">
  <wsdl:part name="parameters" element="tds:GetCertificateInformationResponse"/>
</wsdl:message>
<wsdl:message name="LoadCACertificatesRequest">
  <wsdl:part name="parameters" element="tds:LoadCACertificates"/>
</wsdl:message>
<wsdl:message name="LoadCACertificatesResponse">

```



```

    <wsdl:part name="parameters" element="tds:LoadCACertificatesResponse"/>
</wsdl:message>
<wsdl:message name="CreateDot1XConfigurationRequest">
    <wsdl:part name="parameters" element="tds:CreateDot1XConfiguration"/>
</wsdl:message>
<wsdl:message name="CreateDot1XConfigurationResponse">
    <wsdl:part name="parameters" element="tds:CreateDot1XConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetDot1XConfigurationRequest">
    <wsdl:part name="parameters" element="tds:SetDot1XConfiguration"/>
</wsdl:message>
<wsdl:message name="SetDot1XConfigurationResponse">
    <wsdl:part name="parameters" element="tds:SetDot1XConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetDot1XConfigurationRequest">
    <wsdl:part name="parameters" element="tds:GetDot1XConfiguration"/>
</wsdl:message>
<wsdl:message name="GetDot1XConfigurationResponse">
    <wsdl:part name="parameters" element="tds:GetDot1XConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetDot1XConfigurationsRequest">
    <wsdl:part name="parameters" element="tds:GetDot1XConfigurations"/>
</wsdl:message>
<wsdl:message name="GetDot1XConfigurationsResponse">
    <wsdl:part name="parameters" element="tds:GetDot1XConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="DeleteDot1XConfigurationRequest">
    <wsdl:part name="parameters" element="tds>DeleteDot1XConfiguration"/>
</wsdl:message>
<wsdl:message name="DeleteDot1XConfigurationResponse">
    <wsdl:part name="parameters" element="tds>DeleteDot1XConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetDot11CapabilitiesRequest">
    <wsdl:part name="parameters" element="tds:GetDot11Capabilities"/>
</wsdl:message>
<wsdl:message name="GetDot11CapabilitiesResponse">
    <wsdl:part name="parameters" element="tds:GetDot11CapabilitiesResponse"/>
</wsdl:message>
<wsdl:message name="GetDot11StatusRequest">
    <wsdl:part name="parameters" element="tds:GetDot11Status"/>
</wsdl:message>
<wsdl:message name="GetDot11StatusResponse">
    <wsdl:part name="parameters" element="tds:GetDot11StatusResponse"/>
</wsdl:message>
<wsdl:message name="ScanAvailableDot11NetworksRequest">
    <wsdl:part name="parameters" element="tds:ScanAvailableDot11Networks"/>
</wsdl:message>
<wsdl:message name="ScanAvailableDot11NetworksResponse">
    <wsdl:part name="parameters"
element="tds:ScanAvailableDot11NetworksResponse"/>
</wsdl:message>
<wsdl:message name="GetSystemUriRequest">
    <wsdl:part name="parameters" element="tds:GetSystemUri"/>
</wsdl:message>
<wsdl:message name="GetSystemUriResponse">
    <wsdl:part name="parameters" element="tds:GetSystemUriResponse"/>
</wsdl:message>
<wsdl:message name="StartFirmwareUpgradeRequest">
    <wsdl:part name="parameters" element="tds:StartFirmwareUpgrade"/>
</wsdl:message>
<wsdl:message name="StartFirmwareUpgradeResponse">
    <wsdl:part name="parameters" element="tds:StartFirmwareUpgradeResponse"/>

```

```
</wsdl:message>
<wsdl:message name="StartSystemRestoreRequest">
  <wsdl:part name="parameters" element="tds:StartSystemRestore"/>
</wsdl:message>
<wsdl:message name="StartSystemRestoreResponse">
  <wsdl:part name="parameters" element="tds:StartSystemRestoreResponse"/>
</wsdl:message>
<wsdl:portType name="Device">
  <wsdl:operation name="GetDeviceInformation">
    <wsdl:input message="tds:GetDeviceInformationRequest"/>
    <wsdl:output message="tds:GetDeviceInformationResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SetSystemDateAndTime">
    <wsdl:input message="tds:SetSystemDateAndTimeRequest"/>
    <wsdl:output message="tds:SetSystemDateAndTimeResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetSystemDateAndTime">
    <wsdl:input message="tds:GetSystemDateAndTimeRequest"/>
    <wsdl:output message="tds:GetSystemDateAndTimeResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SetSystemFactoryDefault">
    <wsdl:input message="tds:SetSystemFactoryDefaultRequest"/>
    <wsdl:output message="tds:SetSystemFactoryDefaultResponse"/>
  </wsdl:operation>
  <wsdl:operation name="UpgradeSystemFirmware">
    <wsdl:input message="tds:UpgradeSystemFirmwareRequest"/>
    <wsdl:output message="tds:UpgradeSystemFirmwareResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SystemReboot">
    <wsdl:input message="tds:SystemRebootRequest"/>
    <wsdl:output message="tds:SystemRebootResponse"/>
  </wsdl:operation>
  <wsdl:operation name="RestoreSystem">
    <wsdl:input message="tds:RestoreSystemRequest"/>
    <wsdl:output message="tds:RestoreSystemResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetSystemBackup">
    <wsdl:input message="tds:GetSystemBackupRequest"/>
    <wsdl:output message="tds:GetSystemBackupResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetSystemLog">
    <wsdl:input message="tds:GetSystemLogRequest"/>
    <wsdl:output message="tds:GetSystemLogResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetSystemSupportInformation">
    <wsdl:input message="tds:GetSystemSupportInformationRequest"/>
    <wsdl:output message="tds:GetSystemSupportInformationResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetScopes">
    <wsdl:input message="tds:GetScopesRequest"/>
    <wsdl:output message="tds:GetScopesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SetScopes">
    <wsdl:input message="tds:SetScopesRequest"/>
    <wsdl:output message="tds:SetScopesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="AddScopes">
    <wsdl:input message="tds:AddScopesRequest"/>
    <wsdl:output message="tds:AddScopesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="RemoveScopes">
    <wsdl:input message="tds:RemoveScopesRequest"/>
  </wsdl:operation>
</wsdl:portType>
</wsdl:binding>
</wsdl:service>
</wsdl:definitions>
```

```

    <wsdl:output message="tds:RemoveScopesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDiscoveryMode">
    <wsdl:input message="tds:GetDiscoveryModeRequest"/>
    <wsdl:output message="tds:GetDiscoveryModeResponse"/>
</wsdl:operation>
<wsdl:operation name="SetDiscoveryMode">
    <wsdl:input message="tds:SetDiscoveryModeRequest"/>
    <wsdl:output message="tds:SetDiscoveryModeResponse"/>
</wsdl:operation>
<wsdl:operation name="GetRemoteDiscoveryMode">
    <wsdl:input message="tds:GetRemoteDiscoveryModeRequest"/>
    <wsdl:output message="tds:GetRemoteDiscoveryModeResponse"/>
</wsdl:operation>
<wsdl:operation name="SetRemoteDiscoveryMode">
    <wsdl:input message="tds:SetRemoteDiscoveryModeRequest"/>
    <wsdl:output message="tds:SetRemoteDiscoveryModeResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDPAddresses">
    <wsdl:input message="tds:GetDPAddressesRequest"/>
    <wsdl:output message="tds:GetDPAddressesResponse"/>
</wsdl:operation>
<wsdl:operation name="SetDPAddresses">
    <wsdl:input message="tds:SetDPAddressesRequest"/>
    <wsdl:output message="tds:SetDPAddressesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetEndpointReference">
    <wsdl:input message="tds:GetEndpointReferenceRequest"/>
    <wsdl:output message="tds:GetEndpointReferenceResponse"/>
</wsdl:operation>
<wsdl:operation name="GetRemoteUser">
    <wsdl:input message="tds:GetRemoteUserRequest"/>
    <wsdl:output message="tds:GetRemoteUserResponse"/>
</wsdl:operation>
<wsdl:operation name="SetRemoteUser">
    <wsdl:input message="tds:SetRemoteUserRequest"/>
    <wsdl:output message="tds:SetRemoteUserResponse"/>
</wsdl:operation>
<wsdl:operation name="GetUsers">
    <wsdl:input message="tds:GetUsersRequest"/>
    <wsdl:output message="tds:GetUsersResponse"/>
</wsdl:operation>
<wsdl:operation name="CreateUsers">
    <wsdl:input message="tds:CreateUsersRequest"/>
    <wsdl:output message="tds:CreateUsersResponse"/>
</wsdl:operation>
<wsdl:operation name="DeleteUsers">
    <wsdl:input message="tds>DeleteUsersRequest"/>
    <wsdl:output message="tds>DeleteUsersResponse"/>
</wsdl:operation>
<wsdl:operation name="SetUser">
    <wsdl:input message="tds:SetUserRequest"/>
    <wsdl:output message="tds:SetUserResponse"/>
</wsdl:operation>
<wsdl:operation name="GetWsdUrl">
    <wsdl:input message="tds:GetWsdUrlRequest"/>
    <wsdl:output message="tds:GetWsdUrlResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCapabilities">
    <wsdl:input message="tds:GetCapabilitiesRequest"/>
    <wsdl:output message="tds:GetCapabilitiesResponse"/>
</wsdl:operation>

```

```
<wsdl:operation name="GetHostname">
  <wsdl:input message="tds:GetHostnameRequest"/>
  <wsdl:output message="tds:GetHostnameResponse"/>
</wsdl:operation>
<wsdl:operation name="SetHostname">
  <wsdl:input message="tds:SetHostnameRequest"/>
  <wsdl:output message="tds:SetHostnameResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDNS">
  <wsdl:input message="tds:GetDNSRequest"/>
  <wsdl:output message="tds:GetDNSResponse"/>
</wsdl:operation>
<wsdl:operation name="SetDNS">
  <wsdl:input message="tds:SetDNSRequest"/>
  <wsdl:output message="tds:SetDNSResponse"/>
</wsdl:operation>
<wsdl:operation name="GetNTP">
  <wsdl:input message="tds:GetNTPRequest"/>
  <wsdl:output message="tds:GetNTPResponse"/>
</wsdl:operation>
<wsdl:operation name="SetNTP">
  <wsdl:input message="tds:SetNTPRequest"/>
  <wsdl:output message="tds:SetNTPResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDynamicDNS">
  <wsdl:input message="tds:GetDynamicDNSRequest"/>
  <wsdl:output message="tds:GetDynamicDNSResponse"/>
</wsdl:operation>
<wsdl:operation name="SetDynamicDNS">
  <wsdl:input message="tds:SetDynamicDNSRequest"/>
  <wsdl:output message="tds:SetDynamicDNSResponse"/>
</wsdl:operation>
<wsdl:operation name="GetNetworkInterfaces">
  <wsdl:input message="tds:GetNetworkInterfacesRequest"/>
  <wsdl:output message="tds:GetNetworkInterfacesResponse"/>
</wsdl:operation>
<wsdl:operation name="SetNetworkInterfaces">
  <wsdl:input message="tds:SetNetworkInterfacesRequest"/>
  <wsdl:output message="tds:SetNetworkInterfacesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetNetworkProtocols">
  <wsdl:input message="tds:GetNetworkProtocolsRequest"/>
  <wsdl:output message="tds:GetNetworkProtocolsResponse"/>
</wsdl:operation>
<wsdl:operation name="SetNetworkProtocols">
  <wsdl:input message="tds:SetNetworkProtocolsRequest"/>
  <wsdl:output message="tds:SetNetworkProtocolsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetNetworkDefaultGateway">
  <wsdl:input message="tds:GetNetworkDefaultGatewayRequest"/>
  <wsdl:output message="tds:GetNetworkDefaultGatewayResponse"/>
</wsdl:operation>
<wsdl:operation name="SetNetworkDefaultGateway">
  <wsdl:input message="tds:SetNetworkDefaultGatewayRequest"/>
  <wsdl:output message="tds:SetNetworkDefaultGatewayResponse"/>
</wsdl:operation>
<wsdl:operation name="GetZeroConfiguration">
  <wsdl:input message="tds:GetZeroConfigurationRequest"/>
  <wsdl:output message="tds:GetZeroConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetZeroConfiguration">
  <wsdl:input message="tds:SetZeroConfigurationRequest"/>
```



```

    <wsdl:output message="tds:SetZeroConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetIPAddressFilter">
    <wsdl:input message="tds:GetIPAddressFilterRequest"/>
    <wsdl:output message="tds:GetIPAddressFilterResponse"/>
</wsdl:operation>
<wsdl:operation name="SetIPAddressFilter">
    <wsdl:input message="tds:SetIPAddressFilterRequest"/>
    <wsdl:output message="tds:SetIPAddressFilterResponse"/>
</wsdl:operation>
<wsdl:operation name="AddIPAddressFilter">
    <wsdl:input message="tds:AddIPAddressFilterRequest"/>
    <wsdl:output message="tds:AddIPAddressFilterResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveIPAddressFilter">
    <wsdl:input message="tds:RemoveIPAddressFilterRequest"/>
    <wsdl:output message="tds:RemoveIPAddressFilterResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAccessPolicy">
    <wsdl:input message="tds:GetAccessPolicyRequest"/>
    <wsdl:output message="tds:GetAccessPolicyResponse"/>
</wsdl:operation>
<wsdl:operation name="SetAccessPolicy">
    <wsdl:input message="tds:SetAccessPolicyRequest"/>
    <wsdl:output message="tds:SetAccessPolicyResponse"/>
</wsdl:operation>
<wsdl:operation name="CreateCertificate">
    <wsdl:input message="tds:CreateCertificateRequest"/>
    <wsdl:output message="tds:CreateCertificateResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCertificates">
    <wsdl:input message="tds:GetCertificatesRequest"/>
    <wsdl:output message="tds:GetCertificatesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCertificatesStatus">
    <wsdl:input message="tds:GetCertificatesStatusRequest"/>
    <wsdl:output message="tds:GetCertificatesStatusResponse"/>
</wsdl:operation>
<wsdl:operation name="SetCertificatesStatus">
    <wsdl:input message="tds:SetCertificatesStatusRequest"/>
    <wsdl:output message="tds:SetCertificatesStatusResponse"/>
</wsdl:operation>
<wsdl:operation name="DeleteCertificates">
    <wsdl:input message="tds>DeleteCertificatesRequest"/>
    <wsdl:output message="tds>DeleteCertificatesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetPkcs10Request">
    <wsdl:input message="tds:GetPkcs10RequestRequest"/>
    <wsdl:output message="tds:GetPkcs10RequestResponse"/>
</wsdl:operation>
<wsdl:operation name="LoadCertificates">
    <wsdl:input message="tds:LoadCertificatesRequest"/>
    <wsdl:output message="tds:LoadCertificatesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetClientCertificateMode">
    <wsdl:input message="tds:GetClientCertificateModeRequest"/>
    <wsdl:output message="tds:GetClientCertificateModeResponse"/>
</wsdl:operation>
<wsdl:operation name="SetClientCertificateMode">
    <wsdl:input message="tds:SetClientCertificateModeRequest"/>
    <wsdl:output message="tds:SetClientCertificateModeResponse"/>
</wsdl:operation>

```

```
<wsdl:operation name="GetRelayOutputs">
  <wsdl:input message="tds:GetRelayOutputsRequest"/>
  <wsdl:output message="tds:GetRelayOutputsResponse"/>
</wsdl:operation>
<wsdl:operation name="SetRelayOutputSettings">
  <wsdl:input message="tds:SetRelayOutputSettingsRequest"/>
  <wsdl:output message="tds:SetRelayOutputSettingsResponse"/>
</wsdl:operation>
<wsdl:operation name="SetRelayOutputState">
  <wsdl:input message="tds:SetRelayOutputStateRequest"/>
  <wsdl:output message="tds:SetRelayOutputStateResponse"/>
</wsdl:operation>
<wsdl:operation name="SendAuxiliaryCommand">
  <wsdl:input message="tds:SendAuxiliaryCommandRequest"/>
  <wsdl:output message="tds:SendAuxiliaryCommandResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCACertificates">
  <wsdl:input message="tds:GetCACertificatesRequest"/>
  <wsdl:output message="tds:GetCACertificatesResponse"/>
</wsdl:operation>
<wsdl:operation name="LoadCertificateWithPrivateKey">
  <wsdl:input message="tds:LoadCertificateWithPrivateKeyRequest"/>
  <wsdl:output message="tds:LoadCertificateWithPrivateKeyResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCertificateInformation">
  <wsdl:input message="tds:GetCertificateInformationRequest"/>
  <wsdl:output message="tds:GetCertificateInformationResponse"/>
</wsdl:operation>
<wsdl:operation name="LoadCACertificates">
  <wsdl:input message="tds:LoadCACertificatesRequest"/>
  <wsdl:output message="tds:LoadCACertificatesResponse"/>
</wsdl:operation>
<wsdl:operation name="CreateDot1XConfiguration">
  <wsdl:input message="tds:CreateDot1XConfigurationRequest"/>
  <wsdl:output message="tds:CreateDot1XConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetDot1XConfiguration">
  <wsdl:input message="tds:SetDot1XConfigurationRequest"/>
  <wsdl:output message="tds:SetDot1XConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDot1XConfiguration">
  <wsdl:input message="tds:GetDot1XConfigurationRequest"/>
  <wsdl:output message="tds:GetDot1XConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDot1XConfigurations">
  <wsdl:input message="tds:GetDot1XConfigurationsRequest"/>
  <wsdl:output message="tds:GetDot1XConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="DeleteDot1XConfiguration">
  <wsdl:input message="tds>DeleteDot1XConfigurationRequest"/>
  <wsdl:output message="tds>DeleteDot1XConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDot11Capabilities">
  <wsdl:input message="tds:GetDot11CapabilitiesRequest"/>
  <wsdl:output message="tds:GetDot11CapabilitiesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetDot11Status">
  <wsdl:input message="tds:GetDot11StatusRequest"/>
  <wsdl:output message="tds:GetDot11StatusResponse"/>
</wsdl:operation>
<wsdl:operation name="ScanAvailableDot11Networks">
  <wsdl:input message="tds:ScanAvailableDot11NetworksRequest"/>
```

```

        <wsdl:output message="tds:ScanAvailableDot11NetworksResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetSystemUris">
        <wsdl:input message="tds:GetSystemUrisRequest"/>
        <wsdl:output message="tds:GetSystemUrisResponse"/>
    </wsdl:operation>
    <wsdl:operation name="StartFirmwareUpgrade">
        <wsdl:input message="tds:StartFirmwareUpgradeRequest"/>
        <wsdl:output message="tds:StartFirmwareUpgradeResponse"/>
    </wsdl:operation>
    <wsdl:operation name="StartSystemRestore">
        <wsdl:input message="tds:StartSystemRestoreRequest"/>
        <wsdl:output message="tds:StartSystemRestoreResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="DeviceBinding" type="tds:Device">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="GetDeviceInformation">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetDeviceInformation"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetSystemDateAndTime">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetSystemDateAndTime"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetSystemDateAndTime">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetSystemDateAndTime"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetSystemFactoryDefault">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetSystemFactoryDefault"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="UpgradeSystemFirmware">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/UpgradeSystemFirmware"/>
        <wsdl:input>
            <soap:body use="literal"/>

```

```

        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SystemReboot">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SystemReboot"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RestoreSystem">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/RestoreSystem"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetSystemBackup">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetSystemBackup"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetSystemLog">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetSystemLog"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetSystemSupportInformation">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetSystemSupportInformation"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetScopes">
        <soap:operation soapAction="http://www.onvif.org/ver10/device/wsdl/GetScopes"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetScopes">
        <soap:operation soapAction="http://www.onvif.org/ver10/device/wsdl/SetScopes"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="AddScopes">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/AddScopes"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RemoveScopes">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/RemoveScopes"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetDiscoveryMode">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetDiscoveryMode"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetDiscoveryMode">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetDiscoveryMode"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetRemoteDiscoveryMode">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetRemoteDiscoveryMode"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetRemoteDiscoveryMode">

```



```

        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/GetRemoteDiscoveryMode"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetDPAddresses">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/GetDPAddresses"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetEndpointReference">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/GetEndpointReference"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetRemoteUser">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/GetRemoteUser"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetRemoteUser">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/SetRemoteUser"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetUsers">
        <soap:operation soapAction="http://www.onvif.org/ver10/device/wsd/GetUsers"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateUsers">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/CreateUsers"/>
        <wsdl:input>

```

```

        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="DeleteUsers">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/DeleteUsers"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetUser">
    <soap:operation soapAction="http://www.onvif.org/ver10/device/wsdl/SetUser"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetWsdlUrl">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetWsdlUrl"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetCapabilities">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetCapabilities"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetDPAddresses">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetDPAddresses"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetHostname">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetHostname"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>

```

```

        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetHostname">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetHostname"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetDNS">
    <soap:operation soapAction="http://www.onvif.org/ver10/device/wsdl/GetDNS"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetDNS">
    <soap:operation soapAction="http://www.onvif.org/ver10/device/wsdl/SetDNS"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetNTP">
    <soap:operation soapAction="http://www.onvif.org/ver10/device/wsdl/GetNTP"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetNTP">
    <soap:operation soapAction="http://www.onvif.org/ver10/device/wsdl/SetNTP"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetDynamicDNS">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetDynamicDNS"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetDynamicDNS">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetDynamicDNS"/>

```



```

        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetNetworkInterfaces">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetNetworkInterfaces"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetNetworkInterfaces">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetNetworkInterfaces"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetNetworkProtocols">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetNetworkProtocols"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetNetworkProtocols">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetNetworkProtocols"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetNetworkDefaultGateway">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetNetworkDefaultGateway"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetNetworkDefaultGateway">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetNetworkDefaultGateway"/>
        <wsdl:input>
            <soap:body use="literal"/>

```

```

        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetZeroConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetZeroConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetZeroConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetZeroConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetIPAddressFilter">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetIPAddressFilter"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetIPAddressFilter">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetIPAddressFilter"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="AddIPAddressFilter">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/AddIPAddressFilter"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RemoveIPAddressFilter">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/RemoveIPAddressFilter"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>

```

```

        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAccessPolicy">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetAccessPolicy"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="SetAccessPolicy">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetAccessPolicy"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="CreateCertificate">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/CreateCertificate"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="GetCertificates">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetCertificates"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="GetCertificatesStatus">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetCertificatesStatus"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="SetCertificatesStatus">
    <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetCertificatesStatus"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```
</wsdl:operation>
<wsdl:operation name="DeleteCertificates">
  <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/DeleteCertificates"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetPkcs10Request">
  <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetPkcs10Request"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="LoadCertificates">
  <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/LoadCertificates"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetClientCertificateMode">
  <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetClientCertificateMode"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetClientCertificateMode">
  <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/SetClientCertificateMode"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetRelayOutputs">
  <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetRelayOutputs"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetRelayOutputSettings">
```

```

        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/SetRelayOutputSettings"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetRelayOutputState">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/SetRelayOutputState"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SendAuxiliaryCommand">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/SendAuxiliaryCommand"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCACertificates">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/GetCACertificates"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="LoadCertificateWithPrivateKey">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/LoadCertificateWithPrivateKey"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCertificateInformation">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/GetCertificateInformation"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="LoadCACertificates">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsd/LoadCACertificates"/>

```

```

    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="CreateDot1XConfiguration">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/device/wsdl/CreateDot1XConfiguration"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SetDot1XConfiguration">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/device/wsdl/SetDot1XConfiguration"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetDot1XConfiguration">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/device/wsdl/GetDot1XConfiguration"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetDot1XConfigurations">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/device/wsdl/GetDot1XConfigurations"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="DeleteDot1XConfiguration">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/device/wsdl/DeleteDot1XConfiguration"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetDot11Capabilities">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/device/wsdl/GetDot11Capabilities"/>
    <wsdl:input>
      <soap:body use="literal"/>

```



```

        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetDot11Status">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetDot11Status"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="ScanAvailableDot11Networks">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/ScanAvailableDot11Networks"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetSystemUri">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/GetSystemUri"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="StartFirmwareUpgrade">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/StartFirmwareUpgrade"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="StartSystemRestore">
        <soap:operation
soapAction="http://www.onvif.org/ver10/device/wsdl/StartSystemRestore"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding><!--=====--><!--
=====-->
</wsdl:definitions>

```

C.5 Display service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:tns="http://www.onvif.org/ver10/display/wsdl"
  targetNamespace="http://www.onvif.org/ver10/display/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tns="http://www.onvif.org/ver10/schema"
      targetNamespace="http://www.onvif.org/ver10/display/wsdl" elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
        schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!--
=====
      <xs:element name="GetLayout">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="VideoOutput" type="tns:ReferenceToken"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====
      <xs:element name="GetLayoutResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Layout" type="tns:Layout"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====
      <xs:element name="SetLayout">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="VideoOutput" type="tns:ReferenceToken"/>
            <xs:element name="Layout" type="tns:Layout"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====
      <xs:element name="SetLayoutResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====
      <xs:element name="GetDisplayOptions">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="VideoOutput" type="tns:ReferenceToken"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====
      <xs:element name="GetDisplayOptionsResponse">
        <xs:complexType>

```



```

<xs:sequence>
  <xs:element name="LayoutOptions" type="tt:LayoutOptions" minOccurs="0"/>
  <xs:element name="CodingCapabilities" type="tt:CodingCapabilities"/>
  <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetPaneConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoOutput" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetPaneConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PaneConfiguration" type="tt:PaneConfiguration"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetPaneConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoOutput" type="tt:ReferenceToken"/>
      <xs:element name="Pane" type="tt:ReferenceToken"/>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetPaneConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PaneConfiguration" type="tt:PaneConfiguration"/>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetPaneConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VideoOutput" type="tt:ReferenceToken"/>
      <xs:element name="PaneConfiguration" type="tt:PaneConfiguration"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetPaneConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetPaneConfiguration">

```

```

        <xs:complexType>
          <xs:sequence>
            <xs:element name="VideoOutput" type="tt:ReferenceToken"/>
            <xs:element name="PaneConfiguration" type="tt:PaneConfiguration"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!------->
      <xs:element name="SetPaneConfigurationResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!------->
      <xs:element name="CreatePaneConfiguration">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="VideoOutput" type="tt:ReferenceToken"/>
            <xs:element name="PaneConfiguration" type="tt:PaneConfiguration"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!------->
      <xs:element name="CreatePaneConfigurationResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!------->
      <xs:element name="DeletePaneConfiguration">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="VideoOutput" type="tt:ReferenceToken"/>
            <xs:element name="PaneToken" type="tt:ReferenceToken"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!------->
      <xs:element name="DeletePaneConfigurationResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  </wsdl:types>
  <wsdl:message name="GetLayoutRequest">
    <wsdl:part name="parameters" element="tls:GetLayout"/>
  </wsdl:message>
  <wsdl:message name="GetLayoutResponse">
    <wsdl:part name="parameters" element="tls:GetLayoutResponse"/>
  </wsdl:message>
  <wsdl:message name="SetLayoutRequest">

```

```

    <wsdl:part name="parameters" element="tls:SetLayout"/>
  </wsdl:message>
  <wsdl:message name="SetLayoutResponse">
    <wsdl:part name="parameters" element="tls:SetLayoutResponse"/>
  </wsdl:message>
  <wsdl:message name="GetDisplayOptionsRequest">
    <wsdl:part name="parameters" element="tls:GetDisplayOptions"/>
  </wsdl:message>
  <wsdl:message name="GetDisplayOptionsResponse">
    <wsdl:part name="parameters" element="tls:GetDisplayOptionsResponse"/>
  </wsdl:message>
  <wsdl:message name="GetPaneConfigurationsRequest">
    <wsdl:part name="parameters" element="tls:GetPaneConfigurations"/>
  </wsdl:message>
  <wsdl:message name="GetPaneConfigurationsResponse">
    <wsdl:part name="parameters" element="tls:GetPaneConfigurationsResponse"/>
  </wsdl:message>
  <wsdl:message name="GetPaneConfigurationRequest">
    <wsdl:part name="parameters" element="tls:GetPaneConfiguration"/>
  </wsdl:message>
  <wsdl:message name="GetPaneConfigurationResponse">
    <wsdl:part name="parameters" element="tls:GetPaneConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="SetPaneConfigurationsRequest">
    <wsdl:part name="parameters" element="tls:SetPaneConfigurations"/>
  </wsdl:message>
  <wsdl:message name="SetPaneConfigurationsResponse">
    <wsdl:part name="parameters" element="tls:SetPaneConfigurationsResponse"/>
  </wsdl:message>
  <wsdl:message name="SetPaneConfigurationRequest">
    <wsdl:part name="parameters" element="tls:SetPaneConfiguration"/>
  </wsdl:message>
  <wsdl:message name="SetPaneConfigurationResponse">
    <wsdl:part name="parameters" element="tls:SetPaneConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="CreatePaneConfigurationRequest">
    <wsdl:part name="parameters" element="tls>CreatePaneConfiguration"/>
  </wsdl:message>
  <wsdl:message name="CreatePaneConfigurationResponse">
    <wsdl:part name="parameters" element="tls>CreatePaneConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="DeletePaneConfigurationRequest">
    <wsdl:part name="parameters" element="tls>DeletePaneConfiguration"/>
  </wsdl:message>
  <wsdl:message name="DeletePaneConfigurationResponse">
    <wsdl:part name="parameters" element="tls>DeletePaneConfigurationResponse"/>
  </wsdl:message>
  <wsdl:portType name="DisplayPort">
    <wsdl:operation name="GetLayout">
      <wsdl:input message="tls:GetLayoutRequest"/>
      <wsdl:output message="tls:GetLayoutResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetLayout">
      <wsdl:input message="tls:SetLayoutRequest"/>
      <wsdl:output message="tls:SetLayoutResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetDisplayOptions">
      <wsdl:input message="tls:GetDisplayOptionsRequest"/>
      <wsdl:output message="tls:GetDisplayOptionsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetPaneConfigurations">
      <wsdl:input message="tls:GetPaneConfigurationsRequest"/>
    </wsdl:operation>
  </wsdl:portType>

```

```

        <wsdl:output message="tls:GetPaneConfigurationsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetPaneConfiguration">
        <wsdl:input message="tls:GetPaneConfigurationRequest"/>
        <wsdl:output message="tls:GetPaneConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetPaneConfigurations">
        <wsdl:input message="tls:SetPaneConfigurationsRequest"/>
        <wsdl:output message="tls:SetPaneConfigurationsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetPaneConfiguration">
        <wsdl:input message="tls:SetPaneConfigurationRequest"/>
        <wsdl:output message="tls:SetPaneConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreatePaneConfiguration">
        <wsdl:input message="tls>CreatePaneConfigurationRequest"/>
        <wsdl:output message="tls>CreatePaneConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="DeletePaneConfiguration">
        <wsdl:input message="tls>DeletePaneConfigurationRequest"/>
        <wsdl:output message="tls>DeletePaneConfigurationResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="DisplayBinding" type="tls:DisplayPort">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="GetLayout">
        <soap:operation soapAction="http://www.onvif.org/ver10/display/wsdl/GetLayout"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetLayout">
        <soap:operation soapAction="http://www.onvif.org/ver10/display/wsdl/SetLayout"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetDisplayOptions">
        <soap:operation
soapAction="http://www.onvif.org/ver10/display/wsdl/GetDisplayOptions"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetPaneConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/display/wsdl/GetPaneConfigurations"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

</wsdl:operation>
<wsdl:operation name="GetPaneConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/display/wsdl/GetPaneConfiguration"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetPaneConfigurations">
  <soap:operation
soapAction="http://www.onvif.org/ver10/display/wsdl/SetPaneConfigurations"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetPaneConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/display/wsdl/SetPaneConfiguration"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="CreatePaneConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/display/wsdl/CreatePaneConfiguration"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="DeletePaneConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/display/wsdl/DeletePaneConfiguration"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

C.6 Event service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:wsa="http://www.w3.org/2005/08/addressing"

```



```

xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:wsnt="http://docs.oasis-
open.org/wsn/b-2" xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:wsntw="http://docs.oasis-open.org/wsn/bw-2"
xmlns:tev="http://www.onvif.org/ver10/events/wsd1" xmlns:wsrf-rw="http://docs.oasis-
open.org/wsrf/rw-2" xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsd1"
targetNamespace="http://www.onvif.org/ver10/events/wsd1">
  <wsdl:import namespace="http://docs.oasis-open.org/wsn/bw-2" location="http://docs.oasis-
open.org/wsn/bw-2.wsd1"/>
  <wsdl:types>
    <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
targetNamespace="http://www.onvif.org/ver10/events/wsd1" elementFormDefault="qualified">
      <xs:import namespace="http://www.w3.org/2005/08/addressing"
schemaLocation="http://www.w3.org/2005/08/addressing/ws-addr.xsd"/>
      <xs:import namespace="http://docs.oasis-open.org/wsn/t-1"
schemaLocation="http://docs.oasis-open.org/wsn/t-1.xsd"/>
      <xs:import namespace="http://docs.oasis-open.org/wsn/b-2"
schemaLocation="http://docs.oasis-open.org/wsn/b-2.xsd"/>
      <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/ver10/schema/onvif.xsd"/>
      <xs:element name="GetServiceCapabilities">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetServiceCapabilitiesResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Capabilities" type="tev:Capabilities"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:complexType name="Capabilities">
        <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="WSSubscriptionPolicySupport" type="xs:boolean"/>
        <xs:attribute name="WSPullPointSupport" type="xs:boolean"/>
        <xs:attribute name="WSPausableSubscriptionManagerInterfaceSupport"
type="xs:boolean"/>
      </xs:complexType>
      <xs:element name="Capabilities" type="tev:Capabilities"/>
      <xs:element name="CreatePullPointSubscription">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Filter" type="wsnt:FilterType" minOccurs="0"/>
            <xs:element name="InitialTerminationTime"
type="wsnt:AbsoluteOrRelativeTimeType" nillable="true" minOccurs="0"/>
            <xs:element name="SubscriptionPolicy" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

```

```

<xs:element name="CreatePullPointSubscriptionResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SubscriptionReference"
type="wsa:EndpointReferenceType"/>
      <xs:element ref="wsnt:CurrentTime"/>
      <xs:element ref="wsnt:TerminationTime"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PullMessages">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Timeout" type="xs:duration"/>
      <xs:element name="MessageLimit" type="xs:int"/>
      <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PullMessagesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CurrentTime" type="xs:dateTime"/>
      <xs:element name="TerminationTime" type="xs:dateTime"/>
      <xs:element ref="wsnt:NotificationMessage" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PullMessagesFaultResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="MaxTimeout" type="xs:duration"/>
      <xs:element name="MaxMessageLimit" type="xs:int"/>
      <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!--=====>
<xs:element name="SetSynchronizationPoint">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="SetSynchronizationPointResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<!--=====>
<xs:element name="GetEventProperties">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetEventPropertiesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="TopicNamespaceLocation" type="xs:anyURI" minOccurs="1"
maxOccurs="unbounded"/>
      <xs:element ref="wsnt:FixedTopicSet"/>
      <xs:element ref="wstop:TopicSet"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

        <xs:element ref="wsnt:TopicExpressionDialect" minOccurs="1"
maxOccurs="unbounded"/>
        <xs:element name="MessageContentFilterDialect" type="xs:anyURI"
minOccurs="1" maxOccurs="unbounded"/>
        <xs:element name="ProducerPropertiesFilterDialect" type="xs:anyURI"
minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="MessageContentSchemaLocation" type="xs:anyURI"
minOccurs="1" maxOccurs="unbounded"/>
        <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
</xs:element><!--=====-->
</xs:schema>
</wsdl:types>
<wsdl:message name="GetServiceCapabilitiesRequest">
    <wsdl:part name="parameters" element="tev:GetServiceCapabilities"/>
</wsdl:message>
<wsdl:message name="GetServiceCapabilitiesResponse">
    <wsdl:part name="parameters" element="tev:GetServiceCapabilitiesResponse"/>
</wsdl:message>
<wsdl:message name="CreatePullPointSubscriptionRequest">
    <wsdl:part name="parameters" element="tev:CreatePullPointSubscription"/>
</wsdl:message>
<wsdl:message name="CreatePullPointSubscriptionResponse">
    <wsdl:part name="parameters" element="tev:CreatePullPointSubscriptionResponse"/>
</wsdl:message>
<wsdl:message name="PullMessagesRequest">
    <wsdl:part name="parameters" element="tev:PullMessages"/>
</wsdl:message>
<wsdl:message name="PullMessagesResponse">
    <wsdl:part name="parameters" element="tev:PullMessagesResponse"/>
</wsdl:message>
<wsdl:message name="PullMessagesFaultResponse">
    <wsdl:part name="parameters" element="tev:PullMessagesFaultResponse"/>
</wsdl:message>
<wsdl:message name="SetSynchronizationPointRequest">
    <wsdl:part name="parameters" element="tev:SetSynchronizationPoint"/>
</wsdl:message>
<wsdl:message name="SetSynchronizationPointResponse">
    <wsdl:part name="parameters" element="tev:SetSynchronizationPointResponse"/>
</wsdl:message>
<wsdl:message name="GetEventPropertiesRequest">
    <wsdl:part name="parameters" element="tev:GetEventProperties"/>
</wsdl:message>
<wsdl:message name="GetEventPropertiesResponse">
    <wsdl:part name="parameters" element="tev:GetEventPropertiesResponse"/>
</wsdl:message>
<wsdl:portType name="EventPortType">
    <wsdl:operation name="GetServiceCapabilities">
        <wsdl:input message="tev:GetServiceCapabilitiesRequest"/>
        <wsdl:output message="tev:GetServiceCapabilitiesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreatePullPointSubscription">
        <wsdl:input message="tev:CreatePullPointSubscriptionRequest"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/EventPortType/CreatePullPointSubscrip
tionRequest"/>
        <wsdl:output message="tev:CreatePullPointSubscriptionResponse"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/EventPortType/CreatePullPointSubscrip
tionResponse"/>
        <wsdl:fault name="ResourceUnknownFault" message="wsrf-
rw:ResourceUnknownFault"/>
        <wsdl:fault name="InvalidFilterFault" message="wsntw:InvalidFilterFault"/>
    </wsdl:operation>
</wsdl:portType>

```



```

        <wsdl:fault name="TopicExpressionDialectUnknownFault"
message="wsntw:TopicExpressionDialectUnknownFault"/>
        <wsdl:fault name="InvalidTopicExpressionFault"
message="wsntw:InvalidTopicExpressionFault"/>
        <wsdl:fault name="TopicNotSupportedFault"
message="wsntw:TopicNotSupportedFault"/>
        <wsdl:fault name="InvalidProducerPropertiesExpressionFault"
message="wsntw:InvalidProducerPropertiesExpressionFault"/>
        <wsdl:fault name="InvalidMessageContentExpressionFault"
message="wsntw:InvalidMessageContentExpressionFault"/>
        <wsdl:fault name="UnacceptableInitialTerminationTimeFault"
message="wsntw:UnacceptableInitialTerminationTimeFault"/>
        <wsdl:fault name="UnrecognizedPolicyRequestFault"
message="wsntw:UnrecognizedPolicyRequestFault"/>
        <wsdl:fault name="UnsupportedPolicyRequestFault"
message="wsntw:UnsupportedPolicyRequestFault"/>
        <wsdl:fault name="NotifyMessageNotSupportedFault"
message="wsntw:NotifyMessageNotSupportedFault"/>
        <wsdl:fault name="SubscribeCreationFailedFault"
message="wsntw:SubscribeCreationFailedFault"/>
    </wsdl:operation>
    <wsdl:operation name="GetEventProperties">
        <wsdl:input message="tev:GetEventPropertiesRequest"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/EventPortType/GetEventPropertiesReq
uest"/>
        <wsdl:output message="tev:GetEventPropertiesResponse"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/EventPortType/GetEventPropertiesResp
onse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:portType name="PullPointSubscription">
    <wsdl:operation name="PullMessages">
        <wsdl:input message="tev:PullMessagesRequest"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/PullMessagesReq
uest"/>
        <wsdl:output message="tev:PullMessagesResponse"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/PullMessagesRes
ponse"/>
        <wsdl:fault name="PullMessagesFaultResponse"
message="tev:PullMessagesFaultResponse"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/PullMessages/Fa
ult/PullMessagesFaultResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetSynchronizationPoint">
        <wsdl:input message="tev:SetSynchronizationPointRequest"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/SetSynchronizatio
nPointRequest"/>
        <wsdl:output message="tev:SetSynchronizationPointResponse"
wsaw:Action="http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/SetSynchronizatio
nPointResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="PullPointSubscriptionBinding" type="tev:PullPointSubscription">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="PullMessages">
        <soap:operation
soapAction="http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/PullMessagesReq
uest"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>

```

```

        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="PullMessagesFaultResponse">
        <soap:fault name="PullMessagesFaultResponse" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="SetSynchronizationPoint">
    <soap:operation
soapAction="http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/SetSynchronization
PointRequest"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="EventBinding" type="tev:EventPortType">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="GetServiceCapabilities">
        <soap:operation
soapAction="http://www.onvif.org/ver10/events/wsdl/EventPortType/GetServiceCapabilities"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreatePullPointSubscription">
        <soap:operation
soapAction="http://www.onvif.org/ver10/events/wsdl/EventPortType/CreatePullPointSubscripti
onRequest"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="ResourceUnknownFault">
            <soap:fault name="ResourceUnknownFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidFilterFault">
            <soap:fault name="InvalidFilterFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="TopicExpressionDialectUnknownFault">
            <soap:fault name="TopicExpressionDialectUnknownFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidTopicExpressionFault">
            <soap:fault name="InvalidTopicExpressionFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="TopicNotSupportedFault">
            <soap:fault name="TopicNotSupportedFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidProducerPropertiesExpressionFault">
            <soap:fault name="InvalidProducerPropertiesExpressionFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidMessageContentExpressionFault">
            <soap:fault name="InvalidMessageContentExpressionFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="UnacceptableInitialTerminationTimeFault">

```

```

        <soap:fault name="UnacceptableInitialTerminationTimeFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnrecognizedPolicyRequestFault">
        <soap:fault name="UnrecognizedPolicyRequestFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnsupportedPolicyRequestFault">
        <soap:fault name="UnsupportedPolicyRequestFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="NotifyMessageNotSupportedFault">
        <soap:fault name="NotifyMessageNotSupportedFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="SubscribeCreationFailedFault">
        <soap:fault name="SubscribeCreationFailedFault" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetEventProperties">
    <soap:operation
soapAction="http://www.onvif.org/ver10/events/wsdl/EventPortType/GetEventPropertiesReque
st"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="SubscriptionManagerBinding" type="wsntw:SubscriptionManager">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="Renew">
        <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/SubscriptionManager/RenewRequest"/>
        <wsdl:input name="RenewRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="RenewResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="ResourceUnknownFault">
            <soap:fault name="ResourceUnknownFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="UnacceptableTerminationTimeFault">
            <soap:fault name="UnacceptableTerminationTimeFault" use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="Unsubscribe">
        <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/SubscriptionManager/UnsubscribeRequest"/>
        <wsdl:input name="UnsubscribeRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="UnsubscribeResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="ResourceUnknownFault">
            <soap:fault name="ResourceUnknownFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="UnableToDestroySubscriptionFault">
            <soap:fault name="UnableToDestroySubscriptionFault" use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
</wsdl:binding>

```

```

<wsdl:binding name="NotificationProducerBinding" type="wsntw:NotificationProducer">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="Subscribe">
    <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/NotificationProducer/SubscribeRequest"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="ResourceUnknownFault">
      <soap:fault name="ResourceUnknownFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidFilterFault">
      <soap:fault name="InvalidFilterFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="TopicExpressionDialectUnknownFault">
      <soap:fault name="TopicExpressionDialectUnknownFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidTopicExpressionFault">
      <soap:fault name="InvalidTopicExpressionFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="TopicNotSupportedFault">
      <soap:fault name="TopicNotSupportedFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidProducerPropertiesExpressionFault">
      <soap:fault name="InvalidProducerPropertiesExpressionFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidMessageContentExpressionFault">
      <soap:fault name="InvalidMessageContentExpressionFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnacceptableInitialTerminationTimeFault">
      <soap:fault name="UnacceptableInitialTerminationTimeFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnrecognizedPolicyRequestFault">
      <soap:fault name="UnrecognizedPolicyRequestFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnsupportedPolicyRequestFault">
      <soap:fault name="UnsupportedPolicyRequestFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="NotifyMessageNotSupportedFault">
      <soap:fault name="NotifyMessageNotSupportedFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="SubscribeCreationFailedFault">
      <soap:fault name="SubscribeCreationFailedFault" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="GetCurrentMessage">
    <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/NotificationProducer/GetCurrentMessageRequest"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="ResourceUnknownFault">
      <soap:fault name="ResourceUnknownFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="TopicExpressionDialectUnknownFault">
      <soap:fault name="TopicExpressionDialectUnknownFault" use="literal"/>

```

```

</wsdl:fault>
<wsdl:fault name="InvalidTopicExpressionFault">
  <soap:fault name="InvalidTopicExpressionFault" use="literal"/>
</wsdl:fault>
<wsdl:fault name="TopicNotSupportedFault">
  <soap:fault name="TopicNotSupportedFault" use="literal"/>
</wsdl:fault>
<wsdl:fault name="NoCurrentMessageOnTopicFault">
  <soap:fault name="NoCurrentMessageOnTopicFault" use="literal"/>
</wsdl:fault>
<wsdl:fault name="MultipleTopicsSpecifiedFault">
  <soap:fault name="MultipleTopicsSpecifiedFault" use="literal"/>
</wsdl:fault>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="NotificationConsumerBinding" type="wsntw:NotificationConsumer">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="Notify">
    <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/NotificationConsumer/Notify"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
  </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="PullPointBinding" type="wsntw:PullPoint">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetMessages">
    <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/PullPoint/GetMessagesRequest"/>
    <wsdl:input name="GetMessagesRequest">
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="GetMessagesResponse">
      <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="ResourceUnknownFault">
      <soap:fault name="ResourceUnknownFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnableToGetMessagesFault">
      <soap:fault name="UnableToGetMessagesFault" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="DestroyPullPoint">
    <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/PullPoint/DestroyPullPointRequest"/>
    <wsdl:input name="DestroyPullPointRequest">
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="DestroyPullPointResponse">
      <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="ResourceUnknownFault">
      <soap:fault name="ResourceUnknownFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnableToDestroyPullPointFault">
      <soap:fault name="UnableToDestroyPullPointFault" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="Notify">
    <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-2/PullPoint/Notify"/>
    <wsdl:input>

```



```

        <soap:body use="literal"/>
    </wsdl:input>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="CreatePullPointBinding" type="wsntw:CreatePullPoint">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="CreatePullPoint">
        <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/2/CreatePullPoint/CreatePullPointRequest"/>
        <wsdl:input name="CreatePullPointRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="CreatePullPointResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="UnableToCreatePullPointFault">
            <soap:fault name="UnableToCreatePullPointFault" use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="PausableSubscriptionManagerBinding"
type="wsntw:PausableSubscriptionManager">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="Renew">
        <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/2/PausableSubscriptionManager/RenewRequest"/>
        <wsdl:input name="RenewRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="RenewResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="ResourceUnknownFault">
            <soap:fault name="ResourceUnknownFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="UnacceptableTerminationTimeFault">
            <soap:fault name="UnacceptableTerminationTimeFault" use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="Unsubscribe">
        <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/2/PausableSubscriptionManager/UnsubscribeRequest"/>
        <wsdl:input name="UnsubscribeRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="UnsubscribeResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="ResourceUnknownFault">
            <soap:fault name="ResourceUnknownFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="UnableToDestroySubscriptionFault">
            <soap:fault name="UnableToDestroySubscriptionFault" use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="PauseSubscription">
        <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/2/PausableSubscriptionManager/PauseSubscriptionRequest"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>

```

```

        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="ResourceUnknownFault">
        <soap:fault name="ResourceUnknownFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="PauseFailedFault">
        <soap:fault name="PauseFailedFault" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="ResumeSubscription">
    <soap:operation soapAction="http://docs.oasis-open.org/wsn/bw-
2/PausableSubscriptionManager/ResumeSubscriptionRequest"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="ResourceUnknownFault">
        <soap:fault name="ResourceUnknownFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="ResumeFailedFault">
        <soap:fault name="ResumeFailedFault" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

C.7 Imaging service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://www.onvif.org/ver20/imaging/wsdl"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:tt="http://www.onvif.org/ver10/schema" name="ImagingService"
targetNamespace="http://www.onvif.org/ver20/imaging/wsdl">
    <wsdl:types>
        <xs:schema targetNamespace="http://www.onvif.org/ver20/imaging/wsdl"
elementFormDefault="qualified">
            <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!-- Message
Request/Responses elements --><!--=====
            <xs:element name="GetImagingSettings">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="GetImagingSettingsResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="ImagingSettings" type="tt:ImagingSettings20"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element><!--=====
            <xs:element name="SetImagingSettings">

```

```

    <xs:complexType>
      <xs:sequence>
        <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
        <xs:element name="ImagingSettings" type="tt:ImagingSettings20"/>
        <xs:element name="ForcePersistence" type="xs:boolean" maxOccurs="1"
minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetImagingSettingsResponse">
    <xs:complexType>
    </xs:element><!------->
  <xs:element name="GetOptions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetOptionsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ImagingOptions" type="tt:ImagingOptions20"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element><!------->
  <xs:element name="Move">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
        <xs:element name="Focus" type="tt:FocusMove"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="MoveResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element><!------->
  <xs:element name="GetMoveOptions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetMoveOptionsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="MoveOptions" type="tt:MoveOptions20"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element><!------->
  <xs:element name="Stop">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="StopResponse">
    <xs:complexType>

```



```

        </xs:sequence/>
    </xs:complexType>
</xs:element><!------->
<xs:element name="GetStatus">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="VideoSourceToken" type="tt:ReferenceToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetStatusResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Status" type="tt:ImagingStatus20"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!------->
</xs:schema>
</wsdl:types>
<wsdl:message name="GetImagingSettingsRequest">
    <wsdl:part name="parameters" element="timg:GetImagingSettings"/>
</wsdl:message>
<wsdl:message name="GetImagingSettingsResponse">
    <wsdl:part name="parameters" element="timg:GetImagingSettingsResponse"/>
</wsdl:message>
<wsdl:message name="SetImagingSettingsRequest">
    <wsdl:part name="parameters" element="timg:SetImagingSettings"/>
</wsdl:message>
<wsdl:message name="SetImagingSettingsResponse">
    <wsdl:part name="parameters" element="timg:SetImagingSettingsResponse"/>
</wsdl:message>
<wsdl:message name="GetOptionsRequest">
    <wsdl:part name="parameters" element="timg:GetOptions"/>
</wsdl:message>
<wsdl:message name="GetOptionsResponse">
    <wsdl:part name="parameters" element="timg:GetOptionsResponse"/>
</wsdl:message>
<wsdl:message name="MoveRequest">
    <wsdl:part name="parameters" element="timg:Move"/>
</wsdl:message>
<wsdl:message name="MoveResponse">
    <wsdl:part name="parameters" element="timg:MoveResponse"/>
</wsdl:message>
<wsdl:message name="GetMoveOptionsRequest">
    <wsdl:part name="parameters" element="timg:GetMoveOptions"/>
</wsdl:message>
<wsdl:message name="GetMoveOptionsResponse">
    <wsdl:part name="parameters" element="timg:GetMoveOptionsResponse"/>
</wsdl:message>
<wsdl:message name="StopRequest">
    <wsdl:part name="parameters" element="timg:Stop"/>
</wsdl:message>
<wsdl:message name="StopResponse">
    <wsdl:part name="parameters" element="timg:StopResponse"/>
</wsdl:message>
<wsdl:message name="GetStatusRequest">
    <wsdl:part name="parameters" element="timg:GetStatus"/>
</wsdl:message>
<wsdl:message name="GetStatusResponse">
    <wsdl:part name="parameters" element="timg:GetStatusResponse"/>
</wsdl:message>
<wsdl:portType name="ImagingPort">

```

```

<wsdl:operation name="GetImagingSettings">
  <wsdl:input message="timg:GetImagingSettingsRequest"/>
  <wsdl:output message="timg:GetImagingSettingsResponse"/>
</wsdl:operation>
<wsdl:operation name="SetImagingSettings">
  <wsdl:input message="timg:SetImagingSettingsRequest"/>
  <wsdl:output message="timg:SetImagingSettingsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetOptions">
  <wsdl:input message="timg:GetOptionsRequest"/>
  <wsdl:output message="timg:GetOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="Move">
  <wsdl:input message="timg:MoveRequest"/>
  <wsdl:output message="timg:MoveResponse"/>
</wsdl:operation>
<wsdl:operation name="GetMoveOptions">
  <wsdl:input message="timg:GetMoveOptionsRequest"/>
  <wsdl:output message="timg:GetMoveOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="Stop">
  <wsdl:input message="timg:StopRequest"/>
  <wsdl:output message="timg:StopResponse"/>
</wsdl:operation>
<wsdl:operation name="GetStatus">
  <wsdl:input message="timg:GetStatusRequest"/>
  <wsdl:output message="timg:GetStatusResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ImagingBinding" type="timg:ImagingPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetImagingSettings">
    <soap:operation
soapAction="http://www.onvif.org/ver20/imaging/wsdl/GetImagingSettings"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SetImagingSettings">
    <soap:operation
soapAction="http://www.onvif.org/ver20/imaging/wsdl/SetImagingSettings"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetOptions">
    <soap:operation soapAction="http://www.onvif.org/ver20/imaging/wsdl/GetOptions"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="Move">
    <soap:operation soapAction="http://www.onvif.org/ver20/imaging/wsdl/Move"/>

```

```

<wsdl:input>
  <soap:body use="literal"/>
</wsdl:input>
<wsdl:output>
  <soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="Stop">
  <soap:operation soapAction="http://www.onvif.org/ver20/imaging/wsdl/FocusStop"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetStatus">
  <soap:operation soapAction="http://www.onvif.org/ver20/imaging/wsdl/GetStatus"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetMoveOptions">
  <soap:operation
soapAction="http://www.onvif.org/ver20/imaging/wsdl/GetMoveOptions"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

C.8 Media service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:trt="http://www.onvif.org/ver10/media/wsdl"
targetNamespace="http://www.onvif.org/ver10/media/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
targetNamespace="http://www.onvif.org/ver10/media/wsdl" elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!-- Message
Request/Responses elements --><!--=====-->
      <xs:element name="GetVideoSources">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetVideoSourcesResponse">
        <xs:complexType>

```

```

        <xs:sequence>
          <xs:element name="VideoSources" type="tt:VideoSource" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetAudioSources">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAudioSourcesResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="AudioSources" type="tt:AudioSource" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetAudioOutputs">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetAudioOutputsResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="AudioOutputs" type="tt:AudioOutput" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="CreateProfile">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Name" type="tt:Name"/>
          <xs:element name="Token" type="tt:ReferenceToken" minOccurs="0"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="CreateProfileResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Profile" type="tt:Profile"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetProfile">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetProfileResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Profile" type="tt:Profile"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetProfiles">

```

```

<xs:complexType>
  <xs:sequence/>
</xs:complexType>
</xs:element>
<xs:element name="GetProfilesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Profiles" type="tt:Profile" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!------->
<xs:element name="AddVideoEncoderConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddVideoEncoderConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->
<xs:element name="RemoveVideoEncoderConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveVideoEncoderConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->
<xs:element name="AddVideoSourceConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddVideoSourceConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->
<xs:element name="RemoveVideoSourceConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveVideoSourceConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->

```

```

<xs:element name="AddAudioEncoderConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddAudioEncoderConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->
<xs:element name="RemoveAudioEncoderConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveAudioEncoderConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->
<xs:element name="AddAudioSourceConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddAudioSourceConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->
<xs:element name="RemoveAudioSourceConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveAudioSourceConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!------->
<xs:element name="AddPTZConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddPTZConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>

```



```

</xs:element><!--=====-->
<xs:element name="RemovePTZConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemovePTZConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="AddVideoAnalyticsConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddVideoAnalyticsConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="RemoveVideoAnalyticsConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveVideoAnalyticsConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="AddMetadataConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddMetadataConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="RemoveMetadataConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveMetadataConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>

```

```

</xs:element><!--=====-->
<xs:element name="AddAudioOutputConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddAudioOutputConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="RemoveAudioOutputConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveAudioOutputConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="AddAudioDecoderConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="AddAudioDecoderConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="RemoveAudioDecoderConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="RemoveAudioDecoderConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="DeleteProfile">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="DeleteProfileResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>

```



```

</xs:element><!--=====--><!--
=====--><!--=====-->
<xs:element name="GetVideoEncoderConfigurations">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetVideoEncoderConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:VideoEncoderConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetVideoSourceConfigurations">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetVideoSourceConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:VideoSourceConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioEncoderConfigurations">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioEncoderConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:AudioEncoderConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioSourceConfigurations">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioSourceConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:AudioSourceConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetVideoAnalyticsConfigurations">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetVideoAnalyticsConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>

```

```

        <xs:element name="Configurations" type="tt:VideoAnalyticsConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
</xs:element><!------->
<xs:element name="GetMetadataConfigurations">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetMetadataConfigurationsResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configurations" type="tt:MetadataConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!-------><!------->
<xs:element name="GetAudioOutputConfigurations">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetAudioOutputConfigurationsResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configurations" type="tt:AudioOutputConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!------->
<xs:element name="GetAudioDecoderConfigurations">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="GetAudioDecoderConfigurationsResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configurations" type="tt:AudioDecoderConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!------->
<xs:element name="GetVideoSourceConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!------->
<xs:element name="GetVideoSourceConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:VideoSourceConfiguration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!------->
<xs:element name="GetVideoEncoderConfiguration">
    <xs:complexType>
        <xs:sequence>

```

```

        <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="GetVideoEncoderConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:VideoEncoderConfiguration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioSourceConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetAudioSourceConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:AudioSourceConfiguration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioEncoderConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetAudioEncoderConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:AudioEncoderConfiguration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetVideoAnalyticsConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetVideoAnalyticsConfigurationResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:VideoAnalyticsConfiguration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetMetadataConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetMetadataConfigurationResponse">
    <xs:complexType>

```

```

    <xs:sequence>
      <xs:element name="Configuration" type="tt:MetadataConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====--><!--
=====-->
<xs:element name="GetAudioOutputConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioOutputConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configuration" type="tt:AudioOutputConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioDecoderConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioDecoderConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configuration" type="tt:AudioDecoderConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCompatibleVideoEncoderConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetCompatibleVideoEncoderConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:VideoEncoderConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCompatibleVideoSourceConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetCompatibleVideoSourceConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:VideoSourceConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

</xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCompatibleAudioEncoderConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetCompatibleAudioEncoderConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:AudioEncoderConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCompatibleAudioSourceConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetCompatibleAudioSourceConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:AudioSourceConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCompatibleVideoAnalyticsConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetCompatibleVideoAnalyticsConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:VideoAnalyticsConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetCompatibleMetadataConfigurations">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetCompatibleMetadataConfigurationsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Configurations" type="tt:MetadataConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====--><!--

```

```

=====>
  <xs:element name="GetCompatibleAudioOutputConfigurations">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetCompatibleAudioOutputConfigurationsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Configurations" type="tt:AudioOutputConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element><!--=====-->
  <xs:element name="GetCompatibleAudioDecoderConfigurations">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetCompatibleAudioDecoderConfigurationsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Configurations" type="tt:AudioDecoderConfiguration"
minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element><!--=====--><!--=====--><!--=====--><!--=====-->
  <xs:element name="SetVideoEncoderConfiguration">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Configuration" type="tt:VideoEncoderConfiguration"/>
        <xs:element name="ForcePersistence" type="xs:boolean"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetVideoEncoderConfigurationResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element><!--=====-->
  <xs:element name="SetVideoSourceConfiguration">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Configuration" type="tt:VideoSourceConfiguration"/>
        <xs:element name="ForcePersistence" type="xs:boolean"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetVideoSourceConfigurationResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element><!--=====-->
  <xs:element name="SetAudioEncoderConfiguration">
    <xs:complexType>

```



```

        <xs:sequence>
            <xs:element name="Configuration" type="tt:AudioEncoderConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetAudioEncoderConfigurationResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetAudioSourceConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:AudioSourceConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetAudioSourceConfigurationResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetVideoAnalyticsConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:VideoAnalyticsConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetVideoAnalyticsConfigurationResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetMetadataConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:MetadataConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetMetadataConfigurationResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====--><!--=====-->
<xs:element name="SetAudioOutputConfiguration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:AudioOutputConfiguration"/>
            <xs:element name="ForcePersistence" type="xs:boolean"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SetAudioOutputConfigurationResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>

```

```

    </xs:complexType>
  </xs:element><!--=====>
  <xs:element name="SetAudioDecoderConfiguration">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Configuration" type="tt:AudioDecoderConfiguration"/>
        <xs:element name="ForcePersistence" type="xs:boolean"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetAudioDecoderConfigurationResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element><!--=====>
  <xs:element name="GetVideoSourceConfigurationOptions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="0"/>
        <xs:element name="ProfileToken" type="tt:ReferenceToken" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetVideoSourceConfigurationOptionsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Options" type="tt:VideoSourceConfigurationOptions"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element><!--=====>
  <xs:element name="GetVideoEncoderConfigurationOptions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="0"/>
        <xs:element name="ProfileToken" type="tt:ReferenceToken" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetVideoEncoderConfigurationOptionsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Options" type="tt:VideoEncoderConfigurationOptions"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element><!--=====>
  <xs:element name="GetAudioSourceConfigurationOptions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="0"/>
        <xs:element name="ProfileToken" type="tt:ReferenceToken" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetAudioSourceConfigurationOptionsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Options" type="tt:AudioSourceConfigurationOptions"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```



```

</xs:element><!--=====-->
<xs:element name="GetAudioEncoderConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="0"/>
      <xs:element name="ProfileToken" type="tt:ReferenceToken" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioEncoderConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Options" type="tt:AudioEncoderConfigurationOptions"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetMetadataConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="0"/>
      <xs:element name="ProfileToken" type="tt:ReferenceToken" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetMetadataConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Options" type="tt:MetadataConfigurationOptions"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioOutputConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="0"/>
      <xs:element name="ProfileToken" type="tt:ReferenceToken" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioOutputConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Options" type="tt:AudioOutputConfigurationOptions"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetAudioDecoderConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"
minOccurs="0"/>
      <xs:element name="ProfileToken" type="tt:ReferenceToken" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetAudioDecoderConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Options" type="tt:AudioDecoderConfigurationOptions"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====>
<xs:element name="GetGuaranteedNumberOfVideoEncoderInstances">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetGuaranteedNumberOfVideoEncoderInstancesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="TotalNumber" type="xs:int"/>
      <xs:element name="JPEG" type="xs:int" minOccurs="0"/>
      <xs:element name="H264" type="xs:int" minOccurs="0"/>
      <xs:element name="MPEG4" type="xs:int" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====>
<xs:element name="GetStreamUri">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="StreamSetup" type="tt:StreamSetup"/>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetStreamUriResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="MediaUri" type="tt:MediaUri"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====>
<xs:element name="StartMulticastStreaming">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="StartMulticastStreamingResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====>
<xs:element name="StopMulticastStreaming">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="StopMulticastStreamingResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====>
<xs:element name="SetSynchronizationPoint">
  <xs:complexType>
    <xs:sequence>

```

```

        <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="SetSynchronizationPointResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetSnapshotUri">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetSnapshotUriResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="MediaUri" type="tt:MediaUri"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
</xs:schema>
</wsdl:types>
<wsdl:message name="GetVideoSourcesRequest">
    <wsdl:part name="parameters" element="trt:GetVideoSources"/>
</wsdl:message>
<wsdl:message name="GetVideoSourcesResponse">
    <wsdl:part name="parameters" element="trt:GetVideoSourcesResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioSourcesRequest">
    <wsdl:part name="parameters" element="trt:GetAudioSources"/>
</wsdl:message>
<wsdl:message name="GetAudioSourcesResponse">
    <wsdl:part name="parameters" element="trt:GetAudioSourcesResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputsRequest">
    <wsdl:part name="parameters" element="trt:GetAudioOutputs"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputsResponse">
    <wsdl:part name="parameters" element="trt:GetAudioOutputsResponse"/>
</wsdl:message>
<wsdl:message name="CreateProfileRequest">
    <wsdl:part name="parameters" element="trt:CreateProfile"/>
</wsdl:message>
<wsdl:message name="CreateProfileResponse">
    <wsdl:part name="parameters" element="trt:CreateProfileResponse"/>
</wsdl:message>
<wsdl:message name="GetProfileRequest">
    <wsdl:part name="parameters" element="trt:GetProfile"/>
</wsdl:message>
<wsdl:message name="GetProfileResponse">
    <wsdl:part name="parameters" element="trt:GetProfileResponse"/>
</wsdl:message>
<wsdl:message name="GetProfilesRequest">
    <wsdl:part name="parameters" element="trt:GetProfiles"/>
</wsdl:message>
<wsdl:message name="GetProfilesResponse">
    <wsdl:part name="parameters" element="trt:GetProfilesResponse"/>
</wsdl:message>
<wsdl:message name="AddVideoEncoderConfigurationRequest">

```

```

    <wsdl:part name="parameters" element="trt:AddVideoEncoderConfiguration"/>
  </wsdl:message>
  <wsdl:message name="AddVideoEncoderConfigurationResponse">
    <wsdl:part name="parameters" element="trt:AddVideoEncoderConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="RemoveVideoEncoderConfigurationRequest">
    <wsdl:part name="parameters" element="trt:RemoveVideoEncoderConfiguration"/>
  </wsdl:message>
  <wsdl:message name="RemoveVideoEncoderConfigurationResponse">
    <wsdl:part name="parameters"
element="trt:RemoveVideoEncoderConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="AddVideoSourceConfigurationRequest">
    <wsdl:part name="parameters" element="trt:AddVideoSourceConfiguration"/>
  </wsdl:message>
  <wsdl:message name="AddVideoSourceConfigurationResponse">
    <wsdl:part name="parameters" element="trt:AddVideoSourceConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="RemoveVideoSourceConfigurationRequest">
    <wsdl:part name="parameters" element="trt:RemoveVideoSourceConfiguration"/>
  </wsdl:message>
  <wsdl:message name="RemoveVideoSourceConfigurationResponse">
    <wsdl:part name="parameters"
element="trt:RemoveVideoSourceConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="AddAudioEncoderConfigurationRequest">
    <wsdl:part name="parameters" element="trt:AddAudioEncoderConfiguration"/>
  </wsdl:message>
  <wsdl:message name="AddAudioEncoderConfigurationResponse">
    <wsdl:part name="parameters" element="trt:AddAudioEncoderConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="RemoveAudioEncoderConfigurationRequest">
    <wsdl:part name="parameters" element="trt:RemoveAudioEncoderConfiguration"/>
  </wsdl:message>
  <wsdl:message name="RemoveAudioEncoderConfigurationResponse">
    <wsdl:part name="parameters"
element="trt:RemoveAudioEncoderConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="AddAudioSourceConfigurationRequest">
    <wsdl:part name="parameters" element="trt:AddAudioSourceConfiguration"/>
  </wsdl:message>
  <wsdl:message name="AddAudioSourceConfigurationResponse">
    <wsdl:part name="parameters" element="trt:AddAudioSourceConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="RemoveAudioSourceConfigurationRequest">
    <wsdl:part name="parameters" element="trt:RemoveAudioSourceConfiguration"/>
  </wsdl:message>
  <wsdl:message name="RemoveAudioSourceConfigurationResponse">
    <wsdl:part name="parameters"
element="trt:RemoveAudioSourceConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="AddPTZConfigurationRequest">
    <wsdl:part name="parameters" element="trt:AddPTZConfiguration"/>
  </wsdl:message>
  <wsdl:message name="AddPTZConfigurationResponse">
    <wsdl:part name="parameters" element="trt:AddPTZConfigurationResponse"/>
  </wsdl:message>
  <wsdl:message name="RemovePTZConfigurationRequest">
    <wsdl:part name="parameters" element="trt:RemovePTZConfiguration"/>
  </wsdl:message>
  <wsdl:message name="RemovePTZConfigurationResponse">
    <wsdl:part name="parameters" element="trt:RemovePTZConfigurationResponse"/>
  </wsdl:message>

```

```

</wsdl:message>
<wsdl:message name="AddVideoAnalyticsConfigurationRequest">
  <wsdl:part name="parameters" element="trt:AddVideoAnalyticsConfiguration"/>
</wsdl:message>
<wsdl:message name="AddVideoAnalyticsConfigurationResponse">
  <wsdl:part name="parameters" element="trt:AddVideoAnalyticsConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="RemoveVideoAnalyticsConfigurationRequest">
  <wsdl:part name="parameters" element="trt:RemoveVideoAnalyticsConfiguration"/>
</wsdl:message>
<wsdl:message name="RemoveVideoAnalyticsConfigurationResponse">
  <wsdl:part name="parameters"
element="trt:RemoveVideoAnalyticsConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="AddMetadataConfigurationRequest">
  <wsdl:part name="parameters" element="trt:AddMetadataConfiguration"/>
</wsdl:message>
<wsdl:message name="AddMetadataConfigurationResponse">
  <wsdl:part name="parameters" element="trt:AddMetadataConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="RemoveMetadataConfigurationRequest">
  <wsdl:part name="parameters" element="trt:RemoveMetadataConfiguration"/>
</wsdl:message>
<wsdl:message name="RemoveMetadataConfigurationResponse">
  <wsdl:part name="parameters" element="trt:RemoveMetadataConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="AddAudioOutputConfigurationRequest">
  <wsdl:part name="parameters" element="trt:AddAudioOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="AddAudioOutputConfigurationResponse">
  <wsdl:part name="parameters" element="trt:AddAudioOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="RemoveAudioOutputConfigurationRequest">
  <wsdl:part name="parameters" element="trt:RemoveAudioOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="RemoveAudioOutputConfigurationResponse">
  <wsdl:part name="parameters"
element="trt:RemoveAudioOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="AddAudioDecoderConfigurationRequest">
  <wsdl:part name="parameters" element="trt:AddAudioDecoderConfiguration"/>
</wsdl:message>
<wsdl:message name="AddAudioDecoderConfigurationResponse">
  <wsdl:part name="parameters" element="trt:AddAudioDecoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="RemoveAudioDecoderConfigurationRequest">
  <wsdl:part name="parameters" element="trt:RemoveAudioDecoderConfiguration"/>
</wsdl:message>
<wsdl:message name="RemoveAudioDecoderConfigurationResponse">
  <wsdl:part name="parameters"
element="trt:RemoveAudioDecoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="DeleteProfileRequest">
  <wsdl:part name="parameters" element="trt:DeleteProfile"/>
</wsdl:message>
<wsdl:message name="DeleteProfileResponse">
  <wsdl:part name="parameters" element="trt:DeleteProfileResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetVideoSourceConfigurations"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationsResponse">

```



```
<wsdl:part name="parameters" element="trt:GetVideoSourceConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoEncoderConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetVideoEncoderConfigurations"/>
</wsdl:message>
<wsdl:message name="GetVideoEncoderConfigurationsResponse">
  <wsdl:part name="parameters" element="trt:GetVideoEncoderConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetAudioSourceConfigurations"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationsResponse">
  <wsdl:part name="parameters" element="trt:GetAudioSourceConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioEncoderConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetAudioEncoderConfigurations"/>
</wsdl:message>
<wsdl:message name="GetAudioEncoderConfigurationsResponse">
  <wsdl:part name="parameters" element="trt:GetAudioEncoderConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoAnalyticsConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetVideoAnalyticsConfigurations"/>
</wsdl:message>
<wsdl:message name="GetVideoAnalyticsConfigurationsResponse">
  <wsdl:part name="parameters" element="trt:GetVideoAnalyticsConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetMetadataConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetMetadataConfigurations"/>
</wsdl:message>
<wsdl:message name="GetMetadataConfigurationsResponse">
  <wsdl:part name="parameters" element="trt:GetMetadataConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetAudioOutputConfigurations"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationsResponse">
  <wsdl:part name="parameters" element="trt:GetAudioOutputConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioDecoderConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetAudioDecoderConfigurations"/>
</wsdl:message>
<wsdl:message name="GetAudioDecoderConfigurationsResponse">
  <wsdl:part name="parameters" element="trt:GetAudioDecoderConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetVideoSourceConfiguration"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetVideoSourceConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoEncoderConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetVideoEncoderConfiguration"/>
</wsdl:message>
<wsdl:message name="GetVideoEncoderConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetVideoEncoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetAudioSourceConfiguration"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetAudioSourceConfigurationResponse"/>
</wsdl:message>
```

```

<wsdl:message name="GetAudioEncoderConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetAudioEncoderConfiguration"/>
</wsdl:message>
<wsdl:message name="GetAudioEncoderConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetAudioEncoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoAnalyticsConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetVideoAnalyticsConfiguration"/>
</wsdl:message>
<wsdl:message name="GetVideoAnalyticsConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetVideoAnalyticsConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetMetadataConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetMetadataConfiguration"/>
</wsdl:message>
<wsdl:message name="GetMetadataConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetMetadataConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetAudioOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetAudioOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioDecoderConfigurationRequest">
  <wsdl:part name="parameters" element="trt:GetAudioDecoderConfiguration"/>
</wsdl:message>
<wsdl:message name="GetAudioDecoderConfigurationResponse">
  <wsdl:part name="parameters" element="trt:GetAudioDecoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleVideoEncoderConfigurationsRequest">
  <wsdl:part name="parameters"
element="trt:GetCompatibleVideoEncoderConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleVideoEncoderConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleVideoEncoderConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleVideoSourceConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetCompatibleVideoSourceConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleVideoSourceConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleVideoSourceConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioEncoderConfigurationsRequest">
  <wsdl:part name="parameters"
element="trt:GetCompatibleAudioEncoderConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioEncoderConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleAudioEncoderConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioSourceConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetCompatibleAudioSourceConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioSourceConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleAudioSourceConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleVideoAnalyticsConfigurationsRequest">
  <wsdl:part name="parameters"

```

```

element="trt:GetCompatibleVideoAnalyticsConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleVideoAnalyticsConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleVideoAnalyticsConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleMetadataConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetCompatibleMetadataConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleMetadataConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleMetadataConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioOutputConfigurationsRequest">
  <wsdl:part name="parameters" element="trt:GetCompatibleAudioOutputConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioOutputConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleAudioOutputConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioDecoderConfigurationsRequest">
  <wsdl:part name="parameters"
element="trt:GetCompatibleAudioDecoderConfigurations"/>
</wsdl:message>
<wsdl:message name="GetCompatibleAudioDecoderConfigurationsResponse">
  <wsdl:part name="parameters"
element="trt:GetCompatibleAudioDecoderConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="SetVideoSourceConfigurationRequest">
  <wsdl:part name="parameters" element="trt:SetVideoSourceConfiguration"/>
</wsdl:message>
<wsdl:message name="SetVideoSourceConfigurationResponse">
  <wsdl:part name="parameters" element="trt:SetVideoSourceConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetVideoEncoderConfigurationRequest">
  <wsdl:part name="parameters" element="trt:SetVideoEncoderConfiguration"/>
</wsdl:message>
<wsdl:message name="SetVideoEncoderConfigurationResponse">
  <wsdl:part name="parameters" element="trt:SetVideoEncoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetAudioSourceConfigurationRequest">
  <wsdl:part name="parameters" element="trt:SetAudioSourceConfiguration"/>
</wsdl:message>
<wsdl:message name="SetAudioSourceConfigurationResponse">
  <wsdl:part name="parameters" element="trt:SetAudioSourceConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetAudioEncoderConfigurationRequest">
  <wsdl:part name="parameters" element="trt:SetAudioEncoderConfiguration"/>
</wsdl:message>
<wsdl:message name="SetAudioEncoderConfigurationResponse">
  <wsdl:part name="parameters" element="trt:SetAudioEncoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetVideoAnalyticsConfigurationRequest">
  <wsdl:part name="parameters" element="trt:SetVideoAnalyticsConfiguration"/>
</wsdl:message>
<wsdl:message name="SetVideoAnalyticsConfigurationResponse">
  <wsdl:part name="parameters" element="trt:SetVideoAnalyticsConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetMetadataConfigurationRequest">
  <wsdl:part name="parameters" element="trt:SetMetadataConfiguration"/>
</wsdl:message>
<wsdl:message name="SetMetadataConfigurationResponse">

```



```

    <wsdl:part name="parameters" element="trt:SetMetadataConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetAudioOutputConfigurationRequest">
    <wsdl:part name="parameters" element="trt:SetAudioOutputConfiguration"/>
</wsdl:message>
<wsdl:message name="SetAudioOutputConfigurationResponse">
    <wsdl:part name="parameters" element="trt:SetAudioOutputConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetAudioDecoderConfigurationRequest">
    <wsdl:part name="parameters" element="trt:SetAudioDecoderConfiguration"/>
</wsdl:message>
<wsdl:message name="SetAudioDecoderConfigurationResponse">
    <wsdl:part name="parameters" element="trt:SetAudioDecoderConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationOptionsRequest">
    <wsdl:part name="parameters" element="trt:GetVideoSourceConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetVideoSourceConfigurationOptionsResponse">
    <wsdl:part name="parameters"
element="trt:GetVideoSourceConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetVideoEncoderConfigurationOptionsRequest">
    <wsdl:part name="parameters" element="trt:GetVideoEncoderConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetVideoEncoderConfigurationOptionsResponse">
    <wsdl:part name="parameters"
element="trt:GetVideoEncoderConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationOptionsRequest">
    <wsdl:part name="parameters" element="trt:GetAudioSourceConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetAudioSourceConfigurationOptionsResponse">
    <wsdl:part name="parameters"
element="trt:GetAudioSourceConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioEncoderConfigurationOptionsRequest">
    <wsdl:part name="parameters" element="trt:GetAudioEncoderConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetAudioEncoderConfigurationOptionsResponse">
    <wsdl:part name="parameters"
element="trt:GetAudioEncoderConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetMetadataConfigurationOptionsRequest">
    <wsdl:part name="parameters" element="trt:GetMetadataConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetMetadataConfigurationOptionsResponse">
    <wsdl:part name="parameters"
element="trt:GetMetadataConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationOptionsRequest">
    <wsdl:part name="parameters" element="trt:GetAudioOutputConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetAudioOutputConfigurationOptionsResponse">
    <wsdl:part name="parameters"
element="trt:GetAudioOutputConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetAudioDecoderConfigurationOptionsRequest">
    <wsdl:part name="parameters" element="trt:GetAudioDecoderConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetAudioDecoderConfigurationOptionsResponse">
    <wsdl:part name="parameters"
element="trt:GetAudioDecoderConfigurationOptionsResponse"/>

```

```

</wsdl:message>
<wsdl:message name="GetGuaranteedNumberOfVideoEncoderInstancesRequest">
  <wsdl:part name="parameters"
element="trt:GetGuaranteedNumberOfVideoEncoderInstances"/>
</wsdl:message>
<wsdl:message name="GetGuaranteedNumberOfVideoEncoderInstancesResponse">
  <wsdl:part name="parameters"
element="trt:GetGuaranteedNumberOfVideoEncoderInstancesResponse"/>
</wsdl:message>
<wsdl:message name="GetStreamUriRequest">
  <wsdl:part name="parameters" element="trt:GetStreamUri"/>
</wsdl:message>
<wsdl:message name="GetStreamUriResponse">
  <wsdl:part name="parameters" element="trt:GetStreamUriResponse"/>
</wsdl:message>
<wsdl:message name="StartMulticastStreamingRequest">
  <wsdl:part name="parameters" element="trt:StartMulticastStreaming"/>
</wsdl:message>
<wsdl:message name="StartMulticastStreamingResponse">
  <wsdl:part name="parameters" element="trt:StartMulticastStreamingResponse"/>
</wsdl:message>
<wsdl:message name="StopMulticastStreamingRequest">
  <wsdl:part name="parameters" element="trt:StopMulticastStreaming"/>
</wsdl:message>
<wsdl:message name="StopMulticastStreamingResponse">
  <wsdl:part name="parameters" element="trt:StopMulticastStreamingResponse"/>
</wsdl:message>
<wsdl:message name="SetSynchronizationPointRequest">
  <wsdl:part name="parameters" element="trt:SetSynchronizationPoint"/>
</wsdl:message>
<wsdl:message name="SetSynchronizationPointResponse">
  <wsdl:part name="parameters" element="trt:SetSynchronizationPointResponse"/>
</wsdl:message>
<wsdl:message name="GetSnapshotUriRequest">
  <wsdl:part name="parameters" element="trt:GetSnapshotUri"/>
</wsdl:message>
<wsdl:message name="GetSnapshotUriResponse">
  <wsdl:part name="parameters" element="trt:GetSnapshotUriResponse"/>
</wsdl:message>
<wsdl:portType name="Media"><!--=====--><!--
=====--><!--=====-->
  <wsdl:operation name="GetVideoSources">
    <wsdl:input message="trt:GetVideoSourcesRequest"/>
    <wsdl:output message="trt:GetVideoSourcesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAudioSources">
    <wsdl:input message="trt:GetAudioSourcesRequest"/>
    <wsdl:output message="trt:GetAudioSourcesResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetAudioOutputs">
    <wsdl:input message="trt:GetAudioOutputsRequest"/>
    <wsdl:output message="trt:GetAudioOutputsResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="CreateProfile">
    <wsdl:input message="trt:CreateProfileRequest"/>
    <wsdl:output message="trt:CreateProfileResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetProfile">
    <wsdl:input message="trt:GetProfileRequest"/>
    <wsdl:output message="trt:GetProfileResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetProfiles">

```

```

    <wsdl:input message="trt:GetProfilesRequest"/>
    <wsdl:output message="trt:GetProfilesResponse"/>
</wsdl:operation>
<wsdl:operation name="AddVideoEncoderConfiguration">
    <wsdl:input message="trt:AddVideoEncoderConfigurationRequest"/>
    <wsdl:output message="trt:AddVideoEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveVideoEncoderConfiguration">
    <wsdl:input message="trt:RemoveVideoEncoderConfigurationRequest"/>
    <wsdl:output message="trt:RemoveVideoEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddVideoSourceConfiguration">
    <wsdl:input message="trt:AddVideoSourceConfigurationRequest"/>
    <wsdl:output message="trt:AddVideoSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveVideoSourceConfiguration">
    <wsdl:input message="trt:RemoveVideoSourceConfigurationRequest"/>
    <wsdl:output message="trt:RemoveVideoSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddAudioEncoderConfiguration">
    <wsdl:input message="trt:AddAudioEncoderConfigurationRequest"/>
    <wsdl:output message="trt:AddAudioEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveAudioEncoderConfiguration">
    <wsdl:input message="trt:RemoveAudioEncoderConfigurationRequest"/>
    <wsdl:output message="trt:RemoveAudioEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddAudioSourceConfiguration">
    <wsdl:input message="trt:AddAudioSourceConfigurationRequest"/>
    <wsdl:output message="trt:AddAudioSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveAudioSourceConfiguration">
    <wsdl:input message="trt:RemoveAudioSourceConfigurationRequest"/>
    <wsdl:output message="trt:RemoveAudioSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddPTZConfiguration">
    <wsdl:input message="trt:AddPTZConfigurationRequest"/>
    <wsdl:output message="trt:AddPTZConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemovePTZConfiguration">
    <wsdl:input message="trt:RemovePTZConfigurationRequest"/>
    <wsdl:output message="trt:RemovePTZConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddVideoAnalyticsConfiguration">
    <wsdl:input message="trt:AddVideoAnalyticsConfigurationRequest"/>
    <wsdl:output message="trt:AddVideoAnalyticsConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveVideoAnalyticsConfiguration">
    <wsdl:input message="trt:RemoveVideoAnalyticsConfigurationRequest"/>
    <wsdl:output message="trt:RemoveVideoAnalyticsConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddMetadataConfiguration">
    <wsdl:input message="trt:AddMetadataConfigurationRequest"/>
    <wsdl:output message="trt:AddMetadataConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveMetadataConfiguration">
    <wsdl:input message="trt:RemoveMetadataConfigurationRequest"/>
    <wsdl:output message="trt:RemoveMetadataConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddAudioOutputConfiguration">
    <wsdl:input message="trt:AddAudioOutputConfigurationRequest"/>
    <wsdl:output message="trt:AddAudioOutputConfigurationResponse"/>

```

```

</wsdl:operation>
<wsdl:operation name="RemoveAudioOutputConfiguration">
  <wsdl:input message="trt:RemoveAudioOutputConfigurationRequest"/>
  <wsdl:output message="trt:RemoveAudioOutputConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="AddAudioDecoderConfiguration">
  <wsdl:input message="trt:AddAudioDecoderConfigurationRequest"/>
  <wsdl:output message="trt:AddAudioDecoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="RemoveAudioDecoderConfiguration">
  <wsdl:input message="trt:RemoveAudioDecoderConfigurationRequest"/>
  <wsdl:output message="trt:RemoveAudioDecoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="DeleteProfile">
  <wsdl:input message="trt:DeleteProfileRequest"/>
  <wsdl:output message="trt:DeleteProfileResponse"/>
</wsdl:operation> <!--=====-->
<wsdl:operation name="GetVideoSourceConfigurations">
  <wsdl:input message="trt:GetVideoSourceConfigurationsRequest"/>
  <wsdl:output message="trt:GetVideoSourceConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoEncoderConfigurations">
  <wsdl:input message="trt:GetVideoEncoderConfigurationsRequest"/>
  <wsdl:output message="trt:GetVideoEncoderConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioSourceConfigurations">
  <wsdl:input message="trt:GetAudioSourceConfigurationsRequest"/>
  <wsdl:output message="trt:GetAudioSourceConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioEncoderConfigurations">
  <wsdl:input message="trt:GetAudioEncoderConfigurationsRequest"/>
  <wsdl:output message="trt:GetAudioEncoderConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoAnalyticsConfigurations">
  <wsdl:input message="trt:GetVideoAnalyticsConfigurationsRequest"/>
  <wsdl:output message="trt:GetVideoAnalyticsConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetMetadataConfigurations">
  <wsdl:input message="trt:GetMetadataConfigurationsRequest"/>
  <wsdl:output message="trt:GetMetadataConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioOutputConfigurations">
  <wsdl:input message="trt:GetAudioOutputConfigurationsRequest"/>
  <wsdl:output message="trt:GetAudioOutputConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioDecoderConfigurations">
  <wsdl:input message="trt:GetAudioDecoderConfigurationsRequest"/>
  <wsdl:output message="trt:GetAudioDecoderConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoSourceConfiguration">
  <wsdl:input message="trt:GetVideoSourceConfigurationRequest"/>
  <wsdl:output message="trt:GetVideoSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoEncoderConfiguration">
  <wsdl:input message="trt:GetVideoEncoderConfigurationRequest"/>
  <wsdl:output message="trt:GetVideoEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioSourceConfiguration">
  <wsdl:input message="trt:GetAudioSourceConfigurationRequest"/>
  <wsdl:output message="trt:GetAudioSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioEncoderConfiguration">

```



```

    <wsdl:input message="trt:GetAudioEncoderConfigurationRequest"/>
    <wsdl:output message="trt:GetAudioEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoAnalyticsConfiguration">
    <wsdl:input message="trt:GetVideoAnalyticsConfigurationRequest"/>
    <wsdl:output message="trt:GetVideoAnalyticsConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetMetadataConfiguration">
    <wsdl:input message="trt:GetMetadataConfigurationRequest"/>
    <wsdl:output message="trt:GetMetadataConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioOutputConfiguration">
    <wsdl:input message="trt:GetAudioOutputConfigurationRequest"/>
    <wsdl:output message="trt:GetAudioOutputConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioDecoderConfiguration">
    <wsdl:input message="trt:GetAudioDecoderConfigurationRequest"/>
    <wsdl:output message="trt:GetAudioDecoderConfigurationResponse"/>
</wsdl:operation><!--=====-->
<wsdl:operation name="GetCompatibleVideoEncoderConfigurations">
    <wsdl:input message="trt:GetCompatibleVideoEncoderConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleVideoEncoderConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCompatibleVideoSourceConfigurations">
    <wsdl:input message="trt:GetCompatibleVideoSourceConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleVideoSourceConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCompatibleAudioEncoderConfigurations">
    <wsdl:input message="trt:GetCompatibleAudioEncoderConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleAudioEncoderConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCompatibleAudioSourceConfigurations">
    <wsdl:input message="trt:GetCompatibleAudioSourceConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleAudioSourceConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCompatibleVideoAnalyticsConfigurations">
    <wsdl:input message="trt:GetCompatibleVideoAnalyticsConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleVideoAnalyticsConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCompatibleMetadataConfigurations">
    <wsdl:input message="trt:GetCompatibleMetadataConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleMetadataConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCompatibleAudioOutputConfigurations">
    <wsdl:input message="trt:GetCompatibleAudioOutputConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleAudioOutputConfigurationsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCompatibleAudioDecoderConfigurations">
    <wsdl:input message="trt:GetCompatibleAudioDecoderConfigurationsRequest"/>
    <wsdl:output message="trt:GetCompatibleAudioDecoderConfigurationsResponse"/>
</wsdl:operation><!--=====-->
<wsdl:operation name="SetVideoSourceConfiguration">
    <wsdl:input message="trt:SetVideoSourceConfigurationRequest"/>
    <wsdl:output message="trt:SetVideoSourceConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetVideoEncoderConfiguration">
    <wsdl:input message="trt:SetVideoEncoderConfigurationRequest"/>
    <wsdl:output message="trt:SetVideoEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetAudioSourceConfiguration">
    <wsdl:input message="trt:SetAudioSourceConfigurationRequest"/>
    <wsdl:output message="trt:SetAudioSourceConfigurationResponse"/>

```

```

</wsdl:operation>
<wsdl:operation name="SetAudioEncoderConfiguration">
  <wsdl:input message="trt:SetAudioEncoderConfigurationRequest"/>
  <wsdl:output message="trt:SetAudioEncoderConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetVideoAnalyticsConfiguration">
  <wsdl:input message="trt:SetVideoAnalyticsConfigurationRequest"/>
  <wsdl:output message="trt:SetVideoAnalyticsConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetMetadataConfiguration">
  <wsdl:input message="trt:SetMetadataConfigurationRequest"/>
  <wsdl:output message="trt:SetMetadataConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetAudioOutputConfiguration">
  <wsdl:input message="trt:SetAudioOutputConfigurationRequest"/>
  <wsdl:output message="trt:SetAudioOutputConfigurationResponse"/>
</wsdl:operation>
<wsdl:operation name="SetAudioDecoderConfiguration">
  <wsdl:input message="trt:SetAudioDecoderConfigurationRequest"/>
  <wsdl:output message="trt:SetAudioDecoderConfigurationResponse"/>
</wsdl:operation><!--=====-->
<wsdl:operation name="GetVideoSourceConfigurationOptions">
  <wsdl:input message="trt:GetVideoSourceConfigurationOptionsRequest"/>
  <wsdl:output message="trt:GetVideoSourceConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetVideoEncoderConfigurationOptions">
  <wsdl:input message="trt:GetVideoEncoderConfigurationOptionsRequest"/>
  <wsdl:output message="trt:GetVideoEncoderConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioSourceConfigurationOptions">
  <wsdl:input message="trt:GetAudioSourceConfigurationOptionsRequest"/>
  <wsdl:output message="trt:GetAudioSourceConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioEncoderConfigurationOptions">
  <wsdl:input message="trt:GetAudioEncoderConfigurationOptionsRequest"/>
  <wsdl:output message="trt:GetAudioEncoderConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetMetadataConfigurationOptions">
  <wsdl:input message="trt:GetMetadataConfigurationOptionsRequest"/>
  <wsdl:output message="trt:GetMetadataConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioOutputConfigurationOptions">
  <wsdl:input message="trt:GetAudioOutputConfigurationOptionsRequest"/>
  <wsdl:output message="trt:GetAudioOutputConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="GetAudioDecoderConfigurationOptions">
  <wsdl:input message="trt:GetAudioDecoderConfigurationOptionsRequest"/>
  <wsdl:output message="trt:GetAudioDecoderConfigurationOptionsResponse"/>
</wsdl:operation><!--=====-->
<wsdl:operation name="GetGuaranteedNumberOfVideoEncoderInstances">
  <wsdl:input message="trt:GetGuaranteedNumberOfVideoEncoderInstancesRequest"/>
  <wsdl:output
message="trt:GetGuaranteedNumberOfVideoEncoderInstancesResponse"/>
</wsdl:operation><!--=====-->
<wsdl:operation name="GetStreamUri">
  <wsdl:input message="trt:GetStreamUriRequest"/>
  <wsdl:output message="trt:GetStreamUriResponse"/>
</wsdl:operation>
<wsdl:operation name="StartMulticastStreaming">
  <wsdl:input message="trt:StartMulticastStreamingRequest"/>
  <wsdl:output message="trt:StartMulticastStreamingResponse"/>
</wsdl:operation>

```

```

<wsdl:operation name="StopMulticastStreaming">
  <wsdl:input message="trt:StopMulticastStreamingRequest"/>
  <wsdl:output message="trt:StopMulticastStreamingResponse"/>
</wsdl:operation>
<wsdl:operation name="SetSynchronizationPoint">
  <wsdl:input message="trt:SetSynchronizationPointRequest"/>
  <wsdl:output message="trt:SetSynchronizationPointResponse"/>
</wsdl:operation>
<wsdl:operation name="GetSnapshotUri">
  <wsdl:input message="trt:GetSnapshotUriRequest"/>
  <wsdl:output message="trt:GetSnapshotUriResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="MediaBinding" type="trt:Media">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/><!--
=====--><!--=====--><!--
=====-->
  <wsdl:operation name="GetVideoSources">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoSources"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation><!--=====--><!--
=====-->
  <wsdl:operation name="GetAudioSources">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioSources"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="GetAudioOutputs">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioOutputs"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="CreateProfile">
    <soap:operation soapAction="http://www.onvif.org/ver10/media/wsdl/CreateProfile"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="GetProfile">
    <soap:operation soapAction="http://www.onvif.org/ver10/media/wsdl/GetProfile"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
  </wsdl:operation>

```



```

        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation><!--=====-->
      <wsdl:operation name="GetProfiles">
        <soap:operation soapAction="http://www.onvif.org/ver10/media/wsdl/GetProfiles"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation><!--=====-->
      <wsdl:operation name="AddVideoEncoderConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddVideoEncoderConfiguration"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation><!--=====-->
      <wsdl:operation name="AddVideoSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddVideoSourceConfiguration"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation><!--=====-->
      <wsdl:operation name="AddAudioEncoderConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddAudioEncoderConfiguration"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation><!--=====-->
      <wsdl:operation name="AddAudioSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddAudioSourceConfiguration"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation><!--=====-->
      <wsdl:operation name="AddPTZConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddPTZConfiguration"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation>

```

```

</wsdl:operation><!--=====-->
<wsdl:operation name="AddVideoAnalyticsConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddVideoAnalyticsConfiguration"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="AddMetadataConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddMetadataConfiguration"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="AddAudioOutputConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddAudioOutputConfiguration"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="AddAudioDecoderConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/AddAudioDecoderConfiguration"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveVideoEncoderConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveVideoEncoderConfiguration"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveVideoSourceConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveVideoSourceConfiguration"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveAudioEncoderConfiguration">

```

```

    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveAudioEncoderConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveAudioSourceConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveAudioSourceConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemovePTZConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemovePTZConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveVideoAnalyticsConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveVideoAnalyticsConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveMetadataConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveMetadataConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveAudioOutputConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveAudioOutputConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="RemoveAudioDecoderConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/RemoveAudioDecoderConfiguration"/>

```

```

<wsdl:input>
  <soap:body parts="parameters" use="literal"/>
</wsdl:input>
<wsdl:output>
  <soap:body parts="parameters" use="literal"/>
</wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="DeleteProfile">
  <soap:operation soapAction="http://www.onvif.org/ver10/media/wsdl/DeleteProfile"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====--><!--=====-->
=====
<wsdl:operation name="GetVideoSourceConfigurations">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoSourceConfigurations"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetVideoEncoderConfigurations">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoEncoderConfigurations"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAudioSourceConfigurations">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioSourceConfigurations"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAudioEncoderConfigurations">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioEncoderConfigurations"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetVideoAnalyticsConfigurations">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoAnalyticsConfigurations"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>

```

```

        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetMetadataConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetMetadataConfigurations"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioOutputConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioOutputConfigurations"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioDecoderConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioDecoderConfigurations"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation> <!--=====-->
    <wsdl:operation name="GetVideoSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoSourceConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetVideoEncoderConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoEncoderConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioSourceConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>

```

```

        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAudioEncoderConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioEncoderConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetVideoAnalyticsConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoAnalyticsConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetMetadataConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetMetadataConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAudioOutputConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioOutputConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAudioDecoderConfiguration">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioDecoderConfiguration"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation> <!--=====-->
<wsdl:operation name="GetCompatibleVideoEncoderConfigurations">
    <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleVideoEncoderConfigurations"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>

```



```

        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCompatibleVideoSourceConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleVideoSourceConfiguration
s"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCompatibleAudioEncoderConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleAudioEncoderConfiguratio
ns"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCompatibleAudioSourceConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleAudioSourceConfiguration
s"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCompatibleVideoAnalyticsConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleVideoAnalyticsConfiguratio
ns"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCompatibleMetadataConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleMetadataConfigurations"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCompatibleAudioOutputConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleAudioOutputConfigurations
"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>

```



```

        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCompatibleAudioDecoderConfigurations">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetCompatibleAudioDecoderConfiguratio
ns"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="SetVideoSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/SetVideoSourceConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetVideoEncoderConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/SetVideoEncoderConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetAudioSourceConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/SetAudioSourceConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetAudioEncoderConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/SetAudioEncoderConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetVideoAnalyticsConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/SetVideoAnalyticsConfiguration"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
    
```

```

    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SetMetadataConfiguration">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/media/wsdl/SetMetadataConfiguration"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SetAudioOutputConfiguration">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/media/wsdl/SetAudioOutputConfiguration"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SetAudioDecoderConfiguration">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/media/wsdl/SetAudioDecoderConfiguration"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation> <!--=====-->
  <wsdl:operation name="GetVideoSourceConfigurationOptions">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoSourceConfigurationOptions"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetVideoEncoderConfigurationOptions">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/media/wsdl/GetVideoEncoderConfigurationOptions"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetAudioSourceConfigurationOptions">
    <soap:operation
      soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioSourceConfigurationOptions"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
  </wsdl:operation>

```

```

        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioEncoderConfigurationOptions">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioEncoderConfigurationOptions"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetMetadataConfigurationOptions">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetMetadataConfigurationOptions"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioOutputConfigurationOptions">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioOutputConfigurationOptions"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAudioDecoderConfigurationOptions">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetAudioDecoderConfigurationOptions"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation> <!--=====-->
    <wsdl:operation name="GetGuaranteedNumberOfVideoEncoderInstances">
        <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/GetGuaranteedNumberOfVideoEncoderIn
stances"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation> <!--=====--> <!--
=====--> <!--=====-->
    <wsdl:operation name="GetStreamUri">
        <soap:operation soapAction="http://www.onvif.org/ver10/media/wsdl/GetStreamUri"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

</wsdl:operation><!--=====-->
<wsdl:operation name="StartMulticastStreaming">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/StartMulticastStreaming"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="StopMulticastStreaming">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/StopMulticastStreaming"/>
  <wsdl:input>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="parameters" use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="SetSynchronizationPoint">
  <soap:operation
soapAction="http://www.onvif.org/ver10/media/wsdl/SetSynchronizationPoint"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
<wsdl:operation name="GetSnapshotUri">
  <soap:operation soapAction="http://www.onvif.org/ver10/media/wsdl/GetSnapshotUri"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation><!--=====-->
</wsdl:binding>
</wsdl:definitions>

```

C.9 PTZ service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:ptz="http://www.onvif.org/ver20/ptz/wsdl"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/" name="PTZService"
targetNamespace="http://www.onvif.org/ver20/ptz/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
targetNamespace="http://www.onvif.org/ver20/ptz/wsdl" elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!-- Message
Request/Responses elements --><!--=====-->
      <xs:element name="GetNodes">

```

```

        <xs:complexType/>
    </xs:element>
    <xs:element name="GetNodesResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="PTZNode" type="tt:PTZNode" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetNode">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="NodeToken" type="tt:ReferenceToken"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="GetNodeResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="PTZNode" type="tt:PTZNode"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetConfigurations">
        <xs:complexType/>
    </xs:element>
    <xs:element name="GetConfigurationsResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="PTZConfiguration" type="tt:PTZConfiguration" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="GetConfiguration">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="PTZConfigurationToken" type="tt:ReferenceToken"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="GetConfigurationResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="PTZConfiguration" type="tt:PTZConfiguration"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element><!--=====-->
    <xs:element name="SetConfiguration">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="PTZConfiguration" type="tt:PTZConfiguration"/>
                <xs:element name="ForcePersistence" type="xs:boolean"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="SetConfigurationResponse">
        <xs:complexType>
            <xs:sequence minOccurs="0" maxOccurs="1"/>
        </xs:complexType>
    </xs:element><!--=====-->

```

```

<xs:element name="GetConfigurationOptions">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ConfigurationToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetConfigurationOptionsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PTZConfigurationOptions"
type="tt:PTZConfigurationOptions"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SendAuxiliaryCommand">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="AuxiliaryData" type="tt:AuxiliaryData"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SendAuxiliaryCommandResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AuxiliaryResponse" type="tt:AuxiliaryData"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetPresets">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetPresetsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Preset" type="tt:PTZPreset" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetPreset">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="PresetName" type="xs:string" minOccurs="0"/>
      <xs:element name="PresetToken" type="tt:ReferenceToken" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetPresetResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PresetToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="RemovePreset">

```



```

<xs:complexType>
  <xs:sequence>
    <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    <xs:element name="PresetToken" type="tt:ReferenceToken"/>
  </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="RemovePresetResponse">
  <xs:complexType/>
</xs:element><!--=====-->
<xs:element name="GotoPreset">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="PresetToken" type="tt:ReferenceToken"/>
      <xs:element name="Speed" type="tt:PTZSpeed" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GotoPresetResponse">
  <xs:complexType/>
</xs:element><!--=====-->
<xs:element name="GetStatus">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetStatusResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PTZStatus" type="tt:PTZStatus"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GotoHomePosition">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
      <xs:element name="Speed" type="tt:PTZSpeed" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GotoHomePositionResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="SetHomePosition">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetHomePositionResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="ContinuousMove">

```



```

    <xs:complexType>
      <xs:sequence>
        <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
        <xs:element name="Velocity" type="tt:PTZSpeed"/>
        <xs:element name="Timeout" type="xs:duration" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="ContinuousMoveResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element> <!--=====-->
  <xs:element name="RelativeMove">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
        <xs:element name="Translation" type="tt:PTZVector"/>
        <xs:element name="Speed" type="tt:PTZSpeed" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="RelativeMoveResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element> <!--=====-->
  <xs:element name="AbsoluteMove">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
        <xs:element name="Position" type="tt:PTZVector"/>
        <xs:element name="Speed" type="tt:PTZSpeed" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="AbsoluteMoveResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element> <!--=====-->
  <xs:element name="Stop">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ProfileToken" type="tt:ReferenceToken"/>
        <xs:element name="PanTilt" type="xs:boolean" minOccurs="0"/>
        <xs:element name="Zoom" type="xs:boolean" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="StopResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element> <!--=====-->
</xs:schema>
</wsdl:types>
<wsdl:message name="GetNodesRequest">
  <wsdl:part name="parameters" element="tptz:GetNodes"/>
</wsdl:message>
<wsdl:message name="GetNodesResponse">
  <wsdl:part name="parameters" element="tptz:GetNodesResponse"/>

```

```

</wsdl:message>
<wsdl:message name="GetNodeRequest">
  <wsdl:part name="parameters" element="tptz:GetNode"/>
</wsdl:message>
<wsdl:message name="GetNodeResponse">
  <wsdl:part name="parameters" element="tptz:GetNodeResponse"/>
</wsdl:message>
<wsdl:message name="GetConfigurationsRequest">
  <wsdl:part name="parameters" element="tptz:GetConfigurations"/>
</wsdl:message>
<wsdl:message name="GetConfigurationsResponse">
  <wsdl:part name="parameters" element="tptz:GetConfigurationsResponse"/>
</wsdl:message>
<wsdl:message name="GetConfigurationRequest">
  <wsdl:part name="parameters" element="tptz:GetConfiguration"/>
</wsdl:message>
<wsdl:message name="GetConfigurationResponse">
  <wsdl:part name="parameters" element="tptz:GetConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetConfigurationRequest">
  <wsdl:part name="parameters" element="tptz:SetConfiguration"/>
</wsdl:message>
<wsdl:message name="SetConfigurationResponse">
  <wsdl:part name="parameters" element="tptz:SetConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetConfigurationOptionsRequest">
  <wsdl:part name="parameters" element="tptz:GetConfigurationOptions"/>
</wsdl:message>
<wsdl:message name="GetConfigurationOptionsResponse">
  <wsdl:part name="parameters" element="tptz:GetConfigurationOptionsResponse"/>
</wsdl:message>
<wsdl:message name="GetPresetsRequest">
  <wsdl:part name="parameters" element="tptz:GetPresets"/>
</wsdl:message>
<wsdl:message name="GetPresetsResponse">
  <wsdl:part name="parameters" element="tptz:GetPresetsResponse"/>
</wsdl:message>
<wsdl:message name="SetPresetRequest">
  <wsdl:part name="parameters" element="tptz:SetPreset"/>
</wsdl:message>
<wsdl:message name="SetPresetResponse">
  <wsdl:part name="parameters" element="tptz:SetPresetResponse"/>
</wsdl:message>
<wsdl:message name="RemovePresetRequest">
  <wsdl:part name="parameters" element="tptz:RemovePreset"/>
</wsdl:message>
<wsdl:message name="RemovePresetResponse">
  <wsdl:part name="parameters" element="tptz:RemovePresetResponse"/>
</wsdl:message>
<wsdl:message name="GotoPresetRequest">
  <wsdl:part name="parameters" element="tptz:GotoPreset"/>
</wsdl:message>
<wsdl:message name="GotoPresetResponse">
  <wsdl:part name="parameters" element="tptz:GotoPresetResponse"/>
</wsdl:message>
<wsdl:message name="GetStatusRequest">
  <wsdl:part name="parameters" element="tptz:GetStatus"/>
</wsdl:message>
<wsdl:message name="GetStatusResponse">
  <wsdl:part name="parameters" element="tptz:GetStatusResponse"/>
</wsdl:message>
<wsdl:message name="SendAuxiliaryCommandRequest">

```

```

    <wsdl:part name="parameters" element="tptz:SendAuxiliaryCommand"/>
</wsdl:message>
<wsdl:message name="SendAuxiliaryCommandResponse">
    <wsdl:part name="parameters" element="tptz:SendAuxiliaryCommandResponse"/>
</wsdl:message>
<wsdl:message name="GotoHomePositionRequest">
    <wsdl:part name="parameters" element="tptz:GotoHomePosition"/>
</wsdl:message>
<wsdl:message name="GotoHomePositionResponse">
    <wsdl:part name="parameters" element="tptz:GotoHomePositionResponse"/>
</wsdl:message>
<wsdl:message name="SetHomePositionRequest">
    <wsdl:part name="parameters" element="tptz:SetHomePosition"/>
</wsdl:message>
<wsdl:message name="SetHomePositionResponse">
    <wsdl:part name="parameters" element="tptz:SetHomePositionResponse"/>
</wsdl:message>
<wsdl:message name="ContinuousMoveRequest">
    <wsdl:part name="parameters" element="tptz:ContinuousMove"/>
</wsdl:message>
<wsdl:message name="ContinuousMoveResponse">
    <wsdl:part name="parameters" element="tptz:ContinuousMoveResponse"/>
</wsdl:message>
<wsdl:message name="RelativeMoveRequest">
    <wsdl:part name="parameters" element="tptz:RelativeMove"/>
</wsdl:message>
<wsdl:message name="RelativeMoveResponse">
    <wsdl:part name="parameters" element="tptz:RelativeMoveResponse"/>
</wsdl:message>
<wsdl:message name="AbsoluteMoveRequest">
    <wsdl:part name="parameters" element="tptz:AbsoluteMove"/>
</wsdl:message>
<wsdl:message name="AbsoluteMoveResponse">
    <wsdl:part name="parameters" element="tptz:AbsoluteMoveResponse"/>
</wsdl:message>
<wsdl:message name="StopRequest">
    <wsdl:part name="parameters" element="tptz:Stop"/>
</wsdl:message>
<wsdl:message name="StopResponse">
    <wsdl:part name="parameters" element="tptz:StopResponse"/>
</wsdl:message>
<wsdl:portType name="PTZ">
    <wsdl:operation name="GetNodes">
        <wsdl:input message="tptz:GetNodesRequest"/>
        <wsdl:output message="tptz:GetNodesResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetNode">
        <wsdl:input message="tptz:GetNodeRequest"/>
        <wsdl:output message="tptz:GetNodeResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetConfiguration">
        <wsdl:input message="tptz:GetConfigurationRequest"/>
        <wsdl:output message="tptz:GetConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetConfigurations">
        <wsdl:input message="tptz:GetConfigurationsRequest"/>
        <wsdl:output message="tptz:GetConfigurationsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetConfiguration">
        <wsdl:input message="tptz:SetConfigurationRequest"/>
        <wsdl:output message="tptz:SetConfigurationResponse"/>
    </wsdl:operation>

```

```

<wsdl:operation name="GetConfigurationOptions">
  <wsdl:input message="tptz:GetConfigurationOptionsRequest"/>
  <wsdl:output message="tptz:GetConfigurationOptionsResponse"/>
</wsdl:operation>
<wsdl:operation name="SendAuxiliaryCommand">
  <wsdl:input message="tptz:SendAuxiliaryCommandRequest"/>
  <wsdl:output message="tptz:SendAuxiliaryCommandResponse"/>
</wsdl:operation>
<wsdl:operation name="GetPresets">
  <wsdl:input message="tptz:GetPresetsRequest"/>
  <wsdl:output message="tptz:GetPresetsResponse"/>
</wsdl:operation>
<wsdl:operation name="SetPreset">
  <wsdl:input message="tptz:SetPresetRequest"/>
  <wsdl:output message="tptz:SetPresetResponse"/>
</wsdl:operation>
<wsdl:operation name="RemovePreset">
  <wsdl:input message="tptz:RemovePresetRequest"/>
  <wsdl:output message="tptz:RemovePresetResponse"/>
</wsdl:operation>
<wsdl:operation name="GotoPreset">
  <wsdl:input message="tptz:GotoPresetRequest"/>
  <wsdl:output message="tptz:GotoPresetResponse"/>
</wsdl:operation>
<wsdl:operation name="GotoHomePosition">
  <wsdl:input message="tptz:GotoHomePositionRequest"/>
  <wsdl:output message="tptz:GotoHomePositionResponse"/>
</wsdl:operation>
<wsdl:operation name="SetHomePosition">
  <wsdl:input message="tptz:SetHomePositionRequest"/>
  <wsdl:output message="tptz:SetHomePositionResponse"/>
</wsdl:operation>
<wsdl:operation name="ContinuousMove">
  <wsdl:input message="tptz:ContinuousMoveRequest"/>
  <wsdl:output message="tptz:ContinuousMoveResponse"/>
</wsdl:operation>
<wsdl:operation name="RelativeMove">
  <wsdl:input message="tptz:RelativeMoveRequest"/>
  <wsdl:output message="tptz:RelativeMoveResponse"/>
</wsdl:operation>
<wsdl:operation name="GetStatus">
  <wsdl:input message="tptz:GetStatusRequest"/>
  <wsdl:output message="tptz:GetStatusResponse"/>
</wsdl:operation>
<wsdl:operation name="AbsoluteMove">
  <wsdl:input message="tptz:AbsoluteMoveRequest"/>
  <wsdl:output message="tptz:AbsoluteMoveResponse"/>
</wsdl:operation>
<wsdl:operation name="Stop">
  <wsdl:input message="tptz:StopRequest"/>
  <wsdl:output message="tptz:StopResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="PTZBinding" type="tptz:PTZ">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetConfigurations">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GetConfigurations"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>

```

```
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetPresets">
  <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GetPresets"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetPreset">
  <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/SetPreset"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="RemovePreset">
  <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/RemovePreset"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GotoPreset">
  <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GotoPreset"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetStatus">
  <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GetStatus"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetConfiguration">
  <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GetConfiguration"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetNodes">
  <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GetNodes"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
```



```

        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetNode">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GetNode"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetConfiguration">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/SetConfiguration"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetConfigurationOptions">
    <soap:operation
soapAction="http://www.onvif.org/ver20/ptz/wsdl/GetConfigurationOptions"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GotoHomePosition">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/GotoHomePosition"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetHomePosition">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/SetHomePosition"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="ContinuousMove">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/ContinuousMove"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="RelativeMove">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/RelativeMove"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>

```

```

    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SendAuxiliaryCommand">
    <soap:operation
soapAction="http://www.onvif.org/ver20/ptz/wsdl/SendAuxiliaryCommand"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="AbsoluteMove">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/AbsoluteMove"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="Stop">
    <soap:operation soapAction="http://www.onvif.org/ver20/ptz/wsdl/Stop"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

C.10 Receiver service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:trv="http://www.onvif.org/ver10/receiver/wsdl"
targetNamespace="http://www.onvif.org/ver10/receiver/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
targetNamespace="http://www.onvif.org/ver10/receiver/wsdl"
elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!-- Message
Request/Responses elements --><!--=====-->
      <xs:element name="GetReceivers">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetReceiversResponse">
        <xs:complexType>
          <xs:sequence>

```



```

        <xs:element name="Receivers" type="tt:Receiver" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="GetReceiver">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ReceiverToken" type="tt:ReceiverToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetReceiverResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Receiver" type="tt:Receiver"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="CreateReceiver">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Configuration" type="tt:ReceiverConfiguration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="CreateReceiverResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Receiver" type="tt:Receiver"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="DeleteReceiver">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ReceiverToken" type="tt:ReceiverToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="DeleteReceiverResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="ConfigureReceiver">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ReceiverToken" type="tt:ReceiverToken"/>
            <xs:element name="Configuration" type="tt:ReceiverConfiguration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="ConfigureReceiverResponse">
    <xs:complexType>
        <xs:sequence/>
    </xs:complexType>
</xs:element>
<xs:element name="SetReceiverMode">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ReceiverToken" type="tt:ReceiverToken"/>

```

```

        <xs:element name="Mode" type="tt:ReceiverMode"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetReceiverModeResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetReceiverState">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ReceiverToken" type="tt:ReceiverToken"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetReceiverStateResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ReceiverState" type="tt:ReceiverStateInformation"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
</wsdl:types>
<wsdl:message name="GetReceiversRequest">
  <wsdl:part name="parameters" element="trv:GetReceivers"/>
</wsdl:message>
<wsdl:message name="GetReceiversResponse">
  <wsdl:part name="parameters" element="trv:GetReceiversResponse"/>
</wsdl:message>
<wsdl:message name="GetReceiverRequest">
  <wsdl:part name="parameters" element="trv:GetReceiver"/>
</wsdl:message>
<wsdl:message name="GetReceiverResponse">
  <wsdl:part name="parameters" element="trv:GetReceiverResponse"/>
</wsdl:message>
<wsdl:message name="CreateReceiverRequest">
  <wsdl:part name="parameters" element="trv:CreateReceiver"/>
</wsdl:message>
<wsdl:message name="CreateReceiverResponse">
  <wsdl:part name="parameters" element="trv:CreateReceiverResponse"/>
</wsdl:message>
<wsdl:message name="DeleteReceiverRequest">
  <wsdl:part name="parameters" element="trv:DeleteReceiver"/>
</wsdl:message>
<wsdl:message name="DeleteReceiverResponse">
  <wsdl:part name="parameters" element="trv:DeleteReceiverResponse"/>
</wsdl:message>
<wsdl:message name="ConfigureReceiverRequest">
  <wsdl:part name="parameters" element="trv:ConfigureReceiver"/>
</wsdl:message>
<wsdl:message name="ConfigureReceiverResponse">
  <wsdl:part name="parameters" element="trv:ConfigureReceiverResponse"/>
</wsdl:message>
<wsdl:message name="SetReceiverModeRequest">
  <wsdl:part name="parameters" element="trv:SetReceiverMode"/>
</wsdl:message>
<wsdl:message name="SetReceiverModeResponse">
  <wsdl:part name="parameters" element="trv:SetReceiverModeResponse"/>
</wsdl:message>
<wsdl:message name="GetReceiverStateRequest">

```

```

    <wsdl:part name="parameters" element="trv:GetReceiverState"/>
</wsdl:message>
<wsdl:message name="GetReceiverStateResponse">
    <wsdl:part name="parameters" element="trv:GetReceiverStateResponse"/>
</wsdl:message>
<wsdl:portType name="ReceiverPort">
    <wsdl:operation name="GetReceivers">
        <wsdl:input message="trv:GetReceiversRequest"/>
        <wsdl:output message="trv:GetReceiversResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetReceiver">
        <wsdl:input message="trv:GetReceiverRequest"/>
        <wsdl:output message="trv:GetReceiverResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreateReceiver">
        <wsdl:input message="trv:CreateReceiverRequest"/>
        <wsdl:output message="trv:CreateReceiverResponse"/>
    </wsdl:operation>
    <wsdl:operation name="DeleteReceiver">
        <wsdl:input message="trv:DeleteReceiverRequest"/>
        <wsdl:output message="trv:DeleteReceiverResponse"/>
    </wsdl:operation>
    <wsdl:operation name="ConfigureReceiver">
        <wsdl:input message="trv:ConfigureReceiverRequest"/>
        <wsdl:output message="trv:ConfigureReceiverResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetReceiverMode">
        <wsdl:input message="trv:SetReceiverModeRequest"/>
        <wsdl:output message="trv:SetReceiverModeResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetReceiverState">
        <wsdl:input message="trv:GetReceiverStateRequest"/>
        <wsdl:output message="trv:GetReceiverStateResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ReceiverBinding" type="trv:ReceiverPort">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="GetReceivers">
        <soap:operation soapAction="http://www.onvif.org/ver10/receiver/wsd/GetReceivers"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetReceiver">
        <soap:operation soapAction="http://www.onvif.org/ver10/receiver/wsd/GetReceiver"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateReceiver">
        <soap:operation
soapAction="http://www.onvif.org/ver10/receiver/wsd/CreateReceiver"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>

```

```

        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="DeleteReceiver">
    <soap:operation
soapAction="http://www.onvif.org/ver10/receiver/wsdl/DeleteReceiver"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="ConfigureReceiver">
    <soap:operation
soapAction="http://www.onvif.org/ver10/receiver/wsdl/ConfigureReceiver"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="SetReceiverMode">
    <soap:operation
soapAction="http://www.onvif.org/ver10/receiver/wsdl/SetReceiverMode"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
<wsdl:operation name="GetReceiverState">
    <soap:operation
soapAction="http://www.onvif.org/ver10/receiver/wsdl/GetReceiverState"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

C.11 Recording control service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:trc="http://www.onvif.org/ver10/recording/wsdl"
targetNamespace="http://www.onvif.org/ver10/recording/wsdl">
    <wsdl:types>
        <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
targetNamespace="http://www.onvif.org/ver10/recording/wsdl"
elementFormDefault="qualified">
            <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!-- Message

```

Request/Responses elements --><!--=====-->

```

<xs:element name="CreateRecording">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingConfiguration" type="tt:RecordingConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="CreateRecordingResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingToken" type="tt:RecordingReference"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="DeleteRecording">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingToken" type="tt:RecordingReference"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="DeleteRecordingResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordings">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingItem" type="tt:GetRecordingsResponseItem"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRecordingConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingToken" type="tt:RecordingReference"/>
      <xs:element name="RecordingConfiguration" type="tt:RecordingConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRecordingConfigurationResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingToken" type="tt:RecordingReference"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingConfigurationResponse">
  <xs:complexType>

```

```

        <xs:sequence>
          <xs:element name="RecordingConfiguration" type="tt:RecordingConfiguration"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="CreateTrack">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="RecordingToken" type="tt:RecordingReference"/>
          <xs:element name="TrackConfiguration" type="tt:TrackConfiguration"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="CreateTrackResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="TrackToken" type="tt:TrackReference"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="DeleteTrack">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="RecordingToken" type="tt:RecordingReference"/>
          <xs:element name="TrackToken" type="tt:TrackReference"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="DeleteTrackResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetTrackConfiguration">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="RecordingToken" type="tt:RecordingReference"/>
          <xs:element name="TrackToken" type="tt:TrackReference"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="GetTrackConfigurationResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="TrackConfiguration" type="tt:TrackConfiguration"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetTrackConfiguration">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="RecordingToken" type="tt:RecordingReference"/>
          <xs:element name="TrackToken" type="tt:TrackReference"/>
          <xs:element name="TrackConfiguration" type="tt:TrackConfiguration"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SetTrackConfigurationResponse">
      <xs:complexType>
        <xs:sequence/>
      </xs:complexType>
    </xs:element>

```



```

<xs:element name="CreateRecordingJob">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobConfiguration" type="tt:RecordingJobConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="CreateRecordingJobResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobToken" type="tt:RecordingJobReference"/>
      <xs:element name="JobConfiguration" type="tt:RecordingJobConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="DeleteRecordingJob">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobToken" type="tt:RecordingJobReference"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="DeleteRecordingJobResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingJobs">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingJobsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobItem" type="tt:GetRecordingJobsResponseItem"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRecordingJobConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobToken" type="tt:RecordingJobReference"/>
      <xs:element name="JobConfiguration" type="tt:RecordingJobConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRecordingJobConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobConfiguration" type="tt:RecordingJobConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingJobConfiguration">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobToken" type="tt:RecordingJobReference"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```



```

<xs:element name="GetRecordingJobConfigurationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobConfiguration" type="tt:RecordingJobConfiguration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRecordingJobMode">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobToken" type="tt:RecordingJobReference"/>
      <xs:element name="Mode" type="tt:RecordingJobMode"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="SetRecordingJobModeResponse">
  <xs:complexType>
    <xs:sequence/>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingJobState">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="JobToken" type="tt:RecordingJobReference"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingJobStateResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="State" type="tt:RecordingJobStateInformation"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
</wsdl:types>
<wsdl:message name="CreateRecordingRequest">
  <wsdl:part name="parameters" element="trc:CreateRecording"/>
</wsdl:message>
<wsdl:message name="CreateRecordingResponse">
  <wsdl:part name="parameters" element="trc:CreateRecordingResponse"/>
</wsdl:message>
<wsdl:message name="DeleteRecordingRequest">
  <wsdl:part name="parameters" element="trc:DeleteRecording"/>
</wsdl:message>
<wsdl:message name="DeleteRecordingResponse">
  <wsdl:part name="parameters" element="trc:DeleteRecordingResponse"/>
</wsdl:message>
<wsdl:message name="GetRecordingsRequest">
  <wsdl:part name="parameters" element="trc:GetRecordings"/>
</wsdl:message>
<wsdl:message name="GetRecordingsResponse">
  <wsdl:part name="parameters" element="trc:GetRecordingsResponse"/>
</wsdl:message>
<wsdl:message name="SetRecordingConfigurationRequest">
  <wsdl:part name="parameters" element="trc:SetRecordingConfiguration"/>
</wsdl:message>
<wsdl:message name="SetRecordingConfigurationResponse">
  <wsdl:part name="parameters" element="trc:SetRecordingConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetRecordingConfigurationRequest">
  <wsdl:part name="parameters" element="trc:GetRecordingConfiguration"/>

```

```

</wsdl:message>
<wsdl:message name="GetRecordingConfigurationResponse">
  <wsdl:part name="parameters" element="trc:GetRecordingConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="CreateTrackRequest">
  <wsdl:part name="parameters" element="trc:CreateTrack"/>
</wsdl:message>
<wsdl:message name="CreateTrackResponse">
  <wsdl:part name="parameters" element="trc:CreateTrackResponse"/>
</wsdl:message>
<wsdl:message name="DeleteTrackRequest">
  <wsdl:part name="parameters" element="trc:DeleteTrack"/>
</wsdl:message>
<wsdl:message name="DeleteTrackResponse">
  <wsdl:part name="parameters" element="trc:DeleteTrackResponse"/>
</wsdl:message>
<wsdl:message name="GetTrackConfigurationRequest">
  <wsdl:part name="parameters" element="trc:GetTrackConfiguration"/>
</wsdl:message>
<wsdl:message name="GetTrackConfigurationResponse">
  <wsdl:part name="parameters" element="trc:GetTrackConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetTrackConfigurationRequest">
  <wsdl:part name="parameters" element="trc:SetTrackConfiguration"/>
</wsdl:message>
<wsdl:message name="SetTrackConfigurationResponse">
  <wsdl:part name="parameters" element="trc:SetTrackConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="CreateRecordingJobRequest">
  <wsdl:part name="parameters" element="trc:CreateRecordingJob"/>
</wsdl:message>
<wsdl:message name="CreateRecordingJobResponse">
  <wsdl:part name="parameters" element="trc:CreateRecordingJobResponse"/>
</wsdl:message>
<wsdl:message name="DeleteRecordingJobRequest">
  <wsdl:part name="parameters" element="trc:DeleteRecordingJob"/>
</wsdl:message>
<wsdl:message name="DeleteRecordingJobResponse">
  <wsdl:part name="parameters" element="trc:DeleteRecordingJobResponse"/>
</wsdl:message>
<wsdl:message name="GetRecordingJobsRequest">
  <wsdl:part name="parameters" element="trc:GetRecordingJobs"/>
</wsdl:message>
<wsdl:message name="GetRecordingJobsResponse">
  <wsdl:part name="parameters" element="trc:GetRecordingJobsResponse"/>
</wsdl:message>
<wsdl:message name="SetRecordingJobConfigurationRequest">
  <wsdl:part name="parameters" element="trc:SetRecordingJobConfiguration"/>
</wsdl:message>
<wsdl:message name="SetRecordingJobConfigurationResponse">
  <wsdl:part name="parameters" element="trc:SetRecordingJobConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetRecordingJobConfigurationRequest">
  <wsdl:part name="parameters" element="trc:GetRecordingJobConfiguration"/>
</wsdl:message>
<wsdl:message name="GetRecordingJobConfigurationResponse">
  <wsdl:part name="parameters" element="trc:GetRecordingJobConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="SetRecordingJobModeRequest">
  <wsdl:part name="parameters" element="trc:SetRecordingJobMode"/>
</wsdl:message>
<wsdl:message name="SetRecordingJobModeResponse">

```

```

    <wsdl:part name="parameters" element="trc:SetRecordingJobModeResponse"/>
  </wsdl:message>
  <wsdl:message name="GetRecordingJobStateRequest">
    <wsdl:part name="parameters" element="trc:GetRecordingJobState"/>
  </wsdl:message>
  <wsdl:message name="GetRecordingJobStateResponse">
    <wsdl:part name="parameters" element="trc:GetRecordingJobStateResponse"/>
  </wsdl:message>
  <wsdl:portType name="RecordingPort">
    <wsdl:operation name="CreateRecording">
      <wsdl:input message="trc:CreateRecordingRequest"/>
      <wsdl:output message="trc:CreateRecordingResponse"/>
    </wsdl:operation>
    <wsdl:operation name="DeleteRecording">
      <wsdl:input message="trc:DeleteRecordingRequest"/>
      <wsdl:output message="trc:DeleteRecordingResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetRecordings">
      <wsdl:input message="trc:GetRecordingsRequest"/>
      <wsdl:output message="trc:GetRecordingsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetRecordingConfiguration">
      <wsdl:input message="trc:SetRecordingConfigurationRequest"/>
      <wsdl:output message="trc:SetRecordingConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetRecordingConfiguration">
      <wsdl:input message="trc:GetRecordingConfigurationRequest"/>
      <wsdl:output message="trc:GetRecordingConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreateTrack">
      <wsdl:input message="trc:CreateTrackRequest"/>
      <wsdl:output message="trc:CreateTrackResponse"/>
    </wsdl:operation>
    <wsdl:operation name="DeleteTrack">
      <wsdl:input message="trc:DeleteTrackRequest"/>
      <wsdl:output message="trc:DeleteTrackResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetTrackConfiguration">
      <wsdl:input message="trc:GetTrackConfigurationRequest"/>
      <wsdl:output message="trc:GetTrackConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetTrackConfiguration">
      <wsdl:input message="trc:SetTrackConfigurationRequest"/>
      <wsdl:output message="trc:SetTrackConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CreateRecordingJob">
      <wsdl:input message="trc:CreateRecordingJobRequest"/>
      <wsdl:output message="trc:CreateRecordingJobResponse"/>
    </wsdl:operation>
    <wsdl:operation name="DeleteRecordingJob">
      <wsdl:input message="trc:DeleteRecordingJobRequest"/>
      <wsdl:output message="trc:DeleteRecordingJobResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetRecordingJobs">
      <wsdl:input message="trc:GetRecordingJobsRequest"/>
      <wsdl:output message="trc:GetRecordingJobsResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetRecordingJobConfiguration">
      <wsdl:input message="trc:SetRecordingJobConfigurationRequest"/>
      <wsdl:output message="trc:SetRecordingJobConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetRecordingJobConfiguration">

```

```

        <wsdl:input message="trc:GetRecordingJobConfigurationRequest"/>
        <wsdl:output message="trc:GetRecordingJobConfigurationResponse"/>
    </wsdl:operation>
    <wsdl:operation name="SetRecordingJobMode">
        <wsdl:input message="trc:SetRecordingJobModeRequest"/>
        <wsdl:output message="trc:SetRecordingJobModeResponse"/>
    </wsdl:operation>
    <wsdl:operation name="GetRecordingJobState">
        <wsdl:input message="trc:GetRecordingJobStateRequest"/>
        <wsdl:output message="trc:GetRecordingJobStateResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="RecordingBinding" type="trc:RecordingPort">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="CreateRecording">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/CreateRecording"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DeleteRecording">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/DeleteRecording"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetRecordings">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/GetRecordings"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetRecordingConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/SetRecordingConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetRecordingConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/GetRecordingConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateTrack">
        <soap:operation soapAction="http://www.onvif.org/ver10/recording/wsdl/CreateTrack"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DeleteTrack">
        <soap:operation soapAction="http://www.onvif.org/ver10/recording/wsdl/DeleteTrack"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetTrackConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/GetTrackConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetTrackConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/SetTrackConfiguration"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateRecordingJob">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/CreateRecordingJob"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DeleteRecordingJob">
        <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsdl/DeleteRecordingJob"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetRecordingJobs">
        <soap:operation

```



```

soapAction="http://www.onvif.org/ver10/recording/wsd/GetRecordingJobs"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetRecordingJobConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsd/SetRecordingJobConfiguration"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetRecordingJobConfiguration">
  <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsd/GetRecordingJobConfiguration"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SetRecordingJobMode">
  <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsd/SetRecordingJobMode"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetRecordingJobState">
  <soap:operation
soapAction="http://www.onvif.org/ver10/recording/wsd/GetRecordingJobState"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
</wsdl:binding><!--=====-->
</wsdl:definitions>

```

C.12 Remote discovery proxy services WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:dn="http://www.onvif.org/ver10/network/wsd/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.onvif.org/ver10/network/wsd">

```

```

<wsdl:types>
  <xs:schema xmlns:d="http://schemas.xmlsoap.org/ws/2005/04/discovery"
targetNamespace="http://www.onvif.org/ver10/network/wsdl" elementFormDefault="qualified">
    <xs:import namespace="http://schemas.xmlsoap.org/ws/2005/04/discovery"
schemaLocation="http://schemas.xmlsoap.org/ws/2005/04/discovery/ws-discovery.xsd"/><!--
Message Request/Responses elements --><!--=====-->
    <xs:element name="Hello" type="d:HelloType"/>
    <xs:element name="HelloResponse" type="d:ResolveType"/>
    <xs:element name="Probe" type="d:ProbeType"/>
    <xs:element name="ProbeResponse" type="d:ProbeMatchesType"/>
    <xs:element name="Bye" type="d:ByeType"/>
    <xs:element name="ByeResponse" type="d:ResolveType"/><!--
=====-->
  </xs:schema>
</wsdl:types>
<wsdl:message name="HelloRequest">
  <wsdl:part name="parameters" element="dn:Hello"/>
</wsdl:message>
<wsdl:message name="HelloResponse">
  <wsdl:part name="parameters" element="dn:HelloResponse"/>
</wsdl:message>
<wsdl:message name="ProbeRequest">
  <wsdl:part name="parameters" element="dn:Probe"/>
</wsdl:message>
<wsdl:message name="ProbeResponse">
  <wsdl:part name="parameters" element="dn:ProbeResponse"/>
</wsdl:message>
<wsdl:message name="ByeRequest">
  <wsdl:part name="parameters" element="dn:Bye"/>
</wsdl:message>
<wsdl:message name="ByeResponse">
  <wsdl:part name="parameters" element="dn:ByeResponse"/>
</wsdl:message>
<wsdl:portType name="RemoteDiscoveryPort">
  <wsdl:operation name="Hello">
    <wsdl:input message="dn:HelloRequest"
dn:Action="http://schemas.xmlsoap.org/ws/2005/04/discovery/Hello"/>
    <wsdl:output message="dn:HelloResponse"
dn:Action="http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe"/>
  </wsdl:operation>
  <wsdl:operation name="Bye">
    <wsdl:input message="dn:ByeRequest"
dn:Action="http://schemas.xmlsoap.org/ws/2005/04/discovery/Bye"/>
    <wsdl:output message="dn:ByeResponse"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:portType name="DiscoveryLookupPort">
  <wsdl:operation name="Probe">
    <wsdl:input message="dn:ProbeRequest"
dn:Action="http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe"/>
    <wsdl:output message="dn:ProbeResponse"
dn:Action="http://schemas.xmlsoap.org/ws/2005/04/discovery/ProbeMatches"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="RemoteDiscoveryBinding" type="dn:RemoteDiscoveryPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="Hello">
    <soap:operation soapAction="http://www.onvif.org/ver10/network/wsdl/Hello"/>
    <wsdl:input>
      <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>

```



```

        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="Bye">
    <soap:operation soapAction="http://www.onvif.org/ver10/network/wsdl/Bye"/>
    <wsdl:input>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="parameters" use="literal"/>
    </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="DiscoveryLookupBinding" type="dn:DiscoveryLookupPort">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="Probe">
        <soap:operation soapAction="http://www.onvif.org/ver10/network/wsdl/Probe"/>
        <wsdl:input>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

C.13 Replay service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:trp="http://www.onvif.org/ver10/replay/wsdl"
    targetNamespace="http://www.onvif.org/ver10/replay/wsdl">
    <wsdl:types>
        <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
            targetNamespace="http://www.onvif.org/ver10/replay/wsdl" elementFormDefault="qualified">
            <xs:import namespace="http://www.onvif.org/ver10/schema"
                schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!-- Message
            Request/Responses elements --><!--=====
            <xs:element name="GetReplayUri">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="StreamSetup" type="tt:StreamSetup"/>
                        <xs:element name="RecordingToken" type="tt:ReferenceToken"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="GetReplayUriResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="Uri" type="xs:anyURI"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="SetReplayConfiguration">
                <xs:complexType>
                    <xs:sequence>

```

```

        <xs:element name="Configuration" type="tt:ReplayConfiguration"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SetReplayConfigurationResponse">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetReplayConfiguration">
    <xs:complexType>
      <xs:sequence/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GetReplayConfigurationResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Configuration" type="tt:ReplayConfiguration"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
</wsdl:types>
<wsdl:message name="GetReplayUriRequest">
  <wsdl:part name="parameters" element="trp:GetReplayUri"/>
</wsdl:message>
<wsdl:message name="GetReplayUriResponse">
  <wsdl:part name="parameters" element="trp:GetReplayUriResponse"/>
</wsdl:message>
<wsdl:message name="SetReplayConfigurationRequest">
  <wsdl:part name="parameters" element="trp:SetReplayConfiguration"/>
</wsdl:message>
<wsdl:message name="SetReplayConfigurationResponse">
  <wsdl:part name="parameters" element="trp:SetReplayConfigurationResponse"/>
</wsdl:message>
<wsdl:message name="GetReplayConfigurationRequest">
  <wsdl:part name="parameters" element="trp:GetReplayConfiguration"/>
</wsdl:message>
<wsdl:message name="GetReplayConfigurationResponse">
  <wsdl:part name="parameters" element="trp:GetReplayConfigurationResponse"/>
</wsdl:message>
<wsdl:portType name="ReplayPort">
  <wsdl:operation name="GetReplayUri">
    <wsdl:input message="trp:GetReplayUriRequest"/>
    <wsdl:output message="trp:GetReplayUriResponse"/>
  </wsdl:operation>
  <wsdl:operation name="GetReplayConfiguration">
    <wsdl:input message="trp:GetReplayConfigurationRequest"/>
    <wsdl:output message="trp:GetReplayConfigurationResponse"/>
  </wsdl:operation>
  <wsdl:operation name="SetReplayConfiguration">
    <wsdl:input message="trp:SetReplayConfigurationRequest"/>
    <wsdl:output message="trp:SetReplayConfigurationResponse"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ReplayBinding" type="trp:ReplayPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetReplayUri">
    <soap:operation soapAction="http://www.onvif.org/ver10/replay/wsdl/GetReplayUri"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
  </wsdl:operation>

```

```

        <wsdl:output>
          <soap:body use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="GetReplayConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/replay/wsdl/GetReplayConfiguration"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="SetReplayConfiguration">
        <soap:operation
soapAction="http://www.onvif.org/ver10/replay/wsdl/SetReplayConfiguration"/>
        <wsdl:input>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:input>
        <wsdl:output>
          <soap:body parts="parameters" use="literal"/>
        </wsdl:output>
      </wsdl:operation>
    </wsdl:binding>
  </wsdl:definitions>

```

C.14 Search service WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tse="http://www.onvif.org/ver10/search/wsdl"
targetNamespace="http://www.onvif.org/ver10/search/wsdl">
  <wsdl:types>
    <xs:schema xmlns:tt="http://www.onvif.org/ver10/schema"
targetNamespace="http://www.onvif.org/ver10/search/wsdl" elementFormDefault="qualified">
      <xs:import namespace="http://www.onvif.org/ver10/schema"
schemaLocation="http://www.onvif.org/onvif/ver10/schema/onvif.xsd"/><!-- Message
Request/Responses elements --><!--=====-->
      <xs:element name="GetRecordingSummary">
        <xs:complexType>
          <xs:sequence/>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetRecordingSummaryResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Summary" type="tt:RecordingSummary"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element><!--=====-->
      <xs:element name="GetRecordingInformation">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="RecordingToken" type="tt:RecordingReference"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

```

```

<xs:element name="GetRecordingInformationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingInformation" type="tt:RecordingInformation"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetMediaAttributes">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RecordingTokens" type="tt:RecordingReference"
minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Time" type="xs:dateTime"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetMediaAttributesResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="MediaAttributes" type="tt:MediaAttributes" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="FindRecordings">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Scope" type="tt:SearchScope"/>
      <xs:element name="MaxMatches" type="xs:int" minOccurs="0"/>
      <xs:element name="KeepAliveTime" type="xs:duration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="FindRecordingsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SearchToken" type="tt:JobToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetRecordingSearchResults">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SearchToken" type="tt:JobToken"/>
      <xs:element name="MinResults" type="xs:int" minOccurs="0"/>
      <xs:element name="MaxResults" type="xs:int" minOccurs="0"/>
      <xs:element name="WaitTime" type="xs:duration" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetRecordingSearchResultsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ResultList" type="tt:FindRecordingResultList"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="FindEvents">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="StartPoint" type="xs:dateTime"/>
      <xs:element name="EndPoint" type="xs:dateTime" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

        <xs:element name="Scope" type="tt:SearchScope"/>
        <xs:element name="SearchFilter" type="tt:EventFilter"/>
        <xs:element name="IncludeStartState" type="xs:boolean"/>
        <xs:element name="MaxMatches" type="xs:int" minOccurs="0"/>
        <xs:element name="KeepAliveTime" type="xs:duration"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="FindEventsResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="SearchToken" type="tt:JobToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetEventSearchResults">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="SearchToken" type="tt:JobToken"/>
            <xs:element name="MinResults" type="xs:int" minOccurs="0"/>
            <xs:element name="MaxResults" type="xs:int" minOccurs="0"/>
            <xs:element name="WaitTime" type="xs:duration" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetEventSearchResultsResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ResultList" type="tt:FindEventResultList"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="FindPTZPosition">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="StartPoint" type="xs:dateTime"/>
            <xs:element name="EndPoint" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="Scope" type="tt:SearchScope"/>
            <xs:element name="SearchFilter" type="tt:PTZPositionFilter"/>
            <xs:element name="MaxMatches" type="xs:int" minOccurs="0"/>
            <xs:element name="KeepAliveTime" type="xs:duration"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="FindPTZPositionResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="SearchToken" type="tt:JobToken"/>
        </xs:sequence>
    </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetPTZPositionSearchResults">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="SearchToken" type="tt:JobToken"/>
            <xs:element name="MinResults" type="xs:int" minOccurs="0"/>
            <xs:element name="MaxResults" type="xs:int" minOccurs="0"/>
            <xs:element name="WaitTime" type="xs:duration" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="GetPTZPositionSearchResultsResponse">

```



```

<xs:complexType>
  <xs:sequence>
    <xs:element name="ResultList" type="tt:FindPTZPositionResultList"/>
  </xs:sequence>
</xs:complexType>
</xs:element><!--=====-->
<xs:element name="FindMetadata">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="StartPoint" type="xs:dateTime"/>
      <xs:element name="EndPoint" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="Scope" type="tt:SearchScope"/>
      <xs:element name="MetadataFilter" type="tt:MetadataFilter"/>
      <xs:element name="MaxMatches" type="xs:int" minOccurs="0"/>
      <xs:element name="KeepAliveTime" type="xs:duration"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="FindMetadataResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SearchToken" type="tt:JobToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetMetadataSearchResults">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SearchToken" type="tt:JobToken"/>
      <xs:element name="MinResults" type="xs:int" minOccurs="0"/>
      <xs:element name="MaxResults" type="xs:int" minOccurs="0"/>
      <xs:element name="WaitTime" type="xs:duration" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetMetadataSearchResultsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ResultList" type="tt:FindMetadataResultList"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="GetSearchState">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SearchToken" type="tt:JobToken"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="GetSearchStateResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="State" type="tt:SearchState"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====-->
<xs:element name="EndSearch">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SearchToken" type="tt:JobToken"/>
    </xs:sequence>
  </xs:complexType>

```

```

</xs:element>
<xs:element name="EndSearchResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Endpoint" type="xs:dateTime"/>
    </xs:sequence>
  </xs:complexType>
</xs:element><!--=====--><!--
=====-->
</xs:schema>
</wsdl:types>
<wsdl:message name="FindEventsRequest">
  <wsdl:part name="parameters" element="tse:FindEvents"/>
</wsdl:message>
<wsdl:message name="FindEventsResponse">
  <wsdl:part name="parameters" element="tse:FindEventsResponse"/>
</wsdl:message>
<wsdl:message name="GetEventSearchResultsRequest">
  <wsdl:part name="parameters" element="tse:GetEventSearchResults"/>
</wsdl:message>
<wsdl:message name="GetEventSearchResultsResponse">
  <wsdl:part name="parameters" element="tse:GetEventSearchResultsResponse"/>
</wsdl:message>
<wsdl:message name="GetSearchStateRequest">
  <wsdl:part name="parameters" element="tse:GetSearchState"/>
</wsdl:message>
<wsdl:message name="GetSearchStateResponse">
  <wsdl:part name="parameters" element="tse:GetSearchStateResponse"/>
</wsdl:message>
<wsdl:message name="EndSearchRequest">
  <wsdl:part name="parameters" element="tse:EndSearch"/>
</wsdl:message>
<wsdl:message name="EndSearchResponse">
  <wsdl:part name="parameters" element="tse:EndSearchResponse"/>
</wsdl:message>
<wsdl:message name="FindPTZPositionRequest">
  <wsdl:part name="parameters" element="tse:FindPTZPosition"/>
</wsdl:message>
<wsdl:message name="FindPTZPositionResponse">
  <wsdl:part name="parameters" element="tse:FindPTZPositionResponse"/>
</wsdl:message>
<wsdl:message name="GetPTZPositionSearchResultsRequest">
  <wsdl:part name="parameters" element="tse:GetPTZPositionSearchResults"/>
</wsdl:message>
<wsdl:message name="GetPTZPositionSearchResultsResponse">
  <wsdl:part name="parameters" element="tse:GetPTZPositionSearchResultsResponse"/>
</wsdl:message>
<wsdl:message name="GetRecordingSummaryRequest">
  <wsdl:part name="parameters" element="tse:GetRecordingSummary"/>
</wsdl:message>
<wsdl:message name="GetRecordingSummaryResponse">
  <wsdl:part name="parameters" element="tse:GetRecordingSummaryResponse"/>
</wsdl:message>
<wsdl:message name="GetRecordingInformationRequest">
  <wsdl:part name="parameters" element="tse:GetRecordingInformation"/>
</wsdl:message>
<wsdl:message name="GetRecordingInformationResponse">
  <wsdl:part name="parameters" element="tse:GetRecordingInformationResponse"/>
</wsdl:message>
<wsdl:message name="GetMediaAttributesRequest">
  <wsdl:part name="parameters" element="tse:GetMediaAttributes"/>
</wsdl:message>

```



```

<wsdl:message name="GetMediaAttributesResponse">
  <wsdl:part name="parameters" element="tse:GetMediaAttributesResponse"/>
</wsdl:message>
<wsdl:message name="FindRecordingsRequest">
  <wsdl:part name="parameters" element="tse:FindRecordings"/>
</wsdl:message>
<wsdl:message name="FindRecordingsResponse">
  <wsdl:part name="parameters" element="tse:FindRecordingsResponse"/>
</wsdl:message>
<wsdl:message name="GetRecordingSearchResultsRequest">
  <wsdl:part name="parameters" element="tse:GetRecordingSearchResults"/>
</wsdl:message>
<wsdl:message name="GetRecordingSearchResultsResponse">
  <wsdl:part name="parameters" element="tse:GetRecordingSearchResultsResponse"/>
</wsdl:message>
<wsdl:message name="FindMetadataRequest">
  <wsdl:part name="parameters" element="tse:FindMetadata"/>
</wsdl:message>
<wsdl:message name="FindMetadataResponse">
  <wsdl:part name="parameters" element="tse:FindMetadataResponse"/>
</wsdl:message>
<wsdl:message name="GetMetadataSearchResultsRequest">
  <wsdl:part name="parameters" element="tse:GetMetadataSearchResults"/>
</wsdl:message>
<wsdl:message name="GetMetadataSearchResultsResponse">
  <wsdl:part name="parameters" element="tse:GetMetadataSearchResultsResponse"/>
</wsdl:message>
<wsdl:portType name="SearchPort"><!--=====--><!--
=====-->
  <wsdl:operation name="GetRecordingSummary">
    <wsdl:input message="tse:GetRecordingSummaryRequest"/>
    <wsdl:output message="tse:GetRecordingSummaryResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="GetRecordingInformation">
    <wsdl:input message="tse:GetRecordingInformationRequest"/>
    <wsdl:output message="tse:GetRecordingInformationResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="GetMediaAttributes">
    <wsdl:input message="tse:GetMediaAttributesRequest"/>
    <wsdl:output message="tse:GetMediaAttributesResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="FindRecordings">
    <wsdl:input message="tse:FindRecordingsRequest"/>
    <wsdl:output message="tse:FindRecordingsResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="GetRecordingSearchResults">
    <wsdl:input message="tse:GetRecordingSearchResultsRequest"/>
    <wsdl:output message="tse:GetRecordingSearchResultsResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="FindEvents">
    <wsdl:input message="tse:FindEventsRequest"/>
    <wsdl:output message="tse:FindEventsResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="GetEventSearchResults">
    <wsdl:input message="tse:GetEventSearchResultsRequest"/>
    <wsdl:output message="tse:GetEventSearchResultsResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="FindPTZPosition">
    <wsdl:input message="tse:FindPTZPositionRequest"/>
    <wsdl:output message="tse:FindPTZPositionResponse"/>
  </wsdl:operation><!--=====-->
  <wsdl:operation name="GetPTZPositionSearchResults">

```

```

        <wsdl:input message="tse:GetPTZPositionSearchResultsRequest"/>
        <wsdl:output message="tse:GetPTZPositionSearchResultsResponse"/>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetSearchState">
        <wsdl:input message="tse:GetSearchStateRequest"/>
        <wsdl:output message="tse:GetSearchStateResponse"/>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="EndSearch">
        <wsdl:input message="tse:EndSearchRequest"/>
        <wsdl:output message="tse:EndSearchResponse"/>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="FindMetadata">
        <wsdl:input message="tse:FindMetadataRequest"/>
        <wsdl:output message="tse:FindMetadataResponse"/>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetMetadataSearchResults">
        <wsdl:input message="tse:GetMetadataSearchResultsRequest"/>
        <wsdl:output message="tse:GetMetadataSearchResultsResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="SearchBinding" type="tse:SearchPort">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/><!--
=====--><!--=====-->
    <wsdl:operation name="GetRecordingSummary">
        <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/GetRecordingSummary"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetRecordingInformation">
        <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/GetRecordingInformation"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetMediaAttributes">
        <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/GetMediaAttributes"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="FindRecordings">
        <soap:operation soapAction="http://www.onvif.org/ver10/search/wsdl/FindRecordings"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation><!--=====-->

```

```

    <wsdl:operation name="GetRecordingSearchResults">
      <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/GetRecordingSearchResults"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="FindEvents">
      <soap:operation soapAction="http://www.onvif.org/ver10/search/wsdl/FindEvents"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetEventSearchResults">
      <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/GetEventSearchResults"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="FindPTZPosition">
      <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/FindPTZPosition"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetPTZPositionSearchResults">
      <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/GetPTZPositionSearchResults"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="GetSearchState">
      <soap:operation soapAction="http://www.onvif.org/ver10/search/wsdl/GetSearchState"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation><!--=====-->
    <wsdl:operation name="EndSearch">
      <soap:operation soapAction="http://www.onvif.org/ver10/search/wsdl/EndSearch"/>
      <wsdl:input>
        <soap:body use="literal"/>

```

```

        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation><!------->
    <wsdl:operation name="FindMetadata">
        <soap:operation soapAction="http://www.onvif.org/ver10/search/wsdl/FindMetadata"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation><!------->
    <wsdl:operation name="GetMetadataSearchResults">
        <soap:operation
soapAction="http://www.onvif.org/ver10/search/wsdl/GetMetadataSearchResults"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
/wsdl:definitions>

```

C.15 Common network video schema

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tt="http://www.onvif.org/ver10/schema"
xmlns:xmime="http://www.w3.org/2005/05/xmlmime" xmlns:wsnt="http://docs.oasis-
open.org/wsn/b-2" xmlns:xop="http://www.w3.org/2004/08/xop/include"
targetNamespace="http://www.onvif.org/ver10/schema" elementFormDefault="qualified">
    <xs:import namespace="http://www.w3.org/2005/05/xmlmime"
schemaLocation="http://www.w3.org/2005/05/xmlmime"/>
    <xs:import namespace="http://docs.oasis-open.org/wsn/b-2"
schemaLocation="http://docs.oasis-open.org/wsn/b-2.xsd"/>
    <xs:import namespace="http://www.w3.org/2004/08/xop/include"
schemaLocation="http://www.w3.org/2004/08/xop/include"/>
    <!------->
    <!-- Generic Types -->
    <!------->
    <xs:complexType name="DeviceEntity">
        <xs:attribute name="token" type="tt:ReferenceToken" use="required"/>
    </xs:complexType><!------->
    <xs:simpleType name="ReferenceToken">
        <xs:restriction base="xs:string">
            <xs:maxLength value="64"/>
        </xs:restriction>
    </xs:simpleType><!------->
    <xs:simpleType name="Name">
        <xs:restriction base="xs:string">
            <xs:maxLength value="64"/>
        </xs:restriction>
    </xs:simpleType><!------->
    <xs:complexType name="IntRectangle">
        <xs:attribute name="x" type="xs:int" use="required"/>
        <xs:attribute name="y" type="xs:int" use="required"/>
    </xs:complexType>

```

```

    <xs:attribute name="width" type="xs:int" use="required"/>
    <xs:attribute name="height" type="xs:int" use="required"/>
  </xs:complexType><!--=====-->
  <xs:complexType name="IntRectangleRange">
    <xs:sequence>
      <xs:element name="XRange" type="tt:IntRange"/>
      <xs:element name="YRange" type="tt:IntRange"/>
      <xs:element name="WidthRange" type="tt:IntRange"/>
      <xs:element name="HeightRange" type="tt:IntRange"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="IntRange">
    <xs:sequence>
      <xs:element name="Min" type="xs:int"/>
      <xs:element name="Max" type="xs:int"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="FloatRange">
    <xs:sequence>
      <xs:element name="Min" type="xs:float"/>
      <xs:element name="Max" type="xs:float"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="DurationRange">
    <xs:sequence>
      <xs:element name="Min" type="xs:duration"/>
      <xs:element name="Max" type="xs:duration"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="IntList">
    <xs:sequence>
      <xs:element name="Items" type="xs:int" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
<!--=====-->
<!-- End, Generic Types -->
<!--=====-->
<!--=====-->
<!--=====-->
<!-- Media Related Types -->
<!--=====-->
  <xs:complexType name="VideoSource">
    <xs:complexContent>
      <xs:extension base="tt:DeviceEntity">
        <xs:sequence>
          <xs:element name="Framerate" type="xs:float"/>
          <xs:element name="Resolution" type="tt:VideoResolution"/>
          <xs:element name="Imaging" type="tt:ImagingSettings" minOccurs="0"/>
          <xs:element name="Extension" type="tt:VideoSourceExtension" minOccurs="0"/>
        </xs:sequence>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType><!--=====-->
  <xs:complexType name="VideoSourceExtension">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="AudioSource">
    <xs:complexContent>

```



```

<xs:extension base="tt:DeviceEntity">
  <xs:sequence>
    <xs:element name="Channels" type="xs:int"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:extension>
</xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="Profile">
  <xs:sequence>
    <xs:element name="Name" type="tt:Name"/>
    <xs:element name="VideoSourceConfiguration" type="tt:VideoSourceConfiguration"
minOccurs="0"/>
    <xs:element name="AudioSourceConfiguration" type="tt:AudioSourceConfiguration"
minOccurs="0"/>
    <xs:element name="VideoEncoderConfiguration" type="tt:VideoEncoderConfiguration"
minOccurs="0"/>
    <xs:element name="AudioEncoderConfiguration" type="tt:AudioEncoderConfiguration"
minOccurs="0"/>
    <xs:element name="VideoAnalyticsConfiguration" type="tt:VideoAnalyticsConfiguration"
minOccurs="0"/>
    <xs:element name="PTZConfiguration" type="tt:PTZConfiguration" minOccurs="0"/>
    <xs:element name="MetadataConfiguration" type="tt:MetadataConfiguration"
minOccurs="0"/>
    <xs:element name="Extension" type="tt:ProfileExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="token" type="tt:ReferenceToken" use="required"/>
  <xs:attribute name="fixed" type="xs:boolean"/>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ProfileExtension">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="AudioOutputConfiguration" type="tt:AudioOutputConfiguration"
minOccurs="0"/>
    <xs:element name="AudioDecoderConfiguration" type="tt:AudioDecoderConfiguration"
minOccurs="0"/>
    <xs:element name="Extension" type="tt:ProfileExtension2" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ProfileExtension2">
  <xs:sequence>
    <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:element name="VideoSourceConfiguration" type="tt:VideoSourceConfiguration"/>
<xs:element name="AudioSourceConfiguration" type="tt:AudioSourceConfiguration"/>
<xs:element name="VideoEncoderConfiguration" type="tt:VideoEncoderConfiguration"/>
<xs:element name="AudioEncoderConfiguration" type="tt:AudioEncoderConfiguration"/>
<xs:element name="VideoAnalyticsConfiguration" type="tt:VideoAnalyticsConfiguration"/>
<xs:element name="PTZConfiguration" type="tt:PTZConfiguration"/>
<xs:element name="MetadataConfiguration" type="tt:MetadataConfiguration"/>
<xs:element name="AudioOutputConfiguration" type="tt:AudioOutputConfiguration"/>
<xs:element name="AudioDecoderConfiguration" type="tt:AudioDecoderConfiguration"/><!--
=====-->
<xs:complexType name="ConfigurationEntity">
  <xs:sequence>

```

```

    <xs:element name="Name" type="tt:Name"/>
    <xs:element name="UseCount" type="xs:int"/>
  </xs:sequence>
  <xs:attribute name="token" type="tt:ReferenceToken" use="required"/>
</xs:complexType><!--=====--><!--
VideoSourceConfiguration --><!--=====-->
<xs:complexType name="VideoSourceConfiguration">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="SourceToken" type="tt:ReferenceToken"/>
        <xs:element name="Bounds" type="tt:IntRectangle"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="VideoSourceConfigurationOptions">
  <xs:sequence>
    <xs:element name="BoundsRange" type="tt:IntRectangleRange"/>
    <xs:element name="VideoSourceTokensAvailable" type="tt:ReferenceToken"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:VideoSourceConfigurationOptionsExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="VideoSourceConfigurationOptionsExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====--><!--
VideoEncoderConfiguration --><!--=====-->
<xs:complexType name="VideoEncoderConfiguration">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="Encoding" type="tt:VideoEncoding"/>
        <xs:element name="Resolution" type="tt:VideoResolution"/>
        <xs:element name="Quality" type="xs:float"/>
        <xs:element name="RateControl" type="tt:VideoRateControl" minOccurs="0"/>
        <xs:element name="MPEG4" type="tt:Mpeg4Configuration" minOccurs="0"/>
        <xs:element name="H264" type="tt:H264Configuration" minOccurs="0"/>
        <xs:element name="Multicast" type="tt:MulticastConfiguration"/>
        <xs:element name="SessionTimeout" type="xs:duration"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:simpleType name="VideoEncoding">
  <xs:restriction base="xs:string">
    <xs:enumeration value="JPEG"/>
    <xs:enumeration value="MPEG4"/>
    <xs:enumeration value="H264"/>
  </xs:restriction>
</xs:simpleType><!--=====-->

```



```

<xs:simpleType name="Mpeg4Profile">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SP"/>
    <xs:enumeration value="ASP"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="H264Profile">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Baseline"/>
    <xs:enumeration value="Main"/>
    <xs:enumeration value="Extended"/>
    <xs:enumeration value="High"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="VideoResolution">
  <xs:sequence>
    <xs:element name="Width" type="xs:int"/>
    <xs:element name="Height" type="xs:int"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="VideoRateControl">
  <xs:sequence>
    <xs:element name="FrameRateLimit" type="xs:int"/>
    <xs:element name="EncodingInterval" type="xs:int"/>
    <xs:element name="BitrateLimit" type="xs:int"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Mpeg4Configuration">
  <xs:sequence>
    <xs:element name="GovLength" type="xs:int"/>
    <xs:element name="Mpeg4Profile" type="tt:Mpeg4Profile"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="H264Configuration">
  <xs:sequence>
    <xs:element name="GovLength" type="xs:int"/>
    <xs:element name="H264Profile" type="tt:H264Profile"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="VideoEncoderConfigurationOptions">
  <xs:sequence>
    <xs:element name="QualityRange" type="tt:IntRange"/>
    <xs:element name="JPEG" type="tt:JpegOptions" minOccurs="0"/>
    <xs:element name="MPEG4" type="tt:Mpeg4Options" minOccurs="0"/>
    <xs:element name="H264" type="tt:H264Options" minOccurs="0"/>
    <xs:element name="Extension" type="tt:VideoEncoderOptionsExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="VideoEncoderOptionsExtension">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="JPEG" type="tt:JpegOptions2" minOccurs="0"/>
    <xs:element name="MPEG4" type="tt:Mpeg4Options2" minOccurs="0"/>
    <xs:element name="H264" type="tt:H264Options2" minOccurs="0"/>
    <xs:element name="Extension" type="tt:VideoEncoderOptionsExtension2"
minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="VideoEncoderOptionsExtension2">

```

```

<xs:sequence>
  <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="JpegOptions">
  <xs:sequence>
    <xs:element name="ResolutionsAvailable" type="tt:VideoResolution"
maxOccurs="unbounded"/>
    <xs:element name="FrameRateRange" type="tt:IntRange"/>
    <xs:element name="EncodingIntervalRange" type="tt:IntRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="JpegOptions2">
  <xs:complexContent>
    <xs:extension base="tt:JpegOptions">
      <xs:sequence>
        <xs:element name="BitrateRange" type="tt:IntRange"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="Mpeg4Options">
  <xs:sequence>
    <xs:element name="ResolutionsAvailable" type="tt:VideoResolution"
maxOccurs="unbounded"/>
    <xs:element name="GovLengthRange" type="tt:IntRange"/>
    <xs:element name="FrameRateRange" type="tt:IntRange"/>
    <xs:element name="EncodingIntervalRange" type="tt:IntRange"/>
    <xs:element name="Mpeg4ProfilesSupported" type="tt:Mpeg4Profile"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Mpeg4Options2">
  <xs:complexContent>
    <xs:extension base="tt:Mpeg4Options">
      <xs:sequence>
        <xs:element name="BitrateRange" type="tt:IntRange"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="H264Options">
  <xs:sequence>
    <xs:element name="ResolutionsAvailable" type="tt:VideoResolution"
maxOccurs="unbounded"/>
    <xs:element name="GovLengthRange" type="tt:IntRange"/>
    <xs:element name="FrameRateRange" type="tt:IntRange"/>
    <xs:element name="EncodingIntervalRange" type="tt:IntRange"/>
    <xs:element name="H264ProfilesSupported" type="tt:H264Profile"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="H264Options2">
  <xs:complexContent>
    <xs:extension base="tt:H264Options">

```

```

        <xs:sequence>
          <xs:element name="BitrateRange" type="tt:IntRange"/>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType><!--=====--><!--
AudioSourceConfiguration --><!--=====-->
  <xs:complexType name="AudioSourceConfiguration">
    <xs:complexContent>
      <xs:extension base="tt:ConfigurationEntity">
        <xs:sequence>
          <xs:element name="SourceToken" type="tt:ReferenceToken"/>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType><!--=====-->
  <xs:complexType name="AudioSourceConfigurationOptions">
    <xs:sequence>
      <xs:element name="InputTokensAvailable" type="tt:ReferenceToken"
maxOccurs="unbounded"/>
      <xs:element name="Extension" type="tt:AudioSourceOptionsExtension"
minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType><!--=====-->
  <xs:complexType name="AudioSourceOptionsExtension">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType><!--=====--><!--
AudioEncoderConfiguration --><!--=====-->
  <xs:complexType name="AudioEncoderConfiguration">
    <xs:complexContent>
      <xs:extension base="tt:ConfigurationEntity">
        <xs:sequence>
          <xs:element name="Encoding" type="tt:AudioEncoding"/>
          <xs:element name="Bitrate" type="xs:int"/>
          <xs:element name="SampleRate" type="xs:int"/>
          <xs:element name="Multicast" type="tt:MulticastConfiguration"/>
          <xs:element name="SessionTimeout" type="xs:duration"/>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType><!--=====-->
  <xs:simpleType name="AudioEncoding">
    <xs:restriction base="xs:string">
      <xs:enumeration value="G711"/>
      <xs:enumeration value="G726"/>
      <xs:enumeration value="AAC"/>
    </xs:restriction>
  </xs:simpleType><!--=====-->
  <xs:complexType name="AudioEncoderConfigurationOptions">

```

```

<xs:sequence>
  <xs:element name="Options" type="tt:AudioEncoderConfigurationOption"
minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AudioEncoderConfigurationOption">
  <xs:sequence>
    <xs:element name="Encoding" type="tt:AudioEncoding"/>
    <xs:element name="BitrateList" type="tt:IntList"/>
    <xs:element name="SampleRateList" type="tt:IntList"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====--><!--
VideoAnalyticsConfiguration --><!--=====-->
<xs:complexType name="VideoAnalyticsConfiguration">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="AnalyticsEngineConfiguration"
type="tt:AnalyticsEngineConfiguration"/>
        <xs:element name="RuleEngineConfiguration" type="tt:RuleEngineConfiguration"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====--><!--
MetadataConfiguration --><!--=====-->
<xs:complexType name="MetadataConfiguration">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="PTZStatus" type="tt:PTZFilter" minOccurs="0"/>
        <xs:element name="Events" type="tt:EventSubscription" minOccurs="0"/>
        <xs:element name="Analytics" type="xs:boolean" minOccurs="0"/>
        <xs:element name="Multicast" type="tt:MulticastConfiguration"/>
        <xs:element name="SessionTimeout" type="xs:duration"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="PTZFilter">
  <xs:sequence>
    <xs:element name="Status" type="xs:boolean"/>
    <xs:element name="Position" type="xs:boolean"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="EventSubscription">
  <xs:sequence>
    <xs:element name="Filter" type="wsnt:FilterType" minOccurs="0"/>
    <xs:element name="SubscriptionPolicy" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"

```

```

maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="MetadataConfigurationOptions">
  <xs:sequence>
    <xs:element name="PTZStatusFilterOptions" type="tt:PTZStatusFilterOptions"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PTZStatusFilterOptions">
  <xs:sequence>
    <xs:element name="PanTiltStatusSupported" type="xs:boolean"/>
    <xs:element name="ZoomStatusSupported" type="xs:boolean"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="PanTiltPositionSupported" type="xs:boolean" minOccurs="0"/>
    <xs:element name="ZoomPositionSupported" type="xs:boolean" minOccurs="0"/>
    <xs:element name="Extension" type="tt:PTZStatusFilterOptionsExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PTZStatusFilterOptionsExtension">
  <xs:sequence>
    <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====--><!-- VideoOutput
--><!--=====-->
<xs:complexType name="VideoOutput">
  <xs:complexContent>
    <xs:extension base="tt:DeviceEntity">
      <xs:sequence>
        <xs:element name="Layout" type="tt:Layout"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====--><!--
VideoOutputConfiguration --><!--=====-->
<xs:complexType name="VideoOutputConfiguration">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====--><!--
VideoOutputConfigurationOptions --><!--=====-->

```



```

<xs:complexType name="VideoOutputConfigurationOptions">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====--><!--
VideoDecoderConfigurationOptions      --><!--=====-->
<xs:complexType name="VideoDecoderConfigurationOptions">
  <xs:sequence>
    <xs:element name="JpegDecOptions" type="tt:JpegDecOptions" minOccurs="0"/>
    <xs:element name="H264DecOptions" type="tt:H264DecOptions" minOccurs="0"/>
    <xs:element name="Mpeg4DecOptions" type="tt:Mpeg4DecOptions" minOccurs="0"/>
    <xs:element name="Extension" type="tt:VideoDecoderConfigurationOptionsExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="H264DecOptions">
  <xs:sequence>
    <xs:element name="ResolutionsAvailable" type="tt:VideoResolution"
maxOccurs="unbounded"/>
    <xs:element name="SupportedH264Profiles" type="tt:H264Profile"
maxOccurs="unbounded"/>
    <xs:element name="SupportedInputBitrate" type="tt:IntRange"/>
    <xs:element name="SupportedFrameRate" type="tt:IntRange"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="JpegDecOptions">
  <xs:sequence>
    <xs:element name="ResolutionsAvailable" type="tt:VideoResolution"
maxOccurs="unbounded"/>
    <xs:element name="SupportedInputBitrate" type="tt:IntRange"/>
    <xs:element name="SupportedFrameRate" type="tt:IntRange"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="Mpeg4DecOptions">
  <xs:sequence>
    <xs:element name="ResolutionsAvailable" type="tt:VideoResolution"
maxOccurs="unbounded"/>
    <xs:element name="SupportedMpeg4Profiles" type="tt:Mpeg4Profile"
maxOccurs="unbounded"/>
    <xs:element name="SupportedInputBitrate" type="tt:IntRange"/>
    <xs:element name="SupportedFrameRate" type="tt:IntRange"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="VideoDecoderConfigurationOptionsExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====--><!--
--><!--=====-->

```

AudioOutputs

```

<xs:complexType name="AudioOutput">
  <xs:complexContent>
    <xs:extension base="tt:DeviceEntity">
      <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====--><!--
AudioOutputConfiguration      --><!--=====-->
<xs:complexType name="AudioOutputConfiguration">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="OutputToken" type="tt:ReferenceToken"/>
        <xs:element name="SendPrimacy" type="xs:anyURI" minOccurs="0"/>
        <xs:element name="OutputLevel" type="xs:int"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====--><!--
AudioOutputConfigurationOptions      --><!--=====-->
<xs:complexType name="AudioOutputConfigurationOptions">
  <xs:sequence>
    <xs:element name="OutputTokensAvailable" type="tt:ReferenceToken"
maxOccurs="unbounded"/>
    <xs:element name="SendPrimacyOptions" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="OutputLevelRange" type="tt:IntRange"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====--><!--
AudioDecoderConfiguration      --><!--=====-->
<xs:complexType name="AudioDecoderConfiguration">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====--><!--
AudioDecoderConfigurationOptions      --><!--=====-->
<xs:complexType name="AudioDecoderConfigurationOptions">
  <xs:sequence>
    <xs:element name="AACDecOptions" type="tt:AACDecOptions" minOccurs="0"/>
    <xs:element name="G711DecOptions" type="tt:G711DecOptions" minOccurs="0"/>
    <xs:element name="G726DecOptions" type="tt:G726DecOptions" minOccurs="0"/>
    <xs:element name="Extension" type="tt:AudioDecoderConfigurationOptionsExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->

```



```

<xs:complexType name="G711DecOptions">
  <xs:sequence>
    <xs:element name="Bitrate" type="tt:IntList"/>
    <xs:element name="SampleRateRange" type="tt:IntList"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AACDecOptions">
  <xs:sequence>
    <xs:element name="Bitrate" type="tt:IntList"/>
    <xs:element name="SampleRateRange" type="tt:IntList"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="G726DecOptions">
  <xs:sequence>
    <xs:element name="Bitrate" type="tt:IntList"/>
    <xs:element name="SampleRateRange" type="tt:IntList"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AudioDecoderConfigurationOptionsExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====--><!-- Streaming
--><!--=====-->
<xs:complexType name="MulticastConfiguration">
  <xs:sequence>
    <xs:element name="Address" type="tt:IPAddress"/>
    <xs:element name="Port" type="xs:int"/>
    <xs:element name="TTL" type="xs:int"/>
    <xs:element name="AutoStart" type="xs:boolean"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="StreamSetup">
  <xs:sequence>
    <xs:element name="Stream" type="tt:StreamType"/>
    <xs:element name="Transport" type="tt:Transport"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="StreamType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="RTP-Unicast"/>
    <xs:enumeration value="RTP-Multicast"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Transport">
  <xs:sequence>

```

```

        <xs:element name="Protocol" type="tt:TransportProtocol"/>
        <xs:element name="Tunnel" type="tt:Transport" minOccurs="0"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="TransportProtocol">
    <xs:restriction base="xs:string">
        <xs:enumeration value="UDP"/>
        <xs:enumeration value="TCP"/>
        <xs:enumeration value="RTSP"/>
        <xs:enumeration value="HTTP"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="MediaUri">
    <xs:sequence>
        <xs:element name="Uri" type="xs:anyURI"/>
        <xs:element name="InvalidAfterConnect" type="xs:boolean"/>
        <xs:element name="InvalidAfterReboot" type="xs:boolean"/>
        <xs:element name="Timeout" type="xs:duration"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType>
<!--=====-->
<!-- End, Media Related Types -->
<!--=====-->
<!--=====-->
<!--=====-->
<!-- Discovery Related Types -->
<!--=====-->
<xs:simpleType name="ScopeDefinition">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Fixed"/>
        <xs:enumeration value="Configurable"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Scope">
    <xs:sequence>
        <xs:element name="ScopeDef" type="tt:ScopeDefinition"/>
        <xs:element name="ScopeItem" type="xs:anyURI"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="DiscoveryMode">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Discoverable"/>
        <xs:enumeration value="NonDiscoverable"/>
    </xs:restriction>
</xs:simpleType>
<!--=====-->
<!-- End, Discovery Related Types -->
<!--=====-->
<!--=====-->
<!--=====-->
<!-- Network Related Types -->
<!--=====-->
<xs:complexType name="NetworkInterface">
    <xs:complexContent>
        <xs:extension base="tt:DeviceEntity">
            <xs:sequence>
                <xs:element name="Enabled" type="xs:boolean"/>
                <xs:element name="Info" type="tt:NetworkInterfaceInfo" minOccurs="0"/>
                <xs:element name="Link" type="tt:NetworkInterfaceLink" minOccurs="0"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

```

```

        <xs:element name="IPv4" type="tt:IPv4NetworkInterface" minOccurs="0"/>
        <xs:element name="IPv6" type="tt:IPv6NetworkInterface" minOccurs="0"/>
        <xs:element name="Extension" type="tt:NetworkInterfaceExtension"
minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:extension>
</xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkInterfaceExtension">
    <xs:sequence>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="InterfaceType" type="tt:IANA-IfTypes"/>
        <xs:element name="Dot3" type="tt:Dot3Configuration" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="Dot11" type="tt:Dot11Configuration" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="Extension" type="tt:NetworkInterfaceExtension2" minOccurs="0"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="NetworkInterfaceConfigPriority">
    <xs:restriction base="xs:integer">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="31"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Dot3Configuration">
    <xs:sequence><!-- Placeholder for 802.3 configuration -->
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkInterfaceExtension2">
    <xs:sequence>
        <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkInterfaceLink">
    <xs:sequence>
        <xs:element name="AdminSettings" type="tt:NetworkInterfaceConnectionSetting"/>
        <xs:element name="OperSettings" type="tt:NetworkInterfaceConnectionSetting"/>
        <xs:element name="InterfaceType" type="tt:IANA-IfTypes"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkInterfaceConnectionSetting">
    <xs:sequence>
        <xs:element name="AutoNegotiation" type="xs:boolean"/>
        <xs:element name="Speed" type="xs:int"/>
        <xs:element name="Duplex" type="tt:Duplex"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="Duplex">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Full"/>
        <xs:enumeration value="Half"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="IANA-IfTypes">
    <xs:restriction base="xs:int"/>

```

```

</xs:simpleType><!--=====-->
<xs:complexType name="NetworkInterfaceInfo">
  <xs:sequence>
    <xs:element name="Name" type="xs:string" minOccurs="0"/>
    <xs:element name="HwAddress" type="tt:HwAddress"/>
    <xs:element name="MTU" type="xs:int" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="IPv6NetworkInterface">
  <xs:sequence>
    <xs:element name="Enabled" type="xs:boolean"/>
    <xs:element name="Config" type="tt:IPv6Configuration" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="IPv4NetworkInterface">
  <xs:sequence>
    <xs:element name="Enabled" type="xs:boolean"/>
    <xs:element name="Config" type="tt:IPv4Configuration"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="IPv4Configuration">
  <xs:sequence>
    <xs:element name="Manual" type="tt:PrefixedIPv4Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="LinkLocal" type="tt:PrefixedIPv4Address" minOccurs="0"/>
    <xs:element name="FromDHCP" type="tt:PrefixedIPv4Address" minOccurs="0"/>
    <xs:element name="DHCP" type="xs:boolean"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="IPv6Configuration">
  <xs:sequence>
    <xs:element name="AcceptRouterAdvert" type="xs:boolean" minOccurs="0"/>
    <xs:element name="DHCP" type="tt:IPv6DHCPConfiguration"/>
    <xs:element name="Manual" type="tt:PrefixedIPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="LinkLocal" type="tt:PrefixedIPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="FromDHCP" type="tt:PrefixedIPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="FromRA" type="tt:PrefixedIPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:IPv6ConfigurationExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="IPv6ConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="IPv6DHCPConfiguration">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Auto"/>
    <xs:enumeration value="Stateful"/>
    <xs:enumeration value="Stateless"/>
    <xs:enumeration value="Off"/>
  </xs:restriction>
</xs:simpleType><!--=====-->

```

```

<xs:complexType name="NetworkProtocol">
  <xs:sequence>
    <xs:element name="Name" type="tt:NetworkProtocolType"/>
    <xs:element name="Enabled" type="xs:boolean"/>
    <xs:element name="Port" type="xs:int" maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:NetworkProtocolExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkProtocolExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="NetworkProtocolType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="HTTP"/>
    <xs:enumeration value="HTTPS"/>
    <xs:enumeration value="RTSP"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="NetworkHostType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="IPv4"/>
    <xs:enumeration value="IPv6"/>
    <xs:enumeration value="DNS"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="NetworkHost">
  <xs:sequence>
    <xs:element name="Type" type="tt:NetworkHostType"/>
    <xs:element name="IPv4Address" type="tt:IPv4Address" minOccurs="0"/>
    <xs:element name="IPv6Address" type="tt:IPv6Address" minOccurs="0"/>
    <xs:element name="DNSName" type="tt:DNSName" minOccurs="0"/>
    <xs:element name="Extension" type="tt:NetworkHostExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkHostExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="IPAddress">
  <xs:sequence>
    <xs:element name="Type" type="tt:IPType"/>
    <xs:element name="IPv4Address" type="tt:IPv4Address" minOccurs="0"/>
    <xs:element name="IPv6Address" type="tt:IPv6Address" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PrefixedIPv4Address">
  <xs:sequence>
    <xs:element name="Address" type="tt:IPv4Address"/>
    <xs:element name="PrefixLength" type="xs:int"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="IPv4Address">
  <xs:restriction base="xs:token"/>
</xs:simpleType><!--=====-->
<xs:complexType name="PrefixedIPv6Address">

```



```

<xs:sequence>
  <xs:element name="Address" type="tt:IPv6Address"/>
  <xs:element name="PrefixLength" type="xs:int"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="IPv6Address">
  <xs:restriction base="xs:token"/>
</xs:simpleType><!--=====-->
<xs:simpleType name="HwAddress">
  <xs:restriction base="xs:token"/>
</xs:simpleType><!--=====-->
<xs:simpleType name="IPType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="IPv4"/>
    <xs:enumeration value="IPv6"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="DNSName">
  <xs:restriction base="xs:token"/>
</xs:simpleType><!--=====-->
<xs:complexType name="HostnameInformation">
  <xs:sequence>
    <xs:element name="FromDHCP" type="xs:boolean"/>
    <xs:element name="Name" type="xs:token" minOccurs="0"/>
    <xs:element name="Extension" type="tt:HostnameInformationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="HostnameInformationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="DNSInformation">
  <xs:sequence>
    <xs:element name="FromDHCP" type="xs:boolean"/>
    <xs:element name="SearchDomain" type="xs:token" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="DNSFromDHCP" type="tt:IPAddress" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="DNSManual" type="tt:IPAddress" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:DNSInformationExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="DNSInformationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="NTPInformation">
  <xs:sequence>
    <xs:element name="FromDHCP" type="xs:boolean"/>
    <xs:element name="NTPFromDHCP" type="tt:NetworkHost" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="NTPManual" type="tt:NetworkHost" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:NTPInformationExtension" minOccurs="0"/>
  </xs:sequence>

```

```

</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NTPInformationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="Domain">
  <xs:restriction base="xs:token"/>
</xs:simpleType><!--=====-->
<xs:simpleType name="IPAddressFilterType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Allow"/>
    <xs:enumeration value="Deny"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="DynamicDNSInformation">
  <xs:sequence>
    <xs:element name="Type" type="tt:DynamicDNSType"/>
    <xs:element name="Name" type="tt:DNSName" minOccurs="0"/>
    <xs:element name="TTL" type="xs:duration" minOccurs="0"/>
    <xs:element name="Extension" type="tt:DynamicDNSInformationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="DynamicDNSInformationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="DynamicDNSType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="NoUpdate"/>
    <xs:enumeration value="ClientUpdates"/>
    <xs:enumeration value="ServerUpdates"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="NetworkInterfaceSetConfiguration">
  <xs:sequence>
    <xs:element name="Enabled" type="xs:boolean" minOccurs="0"/>
    <xs:element name="Link" type="tt:NetworkInterfaceConnectionSetting" minOccurs="0"/>
    <xs:element name="MTU" type="xs:int" minOccurs="0"/>
    <xs:element name="IPv4" type="tt:IPv4NetworkInterfaceSetConfiguration"
minOccurs="0"/>
    <xs:element name="IPv6" type="tt:IPv6NetworkInterfaceSetConfiguration"
minOccurs="0"/>
    <xs:element name="Extension" type="tt:NetworkInterfaceSetConfigurationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkInterfaceSetConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Dot3" type="tt:Dot3Configuration" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Dot11" type="tt:Dot11Configuration" minOccurs="0"

```



```

maxOccurs="unbounded"/>
  <xs:element name="Extension" type="tt:NetworkInterfaceSetConfigurationExtension2"
minOccurs="0"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="IPv6NetworkInterfaceSetConfiguration">
  <xs:sequence>
    <xs:element name="Enabled" type="xs:boolean" minOccurs="0"/>
    <xs:element name="AcceptRouterAdvert" type="xs:boolean" minOccurs="0"/>
    <xs:element name="Manual" type="tt:PrefixedIPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="DHCP" type="tt:IPv6DHCPConfiguration" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="IPv4NetworkInterfaceSetConfiguration">
  <xs:sequence>
    <xs:element name="Enabled" type="xs:boolean" minOccurs="0"/>
    <xs:element name="Manual" type="tt:PrefixedIPv4Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="DHCP" type="xs:boolean" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkGateway">
  <xs:sequence>
    <xs:element name="IPv4Address" type="tt:IPv4Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="IPv6Address" type="tt:IPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkZeroConfiguration">
  <xs:sequence>
    <xs:element name="InterfaceToken" type="tt:ReferenceToken"/>
    <xs:element name="Enabled" type="xs:boolean"/>
    <xs:element name="Addresses" type="tt:IPv4Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:NetworkZeroConfigurationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkZeroConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="IPAddressFilter">
  <xs:sequence>
    <xs:element name="Type" type="tt:IPAddressFilterType"/>
    <xs:element name="IPv4Address" type="tt:PrefixedIPv4Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="IPv6Address" type="tt:PrefixedIPv6Address" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:IPAddressFilterExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="IPAddressFilterExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->

```

```

</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Dot11Configuration">
  <xs:sequence>
    <xs:element name="SSID" type="tt:Dot11SSIDType"/>
    <xs:element name="Mode" type="tt:Dot11StationMode"/>
    <xs:element name="Alias" type="tt:Name"/>
    <xs:element name="Priority" type="tt:NetworkInterfaceConfigPriority"/>
    <xs:element name="Security" type="tt:Dot11SecurityConfiguration"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="Dot11SSIDType">
  <xs:restriction base="xs:hexBinary">
    <xs:minLength value="1"/>
    <xs:maxLength value="32"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="Dot11StationMode">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Ad-hoc"/>
    <xs:enumeration value="Infrastructure"/>
    <xs:enumeration value="Extended"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Dot11SecurityConfiguration">
  <xs:sequence>
    <xs:element name="Mode" type="tt:Dot11SecurityMode"/>
    <xs:element name="Algorithm" type="tt:Dot11Cipher" minOccurs="0"/>
    <xs:element name="PSK" type="tt:Dot11PSKSet" minOccurs="0"/>
    <xs:element name="Dot1X" type="tt:ReferenceToken" minOccurs="0"/>
    <xs:element name="Extension" type="tt:Dot11SecurityConfigurationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="Dot11SecurityConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="Dot11SecurityMode">
  <xs:restriction base="xs:string">
    <xs:enumeration value="None"/>
    <xs:enumeration value="WEP"/>
    <xs:enumeration value="PSK"/>
    <xs:enumeration value="Dot1X"/>
    <xs:enumeration value="Extended"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="Dot11Cipher">
  <xs:restriction base="xs:string">
    <xs:enumeration value="CCMP"/>
    <xs:enumeration value="TKIP"/>
    <xs:enumeration value="Any"/>
    <xs:enumeration value="Extended"/>
  </xs:restriction>
</xs:simpleType><!--=====-->

```

```

<xs:simpleType name="Dot11PSK">
  <xs:restriction base="xs:hexBinary">
    <xs:length value="32"/><!-- IEEE802.11 H.4.1: The RSNA PSK consists of 256 bits, or
64 octets when represented in hex. -->
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="Dot11PSKPassphrase">
  <xs:restriction base="xs:string">
    <xs:pattern value="[ -~]{8,63}"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Dot11PSKSet">
  <xs:sequence>
    <xs:element name="Key" type="tt:Dot11PSK" minOccurs="0"/>
    <xs:element name="Passphrase" type="tt:Dot11PSKPassphrase" minOccurs="0"/>
    <xs:element name="Extension" type="tt:Dot11PSKSetExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="Dot11PSKSetExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkInterfaceSetConfigurationExtension2">
  <xs:sequence>
    <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Dot11Capabilities">
  <xs:sequence>
    <xs:element name="TKIP" type="xs:boolean"/>
    <xs:element name="ScanAvailableNetworks" type="xs:boolean"/>
    <xs:element name="MultipleConfiguration" type="xs:boolean"/>
    <xs:element name="AdHocStationMode" type="xs:boolean"/>
    <xs:element name="WEP" type="xs:boolean"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="Dot11SignalStrength">
  <xs:restriction base="xs:string">
    <xs:enumeration value="None"/>
    <xs:enumeration value="Very Bad"/>
    <xs:enumeration value="Bad"/>
    <xs:enumeration value="Good"/>
    <xs:enumeration value="Very Good"/>
    <xs:enumeration value="Extended"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Dot11Status">
  <xs:sequence>
    <xs:element name="SSID" type="tt:Dot11SSIDType"/>
    <xs:element name="BSSID" type="xs:string" minOccurs="0"/>
    <xs:element name="PairCipher" type="tt:Dot11Cipher" minOccurs="0"/>
    <xs:element name="GroupCipher" type="tt:Dot11Cipher" minOccurs="0"/>
    <xs:element name="SignalStrength" type="tt:Dot11SignalStrength" minOccurs="0"/>
    <xs:element name="ActiveConfigAlias" type="tt:ReferenceToken"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"

```

```

maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="Dot11AuthAndMangementSuite">
  <xs:restriction base="xs:string">
    <xs:enumeration value="None"/>
    <xs:enumeration value="Dot1X"/>
    <xs:enumeration value="PSK"/>
    <xs:enumeration value="Extended"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Dot11AvailableNetworks">
  <xs:sequence>
    <xs:element name="SSID" type="tt:Dot11SSIDType"/>
    <xs:element name="BSSID" type="xs:string" minOccurs="0"/>
    <xs:element name="AuthAndMangementSuite" type="tt:Dot11AuthAndMangementSuite"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="PairCipher" type="tt:Dot11Cipher" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="GroupCipher" type="tt:Dot11Cipher" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="SignalStrength" type="tt:Dot11SignalStrength" minOccurs="0"/>
    <xs:element name="Extension" type="tt:Dot11AvailableNetworksExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="Dot11AvailableNetworksExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!--=====-->
<!-- End, network Related Types -->
<!--=====-->
<!--=====-->
<!-- Capabilities Related Types -->
<!--=====-->
<xs:simpleType name="CapabilityCategory">
  <xs:restriction base="xs:string">
    <xs:enumeration value="All"/>
    <xs:enumeration value="Analytics"/>
    <xs:enumeration value="Device"/>
    <xs:enumeration value="Events"/>
    <xs:enumeration value="Imaging"/>
    <xs:enumeration value="Media"/>
    <xs:enumeration value="PTZ"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="Capabilities">
  <xs:sequence>
    <xs:element name="Analytics" type="tt:AnalyticsCapabilities" minOccurs="0"/>
    <xs:element name="Device" type="tt:DeviceCapabilities" minOccurs="0"/>
    <xs:element name="Events" type="tt:EventCapabilities" minOccurs="0"/>
    <xs:element name="Imaging" type="tt:ImagingCapabilities" minOccurs="0"/>
    <xs:element name="Media" type="tt:MediaCapabilities" minOccurs="0"/>
    <xs:element name="PTZ" type="tt:PTZCapabilities" minOccurs="0"/>
    <xs:element name="Extension" type="tt:CapabilitiesExtension" minOccurs="0"/>
  </xs:sequence>

```

```

<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="CapabilitiesExtension">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="DeviceIO" type="tt:DeviceIOCapabilities" minOccurs="0"/>
    <xs:element name="Display" type="tt:DisplayCapabilities" minOccurs="0"/>
    <xs:element name="Recording" type="tt:RecordingCapabilities" minOccurs="0"/>
    <xs:element name="Search" type="tt:SearchCapabilities" minOccurs="0"/>
    <xs:element name="Replay" type="tt:ReplayCapabilities" minOccurs="0"/>
    <xs:element name="Receiver" type="tt:ReceiverCapabilities" minOccurs="0"/>
    <xs:element name="AnalyticsDevice" type="tt:AnalyticsDeviceCapabilities"
minOccurs="0"/>
    <xs:element name="Extensions" type="tt:CapabilitiesExtension2" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="CapabilitiesExtension2">
  <xs:sequence>
    <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsCapabilities">
  <xs:sequence>
    <xs:element name="XAddr" type="xs:anyURI"/>
    <xs:element name="RuleSupport" type="xs:boolean"/>
    <xs:element name="AnalyticsModuleSupport" type="xs:boolean"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="DeviceCapabilities">
  <xs:sequence>
    <xs:element name="XAddr" type="xs:anyURI"/>
    <xs:element name="Network" type="tt:NetworkCapabilities" minOccurs="0"/>
    <xs:element name="System" type="tt:SystemCapabilities" minOccurs="0"/>
    <xs:element name="IO" type="tt:IOCapabilities" minOccurs="0"/>
    <xs:element name="Security" type="tt:SecurityCapabilities" minOccurs="0"/>
    <xs:element name="Extension" type="tt:DeviceCapabilitiesExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="DeviceCapabilitiesExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="EventCapabilities">
  <xs:sequence>
    <xs:element name="XAddr" type="xs:anyURI"/>
    <xs:element name="WSSubscriptionPolicySupport" type="xs:boolean"/>
    <xs:element name="WSPullPointSupport" type="xs:boolean"/>
    <xs:element name="WSPausableSubscriptionManagerInterfaceSupport"
type="xs:boolean"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->

```



```

<xs:complexType name="IOCapabilities">
  <xs:sequence>
    <xs:element name="InputConnectors" type="xs:int" minOccurs="0"/>
    <xs:element name="RelayOutputs" type="xs:int" minOccurs="0"/>
    <xs:element name="Extension" type="tt:IOCapabilitiesExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="IOCapabilitiesExtension">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Auxiliary" type="xs:boolean" minOccurs="0"/>
    <xs:element name="AuxiliaryCommands" type="tt:AuxiliaryData" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:IOCapabilitiesExtension2"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="IOCapabilitiesExtension2">
  <xs:sequence>
    <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="MediaCapabilities">
  <xs:sequence>
    <xs:element name="XAddr" type="xs:anyURI"/>
    <xs:element name="StreamingCapabilities" type="tt:RealTimeStreamingCapabilities"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:MediaCapabilitiesExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="MediaCapabilitiesExtension">
  <xs:sequence>
    <xs:element name="ProfileCapabilities" type="tt:ProfileCapabilities"/>
    <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RealTimeStreamingCapabilities">
  <xs:sequence>
    <xs:element name="RTPMulticast" type="xs:boolean" minOccurs="0"/>
    <xs:element name="RTP_TCP" type="xs:boolean" minOccurs="0"/>
    <xs:element name="RTP_RTSP_TCP" type="xs:boolean" minOccurs="0"/>
    <xs:element name="Extension" type="tt:RealTimeStreamingCapabilitiesExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RealTimeStreamingCapabilitiesExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ProfileCapabilities">
  <xs:sequence>
    <xs:element name="MaximumNumberOfProfiles" type="xs:int"/>

```

```

        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkCapabilities">
    <xs:sequence>
        <xs:element name="IPFilter" type="xs:boolean" minOccurs="0"/>
        <xs:element name="ZeroConfiguration" type="xs:boolean" minOccurs="0"/>
        <xs:element name="IPVersion6" type="xs:boolean" minOccurs="0"/>
        <xs:element name="DynDNS" type="xs:boolean" minOccurs="0"/>
        <xs:element name="Extension" type="tt:NetworkCapabilitiesExtension" minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkCapabilitiesExtension">
    <xs:sequence>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="Dot11Configuration" type="xs:boolean" minOccurs="0"/>
        <xs:element name="Extension" type="tt:NetworkCapabilitiesExtension2"
minOccurs="0"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="NetworkCapabilitiesExtension2">
    <xs:sequence>
        <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="SecurityCapabilities">
    <xs:sequence>
        <xs:element name="TLS1.1" type="xs:boolean"/>
        <xs:element name="TLS1.2" type="xs:boolean"/>
        <xs:element name="OnboardKeyGeneration" type="xs:boolean"/>
        <xs:element name="AccessPolicyConfig" type="xs:boolean"/>
        <xs:element name="X.509Token" type="xs:boolean"/>
        <xs:element name="SAMLToken" type="xs:boolean"/>
        <xs:element name="KerberosToken" type="xs:boolean"/>
        <xs:element name="RELToken" type="xs:boolean"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="Extension" type="tt:SecurityCapabilitiesExtension" minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SecurityCapabilitiesExtension">
    <xs:sequence>
        <xs:element name="TLS1.0" type="xs:boolean"/>
        <xs:element name="Extension" type="tt:SecurityCapabilitiesExtension2"
minOccurs="0"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="SecurityCapabilitiesExtension2">
    <xs:sequence>
        <xs:element name="Dot1X" type="xs:boolean"/>
        <xs:element name="SupportedEAPMethod" type="xs:int" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="RemoteUserHandling" type="xs:boolean"/>
        <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>

```



```

</xs:complexType><!--=====-->
<xs:complexType name="SystemCapabilities">
  <xs:sequence>
    <xs:element name="DiscoveryResolve" type="xs:boolean"/>
    <xs:element name="DiscoveryBye" type="xs:boolean"/>
    <xs:element name="RemoteDiscovery" type="xs:boolean"/>
    <xs:element name="SystemBackup" type="xs:boolean"/>
    <xs:element name="SystemLogging" type="xs:boolean"/>
    <xs:element name="FirmwareUpgrade" type="xs:boolean"/>
    <xs:element name="SupportedVersions" type="tt:OnvifVersion"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:SystemCapabilitiesExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SystemCapabilitiesExtension">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="HttpFirmwareUpgrade" type="xs:boolean" minOccurs="0"/>
    <xs:element name="HttpSystemBackup" type="xs:boolean" minOccurs="0"/>
    <xs:element name="HttpSystemLogging" type="xs:boolean" minOccurs="0"/>
    <xs:element name="HttpSupportInformation" type="xs:boolean" minOccurs="0"/>
    <xs:element name="Extension" type="tt:SystemCapabilitiesExtension2"
minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="SystemCapabilitiesExtension2">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="OnvifVersion">
  <xs:sequence>
    <xs:element name="Major" type="xs:int"/>
    <xs:element name="Minor" type="xs:int"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ImagingCapabilities">
  <xs:sequence>
    <xs:element name="XAddr" type="xs:anyURI"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PTZCapabilities">
  <xs:sequence>
    <xs:element name="XAddr" type="xs:anyURI"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="DeviceIOCapabilities">
  <xs:sequence>
    <xs:element name="XAddr" type="xs:anyURI"/>
    <xs:element name="VideoSources" type="xs:int"/>
    <xs:element name="VideoOutputs" type="xs:int"/>
    <xs:element name="AudioSources" type="xs:int"/>
    <xs:element name="AudioOutputs" type="xs:int"/>
    <xs:element name="RelayOutputs" type="xs:int"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"

```

```

maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="DisplayCapabilities">
<xs:sequence>
<xs:element name="XAddr" type="xs:anyURI"/>
<xs:element name="FixedLayout" type="xs:boolean"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingCapabilities">
<xs:sequence>
<xs:element name="XAddr" type="xs:anyURI"/>
<xs:element name="ReceiverSource" type="xs:boolean"/>
<xs:element name="MediaProfileSource" type="xs:boolean"/>
<xs:element name="DynamicRecordings" type="xs:boolean"/>
<xs:element name="DynamicTracks" type="xs:boolean"/>
<xs:element name="MaxStringLength" type="xs:int"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SearchCapabilities">
<xs:sequence>
<xs:element name="XAddr" type="xs:anyURI"/>
<xs:element name="MetadataSearch" type="xs:boolean"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ReplayCapabilities">
<xs:sequence>
<xs:element name="XAddr" type="xs:anyURI"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ReceiverCapabilities">
<xs:sequence>
<xs:element name="XAddr" type="xs:anyURI"/>
<xs:element name="RTP_Multicast" type="xs:boolean"/>
<xs:element name="RTP_TCP" type="xs:boolean"/>
<xs:element name="RTP_RTSP_TCP" type="xs:boolean"/>
<xs:element name="SupportedReceivers" type="xs:int"/>
<xs:element name="MaximumRTSPURILength" type="xs:int"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsDeviceCapabilities">
<xs:sequence>
<xs:element name="XAddr" type="xs:anyURI"/>
<xs:element name="RuleSupport" type="xs:boolean" minOccurs="0"/>
<xs:element name="Extension" type="tt:AnalyticsDeviceExtension" minOccurs="0"/>
</xs:sequence>

```

```

    <xs:anyAttribute processContents="lax"/>
  </xs:complexType><!--=====-->
  <xs:complexType name="AnalyticsDeviceExtension">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
<!--=====-->
<!--End, Capabilities Related Types-->
<!--=====-->
<!--=====-->
<!--=====-->
<!-- System Related Types -->
<!--=====-->
  <xs:simpleType name="SystemLogType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="System"/>
      <xs:enumeration value="Access"/>
    </xs:restriction>
  </xs:simpleType><!--=====-->
  <xs:complexType name="SystemLog">
    <xs:sequence>
      <xs:element name="Binary" type="tt:AttachmentData" minOccurs="0"/>
      <xs:element name="String" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="SupportInformation">
    <xs:sequence>
      <xs:element name="Binary" type="tt:AttachmentData" minOccurs="0"/>
      <xs:element name="String" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="BinaryData">
    <xs:sequence>
      <xs:element name="Data" type="xs:base64Binary" nillable="false"/>
    </xs:sequence>
    <xs:attribute ref="xmime:contentType" use="optional"/>
  </xs:complexType><!--=====-->
  <xs:complexType name="AttachmentData">
    <xs:sequence>
      <xs:element ref="xop:Include"/>
    </xs:sequence>
    <xs:attribute ref="xmime:contentType" use="optional"/>
  </xs:complexType><!--=====-->
  <xs:complexType name="BackupFile">
    <xs:sequence>
      <xs:element name="Name" type="xs:string"/>
      <xs:element name="Data" type="tt:AttachmentData"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="SystemLogUriList">
    <xs:sequence>
      <xs:element name="SystemLog" type="tt:SystemLogUri" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType><!--=====-->
  <xs:complexType name="SystemLogUri">
    <xs:sequence>
      <xs:element name="Type" type="tt:SystemLogType"/>
      <xs:element name="Uri" type="xs:anyURI"/>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"

```

```

maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="FactoryDefaultType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Hard"/>
    <xs:enumeration value="Soft"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="SetDateTimeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Manual"/>
    <xs:enumeration value="NTP"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="SystemDateTime">
  <xs:sequence>
    <xs:element name="DateTimeType" type="tt:SetDateTimeType"/>
    <xs:element name="DaylightSavings" type="xs:boolean"/>
    <xs:element name="TimeZone" type="tt:TimeZone" minOccurs="0"/>
    <xs:element name="UTCDateTime" type="tt:DateTime" minOccurs="0"/>
    <xs:element name="LocalDateTime" type="tt:DateTime" minOccurs="0"/>
    <xs:element name="Extension" type="tt:SystemDateTimeExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SystemDateTimeExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="DateTime">
  <xs:sequence>
    <xs:element name="Time" type="tt:Time"/>
    <xs:element name="Date" type="tt:Date"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Date">
  <xs:sequence>
    <xs:element name="Year" type="xs:int"/>
    <xs:element name="Month" type="xs:int"/>
    <xs:element name="Day" type="xs:int"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Time">
  <xs:sequence>
    <xs:element name="Hour" type="xs:int"/>
    <xs:element name="Minute" type="xs:int"/>
    <xs:element name="Second" type="xs:int"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="TimeZone">
  <xs:sequence>
    <xs:element name="TZ" type="xs:token"/>
  </xs:sequence>
</xs:complexType>
<!--=====-->
<!-- End, System Related Types -->
<!--=====-->
<!--=====-->

```

```

<!--=====-->
<!-- RemoteUser Handling Types -->
<!--=====-->
  <xs:complexType name="RemoteUser">
    <xs:sequence>
      <xs:element name="Username" type="xs:string"/>
      <xs:element name="Password" type="xs:string" minOccurs="0"/>
      <xs:element name="UseDerivedPassword" type="xs:boolean"/>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>
<!--=====-->
<!-- End, RemoteUser Handling Types -->
<!--=====-->
<!--=====-->
<!-- UserToken Handling Types -->
<!--=====-->
  <xs:simpleType name="UserLevel">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Administrator"/>
      <xs:enumeration value="Operator"/>
      <xs:enumeration value="User"/>
      <xs:enumeration value="Anonymous"/>
      <xs:enumeration value="Extended"/>
    </xs:restriction>
  </xs:simpleType>
  <!--=====-->
  <xs:complexType name="User">
    <xs:sequence>
      <xs:element name="Username" type="xs:string"/>
      <xs:element name="Password" type="xs:string" minOccurs="0"/>
      <xs:element name="UserLevel" type="tt:UserLevel"/>
      <xs:element name="Extension" type="tt:UserExtension" minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>
  <!--=====-->
  <xs:complexType name="UserExtension">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
<!--=====-->
<!-- End, UserToken Handling Types -->
<!--=====-->
<!--=====-->
<!-- Security Management Types -->
<!--=====-->
  <xs:complexType name="CertificateGenerationParameters">
    <xs:sequence>
      <xs:element name="CertificateID" type="xs:token" minOccurs="0"/>
      <xs:element name="Subject" type="xs:string" minOccurs="0"/>
      <xs:element name="ValidNotBefore" type="xs:token" minOccurs="0"/>
      <xs:element name="ValidNotAfter" type="xs:token" minOccurs="0"/>
      <xs:element name="Extension" type="tt:CertificateGenerationParametersExtension"
minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>
  <!--=====-->
  <xs:complexType name="CertificateGenerationParametersExtension">
    <xs:sequence>

```



```

        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Certificate">
    <xs:sequence>
        <xs:element name="CertificateID" type="xs:token"/>
        <xs:element name="Certificate" type="tt:BinaryData"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="CertificateStatus">
    <xs:sequence>
        <xs:element name="CertificateID" type="xs:token"/>
        <xs:element name="Status" type="xs:boolean"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="CertificateWithPrivateKey">
    <xs:sequence>
        <xs:element name="CertificateID" type="xs:token" minOccurs="0"/>
        <xs:element name="Certificate" type="tt:BinaryData"/>
        <xs:element name="PrivateKey" type="tt:BinaryData"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="CertificateInformation">
    <xs:sequence>
        <xs:element name="CertificateID" type="xs:token"/>
        <xs:element name="IssuerDN" type="xs:string" minOccurs="0"/>
        <xs:element name="SubjectDN" type="xs:string" minOccurs="0"/>
        <xs:element name="KeyUsage" type="tt:CertificateUsage" minOccurs="0"/>
        <xs:element name="ExtendedKeyUsage" type="tt:CertificateUsage" minOccurs="0"/>
        <xs:element name="KeyLength" type="xs:int" minOccurs="0"/>
        <xs:element name="Version" type="xs:string" minOccurs="0"/>
        <xs:element name="SerialNum" type="xs:string" minOccurs="0"/>
        <xs:element name="SignatureAlgorithm" type="xs:string" minOccurs="0"/>
        <xs:element name="Validity" type="tt:DateTimeRange" minOccurs="0"/>
        <xs:element name="Extension" type="tt:CertificateInformationExtension"
minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="CertificateUsage">
    <xs:simpleContent>
        <xs:extension base="xs:string">
            <xs:attribute name="Critical" type="xs:boolean" use="required"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType><!--=====-->
<xs:complexType name="CertificateInformationExtension">
    <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<!--=====-->
<!--End, Security management Types -->
<!--=====-->

```

```

<!--=====-->
<!-- Start Dot1X related Types -->
<!--=====-->
<xs:complexType name="Dot1XConfiguration">
  <xs:sequence>
    <xs:element name="Dot1XConfigurationToken" type="tt:ReferenceToken"/>
    <xs:element name="Identity" type="xs:string"/>
    <xs:element name="AnonymousID" type="xs:string" minOccurs="0"/>
    <xs:element name="EAPMethod" type="xs:int"/>
    <xs:element name="CACertificateID" type="xs:token" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="EAPMethodConfiguration" type="tt:EAPMethodConfiguration"
minOccurs="0"/>
    <xs:element name="Extension" type="tt:Dot1XConfigurationExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="Dot1XConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="EAPMethodConfiguration">
  <xs:sequence>
    <xs:element name="TLSConfiguration" type="tt:TLSConfiguration" minOccurs="0"/>
    <xs:element name="Password" type="xs:string" minOccurs="0"/>
    <xs:element name="Extension" type="tt:EapMethodExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="EapMethodExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="TLSConfiguration">
  <xs:sequence>
    <xs:element name="CertificateID" type="xs:token"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="GenericEapPwdConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!--=====-->
<!--End, Dot1X related Types -->
<!--=====-->
<!-- Start IO management Types -->
<!--=====-->
<xs:simpleType name="RelayLogicalState">
  <xs:restriction base="xs:string">
    <xs:enumeration value="active"/>
    <xs:enumeration value="inactive"/>
  </xs:restriction>

```



```

</xs:simpleType><!--=====-->
<xs:simpleType name="RelayIdleState">
  <xs:restriction base="xs:string">
    <xs:enumeration value="closed"/>
    <xs:enumeration value="open"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="RelayOutputSettings">
  <xs:sequence>
    <xs:element name="Mode" type="tt:RelayMode"/>
    <xs:element name="DelayTime" type="xs:duration"/>
    <xs:element name="IdleState" type="tt:RelayIdleState"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="RelayMode">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Monostable"/>
    <xs:enumeration value="Bistable"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="RelayOutput">
  <xs:complexContent>
    <xs:extension base="tt:DeviceEntity">
      <xs:sequence>
        <xs:element name="Properties" type="tt:RelayOutputSettings"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<!--=====-->
<!-- End, IO management Types -->
<!--=====-->
<!--=====-->
<!-- Start PTZ Related Types -->
<!--=====-->
<xs:complexType name="PTZNode">
  <xs:complexContent>
    <xs:extension base="tt:DeviceEntity">
      <xs:sequence>
        <xs:element name="Name" type="tt:Name" minOccurs="0"/>
        <xs:element name="SupportedPTZSpaces" type="tt:PTZSpaces"/>
        <xs:element name="MaximumNumberOfPresets" type="xs:int"/>
        <xs:element name="HomeSupported" type="xs:boolean"/>
        <xs:element name="AuxiliaryCommands" type="tt:AuxiliaryData" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="Extension" type="tt:PTZNodeExtension" minOccurs="0"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="PTZNodeExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PTZConfiguration">
  <xs:complexContent>

```

```

<xs:extension base="tt:ConfigurationEntity">
  <xs:sequence>
    <xs:element name="NodeToken" type="tt:ReferenceToken"/>
    <xs:element name="DefaultAbsolutePanTiltPositionSpace" type="xs:anyURI"
minOccurs="0"/>
    <xs:element name="DefaultAbsoluteZoomPositionSpace" type="xs:anyURI"
minOccurs="0"/>
    <xs:element name="DefaultRelativePanTiltTranslationSpace" type="xs:anyURI"
minOccurs="0"/>
    <xs:element name="DefaultRelativeZoomTranslationSpace" type="xs:anyURI"
minOccurs="0"/>
    <xs:element name="DefaultContinuousPanTiltVelocitySpace" type="xs:anyURI"
minOccurs="0"/>
    <xs:element name="DefaultContinuousZoomVelocitySpace" type="xs:anyURI"
minOccurs="0"/>
    <xs:element name="DefaultPTZSpeed" type="tt:PTZSpeed" minOccurs="0"/>
    <xs:element name="DefaultPTZTimeout" type="xs:duration" minOccurs="0"/>
    <xs:element name="PanTiltLimits" type="tt:PanTiltLimits" minOccurs="0"/>
    <xs:element name="ZoomLimits" type="tt:ZoomLimits" minOccurs="0"/>
    <xs:element name="Extension" type="tt:PTZConfigurationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:extension>
</xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="PTZConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PTZConfigurationOptions">
  <xs:sequence>
    <xs:element name="Spaces" type="tt:PTZSpaces"/>
    <xs:element name="PTZTimeout" type="tt:DurationRange"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PanTiltLimits">
  <xs:sequence>
    <xs:element name="Range" type="tt:Space2DDescription"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ZoomLimits">
  <xs:sequence>
    <xs:element name="Range" type="tt:Space1DDescription"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PTZSpaces">
  <xs:sequence>
    <xs:element name="AbsolutePanTiltPositionSpace" type="tt:Space2DDescription"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="AbsoluteZoomPositionSpace" type="tt:Space1DDescription"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="RelativePanTiltTranslationSpace" type="tt:Space2DDescription"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="RelativeZoomTranslationSpace" type="tt:Space1DDescription"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="ContinuousPanTiltVelocitySpace" type="tt:Space2DDescription"

```

```

minOccurs="0" maxOccurs="unbounded"/>
  <xs:element name="ContinuousZoomVelocitySpace" type="tt:Space1DDescription"
minOccurs="0" maxOccurs="unbounded"/>
  <xs:element name="PanTiltSpeedSpace" type="tt:Space1DDescription" minOccurs="0"
maxOccurs="unbounded"/>
  <xs:element name="ZoomSpeedSpace" type="tt:Space1DDescription" minOccurs="0"
maxOccurs="unbounded"/>
  <xs:element name="Extension" type="tt:PTZSpacesExtension" minOccurs="0"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PTZSpacesExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Space2DDescription">
  <xs:sequence>
    <xs:element name="URI" type="xs:anyURI"/>
    <xs:element name="XRange" type="tt:FloatRange"/>
    <xs:element name="YRange" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Space1DDescription">
  <xs:sequence>
    <xs:element name="URI" type="xs:anyURI"/>
    <xs:element name="XRange" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Vector2D">
  <xs:attribute name="x" type="xs:float" use="required"/>
  <xs:attribute name="y" type="xs:float" use="required"/>
  <xs:attribute name="space" type="xs:anyURI" use="optional"/>
</xs:complexType><!--=====-->
<xs:complexType name="Vector1D">
  <xs:attribute name="x" type="xs:float" use="required"/>
  <xs:attribute name="space" type="xs:anyURI" use="optional"/>
</xs:complexType><!--=====-->
<xs:complexType name="PTZVector">
  <xs:sequence>
    <xs:element name="PanTilt" type="tt:Vector2D" minOccurs="0"/>
    <xs:element name="Zoom" type="tt:Vector1D" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PTZSpeed">
  <xs:sequence>
    <xs:element name="PanTilt" type="tt:Vector2D" minOccurs="0"/>
    <xs:element name="Zoom" type="tt:Vector1D" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PTZStatus">
  <xs:sequence>
    <xs:element name="Position" type="tt:PTZVector" minOccurs="0"/>
    <xs:element name="MoveStatus" type="tt:PTZMoveStatus" minOccurs="0"/>
    <xs:element name="Error" type="xs:string" minOccurs="0"/>
    <xs:element name="UtcTime" type="xs:dateTime"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->

```

```

<xs:complexType name="PTZPreset">
  <xs:sequence>
    <xs:element name="Name" type="tt:Name" minOccurs="0"/>
    <xs:element name="PTZPosition" type="tt:PTZVector" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="token" type="tt:ReferenceToken"/>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PTZMoveStatus">
  <xs:sequence>
    <xs:element name="PanTilt" type="tt:MoveStatus" minOccurs="0"/>
    <xs:element name="Zoom" type="tt:MoveStatus" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="AuxiliaryData">
  <xs:restriction base="xs:string">
    <xs:maxLength value="128"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="MoveStatus">
  <xs:restriction base="xs:string">
    <xs:enumeration value="IDLE"/>
    <xs:enumeration value="MOVING"/>
    <xs:enumeration value="UNKNOWN"/>
  </xs:restriction>
</xs:simpleType>
<!--=====-->
<!-- End, PTZ Related Types -->
<!--=====-->
<!-- Imaging Related Types -->
<!--=====-->
<xs:complexType name="ImagingStatus">
  <xs:sequence>
    <xs:element name="FocusStatus" type="tt:FocusStatus"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="FocusStatus">
  <xs:sequence>
    <xs:element name="Position" type="xs:float"/>
    <xs:element name="MoveStatus" type="tt:MoveStatus"/>
    <xs:element name="Error" type="xs:string"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="FocusConfiguration">
  <xs:sequence>
    <xs:element name="AutoFocusMode" type="tt:AutoFocusMode"/>
    <xs:element name="DefaultSpeed" type="xs:float"/>
    <xs:element name="NearLimit" type="xs:float"/>
    <xs:element name="FarLimit" type="xs:float"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="AutoFocusMode">

```

```

<xs:restriction base="xs:string">
  <xs:enumeration value="AUTO"/>
  <xs:enumeration value="MANUAL"/>
</xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="ImagingSettings">
  <xs:sequence>
    <xs:element name="BacklightCompensation" type="tt:BacklightCompensation"
minOccurs="0"/>
    <xs:element name="Brightness" type="xs:float" minOccurs="0"/>
    <xs:element name="ColorSaturation" type="xs:float" minOccurs="0"/>
    <xs:element name="Contrast" type="xs:float" minOccurs="0"/>
    <xs:element name="Exposure" type="tt:Exposure" minOccurs="0"/>
    <xs:element name="Focus" type="tt:FocusConfiguration" minOccurs="0"/>
    <xs:element name="IrCutFilter" type="tt:IrCutFilterMode" minOccurs="0"/>
    <xs:element name="Sharpness" type="xs:float" minOccurs="0"/>
    <xs:element name="WideDynamicRange" type="tt:WideDynamicRange"
minOccurs="0"/>
    <xs:element name="WhiteBalance" type="tt:WhiteBalance" minOccurs="0"/>
    <xs:element name="Extension" type="tt:ImagingSettingsExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ImagingSettingsExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Exposure">
  <xs:sequence>
    <xs:element name="Mode" type="tt:ExposureMode"/>
    <xs:element name="Priority" type="tt:ExposurePriority"/>
    <xs:element name="Window" type="tt:Rectangle"/>
    <xs:element name="MinExposureTime" type="xs:float"/>
    <xs:element name="MaxExposureTime" type="xs:float"/>
    <xs:element name="MinGain" type="xs:float"/>
    <xs:element name="MaxGain" type="xs:float"/>
    <xs:element name="MinIris" type="xs:float"/>
    <xs:element name="MaxIris" type="xs:float"/>
    <xs:element name="ExposureTime" type="xs:float"/>
    <xs:element name="Gain" type="xs:float"/>
    <xs:element name="Iris" type="xs:float"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="WideDynamicMode">
  <xs:restriction base="xs:string">
    <xs:enumeration value="OFF"/>
    <xs:enumeration value="ON"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="WideDynamicRange">
  <xs:sequence>
    <xs:element name="Mode" type="tt:WideDynamicMode"/>
    <xs:element name="Level" type="xs:float"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="BacklightCompensationMode">
  <xs:restriction base="xs:string">
    <xs:enumeration value="OFF"/>
    <xs:enumeration value="ON"/>
  </xs:restriction>

```



```

</xs:simpleType><!--=====-->
<xs:complexType name="BacklightCompensation">
  <xs:sequence>
    <xs:element name="Mode" type="tt:BacklightCompensationMode"/>
    <xs:element name="Level" type="xs:float"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="ExposurePriority">
  <xs:restriction base="xs:string">
    <xs:enumeration value="LowNoise"/>
    <xs:enumeration value="FrameRate"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="ImagingOptions">
  <xs:sequence>
    <xs:element name="BacklightCompensation" type="tt:BacklightCompensationOptions"/>
    <xs:element name="Brightness" type="tt:FloatRange"/>
    <xs:element name="ColorSaturation" type="tt:FloatRange"/>
    <xs:element name="Contrast" type="tt:FloatRange"/>
    <xs:element name="Exposure" type="tt:ExposureOptions"/>
    <xs:element name="Focus" type="tt:FocusOptions"/>
    <xs:element name="IrCutFilterModes" type="tt:IrCutFilterMode"
maxOccurs="unbounded"/>
    <xs:element name="Sharpness" type="tt:FloatRange"/>
    <xs:element name="WideDynamicRange" type="tt:WideDynamicRangeOptions"/>
    <xs:element name="WhiteBalance" type="tt:WhiteBalanceOptions"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="WideDynamicRangeOptions">
  <xs:sequence>
    <xs:element name="Mode" type="tt:WideDynamicMode" maxOccurs="unbounded"/>
    <xs:element name="Level" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="BacklightCompensationOptions">
  <xs:sequence>
    <xs:element name="Mode" type="tt:WideDynamicMode" maxOccurs="unbounded"/>
    <xs:element name="Level" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FocusOptions">
  <xs:sequence>
    <xs:element name="AutoFocusModes" type="tt:AutoFocusMode" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="DefaultSpeed" type="tt:FloatRange"/>
    <xs:element name="NearLimit" type="tt:FloatRange"/>
    <xs:element name="FarLimit" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ExposureOptions">
  <xs:sequence>
    <xs:element name="Mode" type="tt:ExposureMode" maxOccurs="unbounded"/>
    <xs:element name="Priority" type="tt:ExposurePriority" maxOccurs="unbounded"/>
    <xs:element name="MinExposureTime" type="tt:FloatRange"/>
    <xs:element name="MaxExposureTime" type="tt:FloatRange"/>
    <xs:element name="MinGain" type="tt:FloatRange"/>
    <xs:element name="MaxGain" type="tt:FloatRange"/>
    <xs:element name="MinIris" type="tt:FloatRange"/>
    <xs:element name="MaxIris" type="tt:FloatRange"/>
  </xs:sequence>

```

```

    <xs:element name="ExposureTime" type="tt:FloatRange"/>
    <xs:element name="Gain" type="tt:FloatRange"/>
    <xs:element name="Iris" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="WhiteBalanceOptions">
  <xs:sequence>
    <xs:element name="Mode" type="tt:WhiteBalanceMode" maxOccurs="unbounded"/>
    <xs:element name="YrGain" type="tt:FloatRange"/>
    <xs:element name="YbGain" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FocusMove">
  <xs:sequence>
    <xs:element name="Absolute" type="tt:AbsoluteFocus" minOccurs="0"/>
    <xs:element name="Relative" type="tt:RelativeFocus" minOccurs="0"/>
    <xs:element name="Continuous" type="tt:ContinuousFocus" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="AbsoluteFocus">
  <xs:sequence>
    <xs:element name="Position" type="xs:float"/>
    <xs:element name="Speed" type="xs:float" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="RelativeFocus">
  <xs:sequence>
    <xs:element name="Distance" type="xs:float"/>
    <xs:element name="Speed" type="xs:float" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ContinuousFocus">
  <xs:sequence>
    <xs:element name="Speed" type="xs:float"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="MoveOptions">
  <xs:sequence>
    <xs:element name="Absolute" type="tt:AbsoluteFocusOptions" minOccurs="0"/>
    <xs:element name="Relative" type="tt:RelativeFocusOptions" minOccurs="0"/>
    <xs:element name="Continuous" type="tt:ContinuousFocusOptions" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="AbsoluteFocusOptions">
  <xs:sequence>
    <xs:element name="Position" type="tt:FloatRange"/>
    <xs:element name="Speed" type="tt:FloatRange" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="RelativeFocusOptions">
  <xs:sequence>
    <xs:element name="Distance" type="tt:FloatRange"/>
    <xs:element name="Speed" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ContinuousFocusOptions">
  <xs:sequence>
    <xs:element name="Speed" type="tt:FloatRange"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="ExposureMode">
  <xs:restriction base="xs:string">

```



```

        <xs:enumeration value="AUTO"/>
        <xs:enumeration value="MANUAL"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="Enabled">
    <xs:restriction base="xs:string">
        <xs:enumeration value="ENABLED"/>
        <xs:enumeration value="DISABLED"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="WhiteBalanceMode">
    <xs:restriction base="xs:string">
        <xs:enumeration value="AUTO"/>
        <xs:enumeration value="MANUAL"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="IrCutFilterMode">
    <xs:restriction base="xs:string">
        <xs:enumeration value="ON"/>
        <xs:enumeration value="OFF"/>
        <xs:enumeration value="AUTO"/>
    </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="WhiteBalance">
    <xs:sequence>
        <xs:element name="Mode" type="tt:WhiteBalanceMode"/>
        <xs:element name="CrGain" type="xs:float"/>
        <xs:element name="CbGain" type="xs:float"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType>
<!--=====-->
<!-- End, Imaging Related Types -->
<!--=====-->
<!--=====-->
<!-- Imaging Version 2.0 Related Types -->
<!--=====-->
<xs:complexType name="ImagingStatus20">
    <xs:sequence>
        <xs:element name="FocusStatus20" type="tt:FocusStatus20" minOccurs="0"/>
        <xs:element name="Extension" type="tt:ImagingStatus20Extension" minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ImagingStatus20Extension">
    <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FocusStatus20">
    <xs:sequence>
        <xs:element name="Position" type="xs:float"/>
        <xs:element name="MoveStatus" type="tt:MoveStatus"/>
        <xs:element name="Error" type="xs:string" minOccurs="0"/>
        <xs:element name="Extension" type="tt:FocusStatus20Extension" minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="FocusStatus20Extension">

```

```

<xs:sequence>
  <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ImagingSettings20">
  <xs:sequence>
    <xs:element name="BacklightCompensation" type="tt:BacklightCompensation20"
minOccurs="0"/>
    <xs:element name="Brightness" type="xs:float" minOccurs="0"/>
    <xs:element name="ColorSaturation" type="xs:float" minOccurs="0"/>
    <xs:element name="Contrast" type="xs:float" minOccurs="0"/>
    <xs:element name="Exposure" type="tt:Exposure20" minOccurs="0"/>
    <xs:element name="Focus" type="tt:FocusConfiguration20" minOccurs="0"/>
    <xs:element name="IrCutFilter" type="tt:IrCutFilterMode" minOccurs="0"/>
    <xs:element name="Sharpness" type="xs:float" minOccurs="0"/>
    <xs:element name="WideDynamicRange" type="tt:WideDynamicRange20"
minOccurs="0"/>
    <xs:element name="WhiteBalance" type="tt:WhiteBalance20" minOccurs="0"/>
    <xs:element name="Extension" type="tt:ImagingSettingsExtension20" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ImagingSettingsExtension20">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="WideDynamicRange20">
  <xs:sequence>
    <xs:element name="Mode" type="tt:WideDynamicMode"/>
    <xs:element name="Level" type="xs:float" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="BacklightCompensation20">
  <xs:sequence>
    <xs:element name="Mode" type="tt:BacklightCompensationMode"/>
    <xs:element name="Level" type="xs:float" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Exposure20">
  <xs:sequence>
    <xs:element name="Mode" type="tt:ExposureMode"/>
    <xs:element name="Priority" type="tt:ExposurePriority" minOccurs="0"/>
    <xs:element name="Window" type="tt:Rectangle" minOccurs="0"/>
    <xs:element name="MinExposureTime" type="xs:float" minOccurs="0"/>
    <xs:element name="MaxExposureTime" type="xs:float" minOccurs="0"/>
    <xs:element name="MinGain" type="xs:float" minOccurs="0"/>
    <xs:element name="MaxGain" type="xs:float" minOccurs="0"/>
    <xs:element name="MinIris" type="xs:float" minOccurs="0"/>
    <xs:element name="MaxIris" type="xs:float" minOccurs="0"/>
    <xs:element name="ExposureTime" type="xs:float" minOccurs="0"/>
    <xs:element name="Gain" type="xs:float" minOccurs="0"/>
    <xs:element name="Iris" type="xs:float" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ImagingOptions20">
  <xs:sequence>
    <xs:element name="BacklightCompensation"
type="tt:BacklightCompensationOptions20" minOccurs="0"/>
    <xs:element name="Brightness" type="tt:FloatRange" minOccurs="0"/>

```

```

<xs:element name="ColorSaturation" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="Contrast" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="Exposure" type="tt:ExposureOptions20" minOccurs="0"/>
<xs:element name="Focus" type="tt:FocusOptions20" minOccurs="0"/>
<xs:element name="IrCutFilterModes" type="tt:IrCutFilterMode" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="Sharpness" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="WideDynamicRange" type="tt:WideDynamicRangeOptions20"
minOccurs="0"/>
<xs:element name="WhiteBalance" type="tt:WhiteBalanceOptions20" minOccurs="0"/>
<xs:element name="Extension" type="tt:ImagingOptions20Extension" minOccurs="0"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ImagingOptions20Extension">
<xs:sequence>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="WideDynamicRangeOptions20">
<xs:sequence>
<xs:element name="Mode" type="tt:WideDynamicMode" maxOccurs="unbounded"/>
<xs:element name="Level" type="tt:FloatRange" minOccurs="0"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="BacklightCompensationOptions20">
<xs:sequence>
<xs:element name="Mode" type="tt:BacklightCompensationMode"
maxOccurs="unbounded"/>
<xs:element name="Level" type="tt:FloatRange" minOccurs="0"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ExposureOptions20">
<xs:sequence>
<xs:element name="Mode" type="tt:ExposureMode" maxOccurs="unbounded"/>
<xs:element name="Priority" type="tt:ExposurePriority" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="MinExposureTime" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="MaxExposureTime" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="MinGain" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="MaxGain" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="MinIris" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="MaxIris" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="ExposureTime" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="Gain" type="tt:FloatRange" minOccurs="0"/>
<xs:element name="Iris" type="tt:FloatRange" minOccurs="0"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="MoveOptions20">
<xs:sequence>
<xs:element name="Absolute" type="tt:AbsoluteFocusOptions" minOccurs="0"/>
<xs:element name="Relative" type="tt:RelativeFocusOptions20" minOccurs="0"/>
<xs:element name="Continuous" type="tt:ContinuousFocusOptions" minOccurs="0"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="RelativeFocusOptions20">
<xs:sequence>
<xs:element name="Distance" type="tt:FloatRange"/>
<xs:element name="Speed" type="tt:FloatRange" minOccurs="0"/>
</xs:sequence>
</xs:complexType><!--=====-->

```

```

<xs:complexType name="WhiteBalance20">
  <xs:sequence>
    <xs:element name="Mode" type="tt:WhiteBalanceMode"/>
    <xs:element name="CrGain" type="xs:float" minOccurs="0"/>
    <xs:element name="CbGain" type="xs:float" minOccurs="0"/>
    <xs:element name="Extension" type="tt:WhiteBalance20Extension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="WhiteBalance20Extension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FocusConfiguration20">
  <xs:sequence>
    <xs:element name="AutoFocusMode" type="tt:AutoFocusMode"/>
    <xs:element name="DefaultSpeed" type="xs:float" minOccurs="0"/>
    <xs:element name="NearLimit" type="xs:float" minOccurs="0"/>
    <xs:element name="FarLimit" type="xs:float" minOccurs="0"/>
    <xs:element name="Extension" type="tt:FocusConfiguration20Extension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="FocusConfiguration20Extension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="WhiteBalanceOptions20">
  <xs:sequence>
    <xs:element name="Mode" type="tt:WhiteBalanceMode" maxOccurs="unbounded"/>
    <xs:element name="YrGain" type="tt:FloatRange" minOccurs="0"/>
    <xs:element name="YbGain" type="tt:FloatRange" minOccurs="0"/>
    <xs:element name="Extension" type="tt:WhiteBalanceOptions20Extension"
minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="WhiteBalanceOptions20Extension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FocusOptions20">
  <xs:sequence>
    <xs:element name="AutoFocusModes" type="tt:AutoFocusMode" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="DefaultSpeed" type="tt:FloatRange" minOccurs="0"/>
    <xs:element name="NearLimit" type="tt:FloatRange" minOccurs="0"/>
    <xs:element name="FarLimit" type="tt:FloatRange" minOccurs="0"/>
    <xs:element name="Extension" type="tt:FocusOptions20Extension" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FocusOptions20Extension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>

```

```

</xs:complexType>
<!--=====-->
<!-- End, Imaging Version 2.0 Related Types -->
<!--=====-->
<!--=====-->
<!-- Event and Analytics Types -->
<!--=====-->
<xs:simpleType name="TopicNamespaceLocation">
  <xs:restriction base="xs:anyURI"/>
</xs:simpleType><!--=====-->
<xs:simpleType name="PropertyOperation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Initialized"/>
    <xs:enumeration value="Deleted"/>
    <xs:enumeration value="Changed"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:element name="Message">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Source" type="tt:ItemList" minOccurs="0"/>
      <xs:element name="Key" type="tt:ItemList" minOccurs="0"/>
      <xs:element name="Data" type="tt:ItemList" minOccurs="0"/>
      <xs:element name="Extension" type="tt:MessageExtension" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="UtcTime" type="xs:dateTime" use="required"/>
    <xs:attribute name="PropertyOperation" type="tt:PropertyOperation"/>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>
</xs:element><!--=====-->
<xs:complexType name="MessageExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ItemList">
  <xs:sequence>
    <xs:element name="SimpleItem" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Value" type="xs:anySimpleType" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="ElementItem" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##any"/>
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="Extension" type="tt:ItemListExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ItemListExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

```



```

<!--=====-->
<!-- Message Description -->
<!--=====-->
<xs:complexType name="MessageDescription">
  <xs:sequence>
    <xs:element name="Source" type="tt:ItemListDescription" minOccurs="0"/>
    <xs:element name="Key" type="tt:ItemListDescription" minOccurs="0"/>
    <xs:element name="Data" type="tt:ItemListDescription" minOccurs="0"/>
    <xs:element name="Extension" type="tt:MessageDescriptionExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="IsProperty" type="xs:boolean"/>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="MessageDescriptionExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ItemListDescription">
  <xs:sequence>
    <xs:element name="SimpleItemDescription" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="ElementItemDescription" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="Extension" type="tt:ItemListDescriptionExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ItemListDescriptionExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Vector">
  <xs:attribute name="x" type="xs:float"/>
  <xs:attribute name="y" type="xs:float"/>
</xs:complexType><!--=====-->
<xs:complexType name="Rectangle">
  <xs:attribute name="bottom" type="xs:float"/>
  <xs:attribute name="top" type="xs:float"/>
  <xs:attribute name="right" type="xs:float"/>
  <xs:attribute name="left" type="xs:float"/>
</xs:complexType><!--=====-->
<xs:complexType name="Polygon">
  <xs:sequence>
    <xs:element name="Point" type="tt:Vector" minOccurs="3" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="Polygon" type="tt:Polygon"/><!--
=====-->
<xs:complexType name="Polyline">

```

```

<xs:sequence>
  <xs:element name="Point" type="tt:Vector" minOccurs="2" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<xs:element name="Polyline" type="tt:Polyline"/><!--
=====>
<xs:simpleType name="Direction">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Left"/>
    <xs:enumeration value="Right"/>
    <xs:enumeration value="Any"/>
  </xs:restriction>
</xs:simpleType><!--=====>
<xs:complexType name="Color">
  <xs:attribute name="X" type="xs:float" use="required"/>
  <xs:attribute name="Y" type="xs:float" use="required"/>
  <xs:attribute name="Z" type="xs:float" use="required"/>
  <xs:attribute name="Colorspace" type="xs:anyURI"/>
</xs:complexType><!--=====>
<xs:complexType name="ColorCovariance">
  <xs:attribute name="XX" type="xs:float" use="required"/>
  <xs:attribute name="YY" type="xs:float" use="required"/>
  <xs:attribute name="ZZ" type="xs:float" use="required"/>
  <xs:attribute name="XY" type="xs:float"/>
  <xs:attribute name="XZ" type="xs:float"/>
  <xs:attribute name="YZ" type="xs:float"/>
  <xs:attribute name="Colorspace" type="xs:anyURI"/>
</xs:complexType><!--=====><!-- Scene
Description --><!--=====>
<xs:complexType name="Appearance">
  <xs:sequence>
    <xs:element name="Transformation" type="tt:Transformation" minOccurs="0"/>
    <xs:element name="Shape" type="tt:ShapeDescriptor" minOccurs="0"/>
    <xs:element name="Color" type="tt:ColorDescriptor" minOccurs="0"/>
    <xs:element name="Class" type="tt:ClassDescriptor" minOccurs="0"/>
    <xs:element name="Extension" type="tt:AppearanceExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====>
<xs:complexType name="AppearanceExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====>
<xs:complexType name="ShapeDescriptor">
  <xs:sequence>
    <xs:element name="BoundingBox" type="tt:Rectangle"/>
    <xs:element name="CenterOfGravity" type="tt:Vector"/>
    <xs:element name="Polygon" type="tt:Polygon" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:ShapeDescriptorExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====>
<xs:complexType name="ShapeDescriptorExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====>
<xs:complexType name="ColorDescriptor">

```



```

<xs:sequence>
  <xs:element name="ColorCluster" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Color" type="tt:Color"/>
        <xs:element name="Weight" type="xs:float" minOccurs="0"/>
        <xs:element name="Covariance" type="tt:ColorCovariance" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Extension" type="tt:ColorDescriptorExtension" minOccurs="0"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ColorDescriptorExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="ClassType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Animal"/>
    <xs:enumeration value="Face"/>
    <xs:enumeration value="Human"/>
    <xs:enumeration value="Vehical"/>
    <xs:enumeration value="Other"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="ClassDescriptor">
  <xs:sequence>
    <xs:element name="ClassCandidate" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Type" type="tt:ClassType"/>
          <xs:element name="Likelihood" type="xs:float"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Extension" type="tt:ClassDescriptorExtension" minOccurs="0"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ClassDescriptorExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Object">
  <xs:complexContent>
    <xs:extension base="tt:ObjectId">
      <xs:sequence>
        <xs:element name="Appearance" type="tt:Appearance" minOccurs="0"/>
        <xs:element name="Behaviour" type="tt:Behaviour" minOccurs="0"/>
        <xs:element name="Extension" type="tt:ObjectExtension" minOccurs="0"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="ObjectExtension">
  <xs:sequence>

```

```

    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Transformation">
  <xs:sequence>
    <xs:element name="Translate" type="tt:Vector" minOccurs="0"/>
    <xs:element name="Scale" type="tt:Vector" minOccurs="0"/>
    <xs:element name="Extension" type="tt:TransformationExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="TransformationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Frame">
  <xs:sequence>
    <xs:element name="PTZStatus" type="tt:PTZStatus" minOccurs="0"/>
    <xs:element name="Transformation" type="tt:Transformation" minOccurs="0"/>
    <xs:element name="Object" type="tt:Object" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="ObjectTree" type="tt:ObjectTree" minOccurs="0"/>
    <xs:element name="Extension" type="tt:FrameExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="UtcTime" type="xs:dateTime" use="required"/>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="FrameExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Merge">
  <xs:sequence>
    <xs:element name="from" type="tt:ObjectId" minOccurs="2" maxOccurs="unbounded"/>
    <xs:element name="to" type="tt:ObjectId"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Split">
  <xs:sequence>
    <xs:element name="from" type="tt:ObjectId"/>
    <xs:element name="to" type="tt:ObjectId" minOccurs="2" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Rename">
  <xs:sequence>
    <xs:element name="from" type="tt:ObjectId"/>
    <xs:element name="to" type="tt:ObjectId"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ObjectId">
  <xs:attribute name="ObjectId" type="xs:integer"/>
</xs:complexType><!--=====-->
<xs:complexType name="Behaviour">
  <xs:sequence>
    <xs:element name="Removed" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>

```

```

        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Idle" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Extension" type="tt:BehaviourExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="BehaviourExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="ObjectTree">
  <xs:sequence>
    <xs:element name="Rename" type="tt:Rename" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Split" type="tt:Split" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Merge" type="tt:Merge" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Delete" type="tt:ObjectId" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:ObjectTreeExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ObjectTreeExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====--><!-- Analytics
Configuration --><!--=====-->
<xs:complexType name="AnalyticsEngineConfiguration">
  <xs:sequence>
    <xs:element name="AnalyticsModule" type="tt:Config" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:AnalyticsEngineConfigurationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsEngineConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="RuleEngineConfiguration">
  <xs:sequence>
    <xs:element name="Rule" type="tt:Config" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:RuleEngineConfigurationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->

```

```

<xs:complexType name="RuleEngineConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="Config">
  <xs:sequence>
    <xs:element name="Parameters" type="tt:ItemList"/>
  </xs:sequence>
  <xs:attribute name="Name" type="xs:string" use="required"/>
  <xs:attribute name="Type" type="xs:QName" use="required"/>
</xs:complexType><!--=====-->
<xs:complexType name="ConfigDescription">
  <xs:sequence>
    <xs:element name="Parameters" type="tt:ItemListDescription"/>
    <xs:element name="Messages" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="tt:MessageDescription">
            <xs:sequence>
              <xs:element name="ParentTopic" type="xs:string"/>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
    <xs:element name="Extension" type="tt:ConfigDescriptionExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="Name" type="xs:QName" use="required"/>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ConfigDescriptionExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="SupportedRules">
  <xs:sequence>
    <xs:element name="RuleContentSchemaLocation" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="RuleDescription" type="tt:ConfigDescription" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:SupportedRulesExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SupportedRulesExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="SupportedAnalyticsModules">
  <xs:sequence>
    <xs:element name="AnalyticsModuleContentSchemaLocation" type="xs:anyURI"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="AnalyticsModuleDescription" type="tt:ConfigDescription"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:SupportedAnalyticsModulesExtension"
minOccurs="0"/>

```

```

</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SupportedAnalyticsModulesExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!--=====-->
<!--End, Event and Analytics Types -->
<!--=====-->
<!--=====-->
<!-- Metadata Streaming Types -->
<!--=====-->
<xs:complexType name="MetadataStream">
  <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="VideoAnalytics" type="tt:VideoAnalyticsStream"/>
      <xs:element name="PTZ" type="tt:PTZStream"/>
      <xs:element name="Event" type="tt:EventStream"/>
      <xs:element name="Extension" type="tt:MetadataStreamExtension"/>
    </xs:choice>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="MetadataStreamExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:element name="MetadataStream" type="tt:MetadataStream"/><!--
=====-->
<xs:complexType name="VideoAnalyticsStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="Frame" type="tt:Frame"/>
    <xs:element name="Extension" type="tt:VideoAnalyticsStreamExtension"/>
  </xs:choice>
</xs:complexType><!--=====-->
<xs:complexType name="VideoAnalyticsStreamExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PTZStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="PTZStatus" type="tt:PTZStatus"/>
    <xs:element name="Extension" type="tt:PTZStreamExtension"/>
  </xs:choice>
</xs:complexType><!--=====-->
<xs:complexType name="PTZStreamExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="EventStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="wsnt:NotificationMessage"/>
    <xs:element name="Extension" type="tt:EventStreamExtension"/>
  </xs:choice>
</xs:complexType>

```



```

</xs:choice>
</xs:complexType><!--=====-->
<xs:complexType name="EventStreamExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!--=====-->
<!-- End, Metadata Streaming Types -->
<!--=====-->
<!--=====-->
<!-- Display Related Types -->
<!--=====-->
<xs:complexType name="PaneConfiguration">
  <xs:sequence>
    <xs:element name="PaneName" type="xs:string" minOccurs="0"/>
    <xs:element name="AudioOutputToken" type="tt:ReferenceToken" minOccurs="0"/>
    <xs:element name="AudioSourceToken" type="tt:ReferenceToken" minOccurs="0"/>
    <xs:element name="AudioEncoderConfiguration" type="tt:AudioEncoderConfiguration"
minOccurs="0"/>
    <xs:element name="ReceiverToken" type="tt:ReferenceToken" minOccurs="0"/>
    <xs:element name="Token" type="tt:ReferenceToken"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PaneLayout">
  <xs:sequence>
    <xs:element name="Pane" type="tt:ReferenceToken"/>
    <xs:element name="Area" type="tt:Rectangle"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="Layout">
  <xs:sequence>
    <xs:element name="PaneLayout" type="tt:PaneLayout" maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:LayoutExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="LayoutExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="CodingCapabilities">
  <xs:sequence>
    <xs:element name="AudioEncodingCapabilities"
type="tt:AudioEncoderConfigurationOptions" minOccurs="0"/>
    <xs:element name="AudioDecodingCapabilities"
type="tt:AudioDecoderConfigurationOptions" minOccurs="0"/>
    <xs:element name="VideoDecodingCapabilities"
type="tt:VideoDecoderConfigurationOptions"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>

```

```

</xs:complexType><!--=====-->
<xs:complexType name="LayoutOptions">
  <xs:sequence>
    <xs:element name="PaneLayoutOptions" type="tt:PaneLayoutOptions"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:LayoutOptionsExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="LayoutOptionsExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="PaneLayoutOptions">
  <xs:sequence>
    <xs:element name="Area" type="tt:Rectangle" maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:PaneOptionExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="PaneOptionExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!--=====-->
<!-- End, Display Related Types -->
<!--=====-->
<!-- Receiver Types -->
<!--=====-->
<xs:simpleType name="ReceiverToken">
  <xs:restriction base="tt:ReferenceToken"/>
</xs:simpleType><!--=====-->
<xs:complexType name="Receiver">
  <xs:sequence>
    <xs:element name="Token" type="tt:ReceiverToken"/>
    <xs:element name="Configuration" type="tt:ReceiverConfiguration"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="ReceiverConfiguration">
  <xs:sequence>
    <xs:element name="Mode" type="tt:ReceiverMode"/>
    <xs:element name="MediaUri" type="xs:anyURI"/>
    <xs:element name="StreamSetup" type="tt:StreamSetup"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="ReceiverMode">
  <xs:restriction base="xs:string">
    <xs:enumeration value="AutoConnect"/>
    <xs:enumeration value="AlwaysConnect"/>
    <xs:enumeration value="NeverConnect"/>
    <xs:enumeration value="Unknown"/>
  </xs:restriction>
</xs:simpleType>

```



```

</xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="ReceiverState">
  <xs:restriction base="xs:string">
    <xs:enumeration value="NotConnected"/>
    <xs:enumeration value="Connecting"/>
    <xs:enumeration value="Connected"/>
    <xs:enumeration value="Unknown"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="ReceiverStateInformation">
  <xs:sequence>
    <xs:element name="State" type="tt:ReceiverState"/>
    <xs:element name="AutoCreated" type="xs:boolean"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType>
<!--=====-->
<!-- End, Receiver Types -->
<!--=====-->
<!--=====-->
<!-- Storage Types -->
<!--=====-->
<xs:simpleType name="ReceiverReference">
  <xs:restriction base="tt:ReferenceToken"/>
</xs:simpleType><!--=====-->
<xs:simpleType name="RecordingReference">
  <xs:restriction base="tt:ReferenceToken"/>
</xs:simpleType><!--=====-->
<xs:complexType name="SourceReference">
  <xs:sequence>
    <xs:element name="Token" type="tt:ReferenceToken"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="Type" type="xs:anyURI" use="optional"
default="http://www.onvif.org/ver10/schema/Receiver"/>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="TrackReference">
  <xs:restriction base="tt:ReferenceToken"/>
</xs:simpleType><!--=====-->
<xs:simpleType name="Description">
  <xs:restriction base="xs:string"/>
</xs:simpleType><!--=====-->
<xs:complexType name="DateTimeRange">
  <xs:sequence>
    <xs:element name="From" type="xs:dateTime"/>
    <xs:element name="Until" type="xs:dateTime"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingSummary">
  <xs:sequence>
    <xs:element name="DataFrom" type="xs:dateTime"/>
    <xs:element name="DataUntil" type="xs:dateTime"/>
    <xs:element name="NumberRecordings" type="xs:int"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"

```

```

maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SearchScope">
<xs:sequence>
<xs:element name="IncludedSources" type="tt:SourceReference" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="IncludedRecordings" type="tt:RecordingReference" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="RecordingInformationFilter" type="tt:XPathExpression"
minOccurs="0"/>
<xs:element name="Extension" type="tt:SearchScopeExtension" minOccurs="0"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SearchScopeExtension">
<xs:sequence>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="EventFilter">
<xs:complexContent>
<xs:extension base="wsnt:FilterType">
<xs:anyAttribute processContents="lax"/>
</xs:extension>
</xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="PTZPositionFilter">
<xs:sequence>
<xs:element name="MinPosition" type="tt:PTZVector"/>
<xs:element name="MaxPosition" type="tt:PTZVector"/>
<xs:element name="EnterOrExit" type="xs:boolean"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="MetadataFilter">
<xs:sequence>
<xs:element name="MetadataStreamFilter" type="tt:XPathExpression"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="XPathExpression">
<xs:restriction base="xs:string"/>
</xs:simpleType><!--=====-->
<xs:complexType name="FindRecordingResultList">
<xs:sequence>
<xs:element name="SearchState" type="tt:SearchState"/>
<xs:element name="RecordingInformation" type="tt:RecordingInformation"
minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FindEventResultList">
<xs:sequence>
<xs:element name="SearchState" type="tt:SearchState"/>
<xs:element name="Result" type="tt:FindEventResult" minOccurs="0"
maxOccurs="unbounded"/>

```

```

</xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FindEventResult">
  <xs:sequence>
    <xs:element name="RecordingToken" type="tt:RecordingReference"/>
    <xs:element name="TrackToken" type="tt:TrackReference"/>
    <xs:element name="Time" type="xs:dateTime"/>
    <xs:element name="Event" type="wsnt:NotificationMessageHolderType"/>
    <xs:element name="StartStateEvent" type="xs:boolean"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="FindPTZPositionResultList">
  <xs:sequence>
    <xs:element name="SearchState" type="tt:SearchState"/>
    <xs:element name="Result" type="tt:FindPTZPositionResult" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FindPTZPositionResult">
  <xs:sequence>
    <xs:element name="RecordingToken" type="tt:RecordingReference"/>
    <xs:element name="TrackToken" type="tt:TrackReference"/>
    <xs:element name="Time" type="xs:dateTime"/>
    <xs:element name="Position" type="tt:PTZVector"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="FindMetadataResultList">
  <xs:sequence>
    <xs:element name="SearchState" type="tt:SearchState"/>
    <xs:element name="Result" type="tt:FindMetadataResult" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="FindMetadataResult">
  <xs:sequence>
    <xs:element name="RecordingToken" type="tt:RecordingReference"/>
    <xs:element name="TrackToken" type="tt:TrackReference"/>
    <xs:element name="Time" type="xs:dateTime"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="SearchState">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Queued"/>
    <xs:enumeration value="Searching"/>
    <xs:enumeration value="Completed"/>
    <xs:enumeration value="Unknown"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:simpleType name="JobToken">
  <xs:restriction base="tt:ReferenceToken"/>
</xs:simpleType><!--=====-->
<xs:complexType name="RecordingInformation">
  <xs:sequence>

```

```

<xs:element name="RecordingToken" type="tt:RecordingReference"/>
<xs:element name="Source" type="tt:RecordingSourceInformation"/>
<xs:element name="EarliestRecording" type="xs:dateTime" minOccurs="0"/>
<xs:element name="LatestRecording" type="xs:dateTime" minOccurs="0"/>
<xs:element name="Content" type="tt:Description"/>
<xs:element name="Track" type="tt:TrackInformation" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="RecordingStatus" type="tt:RecordingStatus"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingSourceInformation">
<xs:sequence>
<xs:element name="SourceId" type="xs:anyURI"/>
<xs:element name="Name" type="tt:Name"/>
<xs:element name="Location" type="tt:Description"/>
<xs:element name="Description" type="tt:Description"/>
<xs:element name="Address" type="xs:anyURI"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="RecordingStatus">
<xs:restriction base="xs:string">
<xs:enumeration value="Initiated"/>
<xs:enumeration value="Recording"/>
<xs:enumeration value="Stopped"/>
<xs:enumeration value="Removing"/>
<xs:enumeration value="Removed"/>
<xs:enumeration value="Unknown"/>
</xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="TrackInformation">
<xs:sequence>
<xs:element name="TrackToken" type="tt:TrackReference"/>
<xs:element name="TrackType" type="tt:TrackType"/>
<xs:element name="Description" type="tt:Description"/>
<xs:element name="DataFrom" type="xs:dateTime"/>
<xs:element name="DataTo" type="xs:dateTime"/>
<xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="TrackType">
<xs:restriction base="xs:string">
<xs:enumeration value="Video"/>
<xs:enumeration value="Audio"/>
<xs:enumeration value="Metadata"/>
<xs:enumeration value="Extended"/>
</xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="MediaAttributes">
<xs:sequence>
<xs:element name="RecordingToken" type="tt:RecordingReference"/>
<xs:element name="TrackAttributes" type="tt:TrackAttributes" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="From" type="xs:dateTime"/>
<xs:element name="Until" type="xs:dateTime"/>

```

```

        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="TrackAttributes">
    <xs:sequence>
        <xs:element name="TrackInformation" type="tt:TrackInformation"/>
        <xs:element name="VideoAttributes" type="tt:VideoAttributes" minOccurs="0"/>
        <xs:element name="AudioAttributes" type="tt:AudioAttributes" minOccurs="0"/>
        <xs:element name="MetadataAttributes" type="tt:MetadataAttributes" minOccurs="0"/>
        <xs:element name="Extension" type="tt:TrackAttributesExtension" minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="TrackAttributesExtension">
    <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="VideoAttributes">
    <xs:sequence>
        <xs:element name="Bitrate" type="xs:int" minOccurs="0"/>
        <xs:element name="Width" type="xs:int"/>
        <xs:element name="Height" type="xs:int"/>
        <xs:element name="Encoding" type="tt:VideoEncoding"/>
        <xs:element name="Framerate" type="xs:float"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AudioAttributes">
    <xs:sequence>
        <xs:element name="Bitrate" type="xs:int" minOccurs="0"/>
        <xs:element name="Encoding" type="tt:AudioEncoding"/>
        <xs:element name="Samplerate" type="xs:int"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="MetadataAttributes">
    <xs:sequence>
        <xs:element name="CanContainPTZ" type="xs:boolean"/>
        <xs:element name="CanContainAnalytics" type="xs:boolean"/>
        <xs:element name="CanContainNotifications" type="xs:boolean"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType>
<!--=====-->
<!-- RecordingService Types -->
<!--=====-->
<xs:simpleType name="RecordingJobReference">
    <xs:restriction base="tt:ReferenceToken"/>
</xs:simpleType><!--=====-->
<xs:complexType name="RecordingConfiguration">
    <xs:sequence>
        <xs:element name="Source" type="tt:RecordingSourceInformation"/>

```



```

        <xs:element name="Content" type="tt:Description"/>
        <xs:element name="MaximumRetentionTime" type="xs:duration"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="TrackConfiguration">
    <xs:sequence>
        <xs:element name="TrackType" type="tt:TrackType"/>
        <xs:element name="Description" type="tt:Description"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="GetRecordingsResponseItem">
    <xs:sequence>
        <xs:element name="RecordingToken" type="tt:RecordingReference"/>
        <xs:element name="Configuration" type="tt:RecordingConfiguration"/>
        <xs:element name="Tracks" type="tt:GetTracksResponseList"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="GetTracksResponseList">
    <xs:sequence>
        <xs:element name="Track" type="tt:GetTracksResponseItem" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="GetTracksResponseItem">
    <xs:sequence>
        <xs:element name="TrackToken" type="tt:TrackReference"/>
        <xs:element name="Configuration" type="tt:TrackConfiguration"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobConfiguration">
    <xs:sequence>
        <xs:element name="RecordingToken" type="tt:RecordingReference"/>
        <xs:element name="Mode" type="tt:RecordingJobMode"/>
        <xs:element name="Priority" type="xs:int"/>
        <xs:element name="Source" type="tt:RecordingJobSource" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="Extension" type="tt:RecordingJobConfigurationExtension"
minOccurs="0"/>
    </xs:sequence>
    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:simpleType name="RecordingJobMode">
    <xs:restriction base="xs:string"/>
</xs:simpleType><!--=====-->
<xs:complexType name="RecordingJobConfigurationExtension">
    <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>

```

```

</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobSource">
  <xs:sequence>
    <xs:element name="SourceToken" type="tt:SourceReference" minOccurs="0"/>
    <xs:element name="AutoCreateReceiver" type="xs:boolean" minOccurs="0"/>
    <xs:element name="Tracks" type="tt:RecordingJobTrack" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:RecordingJobSourceExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobSourceExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobTrack">
  <xs:sequence>
    <xs:element name="SourceTag" type="xs:string"/>
    <xs:element name="Destination" type="tt:TrackReference"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobStateInformation">
  <xs:sequence>
    <xs:element name="RecordingToken" type="tt:RecordingReference"/>
    <xs:element name="State" type="tt:RecordingJobState"/>
    <xs:element name="Sources" type="tt:RecordingJobStateSource" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:RecordingJobStateInformationExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobStateInformationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:simpleType name="RecordingJobState">
  <xs:restriction base="xs:string"/>
</xs:simpleType><!--=====-->
<xs:complexType name="RecordingJobStateSource">
  <xs:sequence>
    <xs:element name="SourceToken" type="tt:SourceReference"/>
    <xs:element name="State" type="tt:RecordingJobState"/>
    <xs:element name="Tracks" type="tt:RecordingJobStateTracks"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobStateTracks">
  <xs:sequence>
    <xs:element name="Track" type="tt:RecordingJobStateTrack" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>

```



```

    <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="RecordingJobStateTrack">
  <xs:sequence>
    <xs:element name="SourceTag" type="xs:string"/>
    <xs:element name="Destination" type="tt:TrackReference"/>
    <xs:element name="Error" type="xs:string" minOccurs="0"/>
    <xs:element name="State" type="tt:RecordingJobState"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="GetRecordingJobsResponseItem">
  <xs:sequence>
    <xs:element name="JobToken" type="tt:RecordingJobReference"/>
    <xs:element name="JobConfiguration" type="tt:RecordingJobConfiguration"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType>
<!--=====-->
<!--      End, RecordingService Types      -->
<!--=====-->
<!--=====-->
<!--      Replay Types      -->
<!--=====-->
<xs:complexType name="ReplayConfiguration">
  <xs:sequence>
    <xs:element name="SessionTimeout" type="xs:duration"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType>
<!--=====-->
<!--      End, Replay Types      -->
<!--=====-->
<!--=====-->
<!--      Analytics Device Types      -->
<!--=====-->
<xs:complexType name="AnalyticsEngine">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="name" type="xs:string"/>
        <xs:element name="AnalyticsEngineConfiguration"
type="tt:AnalyticsDeviceEngineConfiguration"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsDeviceEngineConfiguration">
  <xs:sequence>
    <xs:element name="EngineConfiguration" type="tt:EngineConfiguration"
maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:AnalyticsDeviceEngineConfigurationExtension"
minOccurs="0"/>
  </xs:sequence>

```

```

</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsDeviceEngineConfigurationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="EngineConfiguration">
  <xs:sequence>
    <xs:element name="VideoAnalyticsConfiguration"
type="tt:VideoAnalyticsConfiguration"/>
    <xs:element name="AnalyticsEngineInputInfo" type="tt:AnalyticsEngineInputInfo"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsEngineInputInfo">
  <xs:sequence>
    <xs:element name="InputInfo" type="tt:Config" minOccurs="0"/>
    <xs:element name="Extension" type="tt:AnalyticsEngineInputInfoExtension"
minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsEngineInputInfoExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsEngineInput">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="SourceIdentification" type="tt:SourceIdentification"/>
        <xs:element name="VideoInput" type="tt:VideoEncoderConfiguration"/>
        <xs:element name="MetadataInput" type="tt:MetadataInput"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:complexType name="SourceIdentification">
  <xs:sequence>
    <xs:element name="Name" type="xs:string"/>
    <xs:element name="Token" type="tt:ReferenceToken" maxOccurs="unbounded"/>
    <xs:element name="Extension" type="tt:SourceIdentificationExtension" minOccurs="0"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="SourceIdentificationExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="MetadataInput">

```

```

<xs:sequence>
  <xs:element name="MetadataConfig" type="tt:Config" minOccurs="0"
maxOccurs="unbounded"/>
  <xs:element name="Extension" type="tt:MetadataInputExtension" minOccurs="0"/>
</xs:sequence>
<xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="MetadataInputExtension">
  <xs:sequence>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsEngineControl">
  <xs:complexContent>
    <xs:extension base="tt:ConfigurationEntity">
      <xs:sequence>
        <xs:element name="EngineToken" type="tt:ReferenceToken"/>
        <xs:element name="EngineConfigToken" type="tt:ReferenceToken"/>
        <xs:element name="InputToken" type="tt:ReferenceToken"/>
maxOccurs="unbounded"/>
        <xs:element name="ReceiverToken" type="tt:ReferenceToken"
maxOccurs="unbounded"/>
        <xs:element name="Multicast" type="tt:MulticastConfiguration" minOccurs="0"/>
        <xs:element name="Subscription" type="tt:Config"/>
        <xs:element name="Mode" type="tt:ModeOfOperation"/>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType><!--=====-->
<xs:simpleType name="ModeOfOperation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Idle"/>
    <xs:enumeration value="Active"/>
    <xs:enumeration value="Unknown"/>
  </xs:restriction>
</xs:simpleType><!--=====-->
<xs:complexType name="AnalyticsStateInformation">
  <xs:sequence>
    <xs:element name="AnalyticsEngineControlToken" type="tt:ReferenceToken"/>
    <xs:element name="State" type="tt:AnalyticsState"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType><!--=====-->
<xs:complexType name="AnalyticsState">
  <xs:sequence>
    <xs:element name="Error" type="xs:string" minOccurs="0"/>
    <xs:element name="State" type="xs:string"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType>
<!--=====-->
<!-- End, Analytics Device Types -->
<!--=====-->
</xs:schema>

```

C.16 Topic namespace XML

```

<?xml version="1.0" encoding="utf-8"?>
<wstop:TopicNamespace xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:tt="http://www.onvif.org/ver10/schema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="ONVIF"
targetNamespace="http://www.onvif.org/ver10/topics"
xsi:schemaLocation="http://www.onvif.org/ver10/schema
http://www.onvif.org/onvif/ver10/schema/onvif.xsd">
  <wstop:Topic name="Device">
    <wstop:Topic name="Trigger">
      <wstop:Topic name="Relay" messageTypes="tt:Message"/>
    </wstop:Topic>
    <wstop:Topic name="OperationMode">
      <wstop:Topic name="ShutdownInitiated" messageTypes="tt:Message"/>
      <wstop:Topic name="UploadInitiated" messageTypes="tt:Message"/>
    </wstop:Topic>
    <wstop:Topic name="HardwareFailure">
      <wstop:Topic name="FanFailure" messageTypes="tt:Message"/>
      <wstop:Topic name="PowerSupplyFailure" messageTypes="tt:Message"/>
      <wstop:Topic name="StorageFailure" messageTypes="tt:Message"/>
      <wstop:Topic name="TemperatureCritical" messageTypes="tt:Message"/>
    </wstop:Topic>
  </wstop:Topic>
  <wstop:Topic name="VideoSource"/>
  <wstop:Topic name="VideoEncoder"/>
  <wstop:Topic name="VideoAnalytics"/>
  <wstop:Topic name="RuleEngine">
    <wstop:Topic name="FieldDetector">
      <wstop:Topic name="ObjectsInside" messageTypes="tt:Message"/>
    </wstop:Topic>
    <wstop:Topic name="LineDetector">
      <wstop:Topic name="Crossed" messageTypes="tt:Message"/>
    </wstop:Topic>
  </wstop:Topic>
  <wstop:Topic name="PTZController">
    <wstop:Topic name="PTZPresets">
      <wstop:Topic name="Invoked" messageTypes="tt:Message"/>
      <wstop:Topic name="Reached" messageTypes="tt:Message"/>
      <wstop:Topic name="Aborted" messageTypes="tt:Message"/>
      <wstop:Topic name="Left" messageTypes="tt:Message"/>
    </wstop:Topic>
  </wstop:Topic>
  <wstop:Topic name="AudioSource"/>
  <wstop:Topic name="AudioEncoder"/>
  <wstop:Topic name="UserAlarm"/>
  <wstop:Topic name="MediaControl">
    <wstop:Topic name="Profile" messageTypes="tt:Message"/>
    <wstop:Topic name="VideoSourceConfiguration" messageTypes="tt:Message"/>
    <wstop:Topic name="AudioSourceConfiguration" messageTypes="tt:Message"/>
    <wstop:Topic name="VideoEncoderConfiguration" messageTypes="tt:Message"/>
    <wstop:Topic name="AudioEncoderConfiguration" messageTypes="tt:Message"/>
    <wstop:Topic name="VideoAnalyticsConfiguration" messageTypes="tt:Message"/>
    <wstop:Topic name="PTZConfiguration" messageTypes="tt:Message"/>
    <wstop:Topic name="MetaDataConfiguration" messageTypes="tt:Message"/>
  </wstop:Topic>
  <wstop:Topic name="RecordingConfig">
    <wstop:Topic name="JobState" messageTypes="tt:Message"/>
    <wstop:Topic name="RecordingConfiguration" messageTypes="tt:Message"/>
  </wstop:Topic>
</wstop:TopicNamespace>

```

```

<wstop:Topic name="TrackConfiguration" messageTypes="tt:Message"/>
<wstop:Topic name="RecordingJobConfiguration" messageTypes="tt:Message"/>
<wstop:Topic name="DeleteTrackData" messageTypes="tt:Message"/>
<wstop:Topic name="CreateRecording" messageTypes="tt:Message"/>
<wstop:Topic name="DeleteRecording" messageTypes="tt:Message"/>
<wstop:Topic name="CreateTrack" messageTypes="tt:Message"/>
<wstop:Topic name="DeleteTrack" messageTypes="tt:Message"/>
</wstop:Topic>
<wstop:Topic name="RecordingHistory">
  <wstop:Topic name="Recording">
    <wstop:Topic name="State" messageTypes="tt:Message"/>
  </wstop:Topic>
  <wstop:Topic name="Track">
    <wstop:Topic name="State" messageTypes="tt:Message"/>
    <wstop:Topic name="VideoParameters" messageTypes="tt:Message"/>
    <wstop:Topic name="AudioParameters" messageTypes="tt:Message"/>
  </wstop:Topic>
</wstop:Topic>
<wstop:Topic name="VideoOutput"/>
<wstop:Topic name="AudioOutput"/>
<wstop:Topic name="VideoDecoder">
  <wstop:Topic name="DecodingError" messageTypes="tt:Message"/>
</wstop:Topic>
<wstop:Topic name="AudioDecoder"/>
<wstop:Topic name="Receiver">
  <wstop:Topic name="ChangeState" messageTypes="tt:Message"/>
  <wstop:Topic name="ConnectionFailed" messageTypes="tt:Message"/>
</wstop:Topic>
</wstop:TopicNamespace>

```


Bibliography

ISO/IEC 10918-1, *Information technology – Digital compression and coding of continuous-tone still images: Requirements and guidelines*

ISO/IEC 14496-2:2004, *Information technology – Coding of audio-visual objects – Part 2: Visual*

ISO/IEC 14496-3:2009, *Information technology – Coding of audio-visual objects – Part 3: Audio*

ISO/IEC 14496-10:2009, *Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding*

ITU-T G.726, 40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)

<http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.726-199012-I-PDF-E&type=items>

ANSI/SIA DVI-01:2008, *Digital Video Interface Model*

[EAP-Registry] *Extensible Authentication Protocol (EAP) Registry*

<<http://www.iana.org/assignments/eap-numbers/eap-numbers.xml>>

[ONVIF Security] *ONVIF Security Recommendations White Paper*

<http://www.onvif.org/portals/3/documents/whitepapers/ONVIF_Security_Recommendations_ver10.pdf>

[ONVIF PTZ] *ONVIF PTZ Coordinate Spaces White Paper*

<http://www.onvif.org/Portals/0/documents/whitepapers/ONVIF_PTZ_coordinate_spaces.pdf>

RFC 1305, *Network Time Protocol (Version 3), Specification, Implementation and Analysis*

<<http://www.ietf.org/rfc/rfc1305.txt>>

RFC 2104, *HMAC: Keyed-Hashing for Message Authentication* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc2104.txt>>

RFC 2396, *Uniform Resource Identifiers (URI): Generic Syntax*, T. Berners-Lee et al., August 1998

<<http://www.ietf.org/rfc/rfc2396.txt>>

RFC 2986 PKCS #10 *Certification Request Syntax Specification version 1.7*

<<http://www.ietf.org/rfc/rfc2986>>

[WS-KerberosToken] “Web Services Security Kerberos Token Profile 1.1”, OASIS Standard, 1 February 2006.

<<http://www.oasis-open.org/committees/download.php/16788/wss-v1.1-spec-os-KerberosTokenProfile.pdf>>

[WS-SAMLToken] “*Web Services Security: SAML Token Profile 1.1*”, OASIS Standard, 1 February 2006.

<http://www.oasis-open.org/committees/download.php/16768/wss-v1.1-spec-os-SAMLTokenProfile.pdf>

[WS-RELTToken] “*Web Services Security Rights Expression Language (REL) Token Profile 1.1*”, OASIS Standard, 1 February 2006

<http://www.oasis-open.org/committees/download.php/16687/oasis-wss-rel-token-profile-1.1.pdf>

WS-I, *Basic Profile Version 2.0 – Working Group Draft*, C. Ferris (Ed), A. Karmarkar (Ed) and P. Yendluri (Ed), October 2007.

[http://www.ws-i.org/Profiles/BasicProfile-2_0\(WGD\).html](http://www.ws-i.org/Profiles/BasicProfile-2_0(WGD).html)

IETF RFC 2818, *HTTP over TLS*

<http://www.ietf.org/rfc/rfc2818.txt>

IETF RFC 3548, *The Base16, Base32, and Base64 Data Encodings*

<http://www.ietf.org/rfc/rfc3548.txt>

IETF RFC 4122, *A Universally Unique IDentifier (UUID) URN Namespace*

<http://www.ietf.org/rfc/rfc4122.txt>

IETF RFC 4346, *The Transport Layer Security (TLS) Protocol Version 1.1*

<http://www.ietf.org/rfc/rfc4346.txt>

IETF RFC 4585, *Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)*

<http://www.ietf.org/rfc/rfc4585.txt>

IETF RFC 5104, *Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)*

<http://www.ietf.org/rfc/rfc5104.txt>

IETF RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*

<http://www.ietf.org/rfc/rfc5246.txt>

W3C SOAP Message Transmission Optimization Mechanism

<http://www.w3.org/TR/soap12-mtom/>

W3C Web Services Addressing 1.0 – Core

<http://www.w3.org/TR/ws-addr-core/>

XMLSOAP, Web Services Dynamic Discovery (WS-Discovery)”, J. Beatty et al., April 2005.

<http://specs.xmlsoap.org/ws/2005/04/discovery/ws-discovery.pdf>

OASIS Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)

<<http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>>

OASIS Web Services Topics 1.3

<http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf>

W3C Web Services Description Language (WSDL) 1.1

<<http://www.w3.org/TR/wsdl>>

IECNORM.COM: Click to view the full PDF of IEC 62676-2-3:2013

Withdrawing

SOMMAIRE

AVANT-PROPOS	671
INTRODUCTION.....	673
1 Domaine d'application	674
2 Références normatives	676
3 Termes, définitions et abréviations	679
3.1 Termes et définitions	679
3.2 Abréviations	683
4 Présentation	685
4.1 Services web	685
4.2 Configuration IP	686
4.3 Découverte de dispositif	687
4.4 Types de dispositifs	687
4.5 Gestion de dispositif	687
4.5.1 Fonctionnalités	687
4.5.2 Réseau	688
4.5.3 Système	688
4.5.4 Obtention des informations sur le système	689
4.5.5 Mise à niveau de micrologiciel	689
4.5.6 Restauration du système	689
4.5.7 Sécurité	689
4.6 DeviceIO	689
4.7 Configuration d'imagerie	690
4.8 Configuration multimédia	690
4.8.1 Généralités	690
4.8.2 Profils multimédia	691
4.9 Transmission en temps réel	694
4.10 Traitement des événements	695
4.11 Contrôle PTZ	695
4.12 Analyse vidéo	697
4.13 Dispositif d'analyse	699
4.14 Affichage	699
4.15 Récepteur	700
4.15.1 Généralités	700
4.15.2 Points de synchronisation	700
4.16 Stockage	700
4.16.1 Modèle de stockage	701
4.16.2 Enregistrement	702
4.16.3 Recherche	702
4.16.4 Lecture	703
4.17 Sécurité	703
5 Interopérabilité des services Web	704
5.1 Présentation des services	704
5.1.1 Généralités	704
5.1.2 Exigences des services	704
5.2 Présentation de WSDL	705
5.3 Espaces de nom	706

5.4	Types	708
5.5	Messages	708
5.6	Opérations	709
5.6.1	Type d'opération unidirectionnelle	710
5.6.2	Type d'opération demande-réponse (request-response)	711
5.7	Types de port	711
5.8	Liaison	711
5.9	Ports	711
5.10	Services	711
5.11	Traitement des erreurs	711
5.11.1	Erreurs de protocole	712
5.11.2	Erreurs SOAP	712
5.12	Sécurité	715
5.12.1	Contrôle d'accès à base d'utilisateurs	716
5.12.2	Profil de jeton d'utilisateur	716
6	Configuration IP	717
7	Découverte de dispositif	718
7.1	Généralités	718
7.2	Modes de fonctionnement	719
7.3	Définitions de découverte	719
7.3.1	Référence de point terminal	719
7.3.2	Hello	719
7.3.3	Sonde et correspondance de sonde	722
7.3.4	Résolution et correspondance de résolution	723
7.3.5	Bye	723
7.3.6	Messages de défaut SOAP	723
7.4	Extensions de découverte à distance	723
7.4.1	Scénarios de réseau	724
7.4.2	Proxy de découverte (DP)	726
7.4.3	Comportement pour les messages Hello et Probe à distance	727
7.4.4	Comportement du client	728
7.4.5	Sécurité	729
8	Gestion de dispositif	730
8.1	Fonctionnalités	730
8.1.1	Get WSDL URL	730
8.1.2	Echange de fonctionnalité	730
8.2	Réseau	734
8.2.1	Obtention de nom d'hôte	734
8.2.2	Définition de nom d'hôte	734
8.2.3	Obtention de paramètres DNS	735
8.2.4	Définition des paramètres DNS	736
8.2.5	Obtention des paramètres NTP	737
8.2.6	Définition des paramètres NTP	738
8.2.7	Obtention des paramètres de DNS dynamiques	739
8.2.8	Définition des paramètres de DNS dynamiques	740
8.2.9	Obtention de configuration d'interface réseau	741
8.2.10	Définition de configuration d'interface réseau	742
8.2.11	Obtention de protocoles réseau	744
8.2.12	Définition des protocoles réseau	744

8.2.13	Obtention de passerelle par défaut	745
8.2.14	Définition de passerelle par défaut	746
8.2.15	Obtention de configuration zéro	746
8.2.16	Définition de configuration zéro	747
8.2.17	Obtention de filtre d'adresse IP	748
8.2.18	Définition de filtre d'adresse IP	749
8.2.19	Ajout d'une adresse de filtre IP	750
8.2.20	Suppression d'une adresse de filtre IP	751
8.2.21	Configuration IEEE 802.11	752
8.3	Système	757
8.3.1	Informations de dispositif	757
8.3.2	Obtention des URI du système	758
8.3.3	Sauvegarde	759
8.3.4	Restauration	759
8.3.5	Démarrage de la restauration du système	760
8.3.6	Obtention des date et heure système	761
8.3.7	Définition des date et heure système	762
8.3.8	Réglages par défaut d'usine	763
8.3.9	Mise à niveau de micrologiciel	764
8.3.10	Démarrage de la mise à niveau du micrologiciel	765
8.3.11	Obtention des journaux système	766
8.3.12	Obtention d'informations d'assistance	767
8.3.13	Redémarrage	768
8.3.14	Obtention de paramètres de domaine d'application	768
8.3.15	Définition des paramètres de domaine d'application	769
8.3.16	Ajout de paramètres de domaine d'application	770
8.3.17	Suppression de paramètres de domaine d'application	771
8.3.18	Obtention de mode de découverte	771
8.3.19	Définition du mode de découverte	772
8.3.20	Obtention du mode de découverte à distance	772
8.3.21	Définition de mode de découverte à distance	773
8.3.22	Obtention d'adresses DP distantes	773
8.3.23	Définition d'adresses DP distantes	774
8.4	Sécurité	774
8.4.1	Obtention de politique d'accès	774
8.4.2	Définition de politique d'accès	775
8.4.3	Obtention d'utilisateurs	776
8.4.4	Création d'utilisateurs	776
8.4.5	Suppression d'utilisateurs	777
8.4.6	Définition de paramètres d'utilisateur	778
8.4.7	Configuration IEEE 802.1X	779
8.4.8	Création d'un certificat auto-signé	784
8.4.9	Obtention de certificats	785
8.4.10	Obtention de certificats CA	786
8.4.11	Obtention de statut de certificat	786
8.4.12	Définition de statut de certificat	787
8.4.13	Obtention de demande de certificat	787
8.4.14	Obtention de statut de certificat de client	788
8.4.15	Définition de statut de certificat de client	789

8.4.16	Chargement de certificat de dispositif	789
8.4.17	Chargement de certificat de dispositif avec sa clé privée	790
8.4.18	Obtention de demande d'informations de certificat	791
8.4.19	Chargement de certificats CA	792
8.4.20	Suppression de certificat	793
8.4.21	Obtention de l'utilisateur distant	794
8.4.22	Définition de l'utilisateur distant	795
8.4.23	Obtention de la référence de point terminal	796
8.5	Entrée/sortie (E/S)	796
8.5.1	Obtention de sorties de relais	797
8.5.2	Définition de réglages de sortie de relais	797
8.5.3	Déclenchement de sortie de relais	798
8.5.4	Opérations auxiliaires	799
8.6	Codes de défaut spécifiques au service	800
9	Service ES de dispositif	806
9.1	VideoOutputs	806
9.1.1	General	806
9.1.2	GetVideoOutputs	807
9.2	VideoOutputConfiguration	807
9.2.1	GetVideoOutputConfiguration	807
9.2.2	SetVideoOutputConfiguration	808
9.2.3	GetVideoOutputConfigurationOptions	808
9.3	VideoSources	809
9.3.1	Généralités	809
9.3.2	GetVideoSources	809
9.4	VideoSourceConfiguration	810
9.4.1	GetVideoSourceConfiguration	810
9.4.2	SetVideoSourceConfiguration	811
9.4.3	GetVideoSourceConfigurationOptions	811
9.5	AudioOutputs	812
9.5.1	Généralités	812
9.5.2	GetAudioOutputs	812
9.6	AudioOutputConfiguration	813
9.6.1	GetAudioOutputConfiguration	813
9.6.2	SetAudioOutputConfiguration	814
9.6.3	GetAudioOutputConfigurationOptions	814
9.7	AudioSources	815
9.7.1	Généralités	815
9.7.2	GetAudioSources	815
9.8	AudioSourceConfiguration	816
9.8.1	GetAudioSourceConfiguration	816
9.8.2	SetAudioSourceConfiguration	817
9.8.3	GetAudioSourceConfigurationOptions	817
9.9	Sorties relais	818
9.9.1	Obtention de sorties de relais	818
9.9.2	Définition de réglages de sortie de relais	819
9.9.3	Déclenchement de sortie de relais	820
9.10	Codes de défaut spécifiques au service	821
10	Configuration d'imagerie	822

10.1	Paramètres d'imagerie	822
10.1.1	Obtention de paramètres d'imagerie	823
10.1.2	Définition des paramètres d'imagerie	824
10.1.3	Obtention d'options	825
10.1.4	Move	826
10.1.5	Obtention d'options de déplacement	827
10.1.6	Stop	828
10.1.7	Obtention du statut d'imagerie	829
10.2	Codes de défaut spécifiques au service	830
11	Configuration multimédia	831
11.1	Codecs audio et vidéo	831
11.2	Profil multimédia	831
11.2.1	Création de profil multimédia	832
11.2.2	Obtention de profils multimédia	833
11.2.3	Obtention de profil multimédia	834
11.2.4	Ajout de configuration de source vidéo à un profil	835
11.2.5	Ajout de configuration de codeur vidéo à un profil	835
11.2.6	Ajout de configuration de source audio à un profil	836
11.2.7	Ajout de configuration de codeur audio à un profil	837
11.2.8	Ajout de configuration PTZ à un profil	838
11.2.9	Ajout de configuration d'analyse vidéo à un profil	839
11.2.10	Ajout de configuration de métadonnées à un profil	840
11.2.11	Ajout de configuration de sortie audio	841
11.2.12	Ajout de configuration de décodeur audio	842
11.2.13	Suppression de configuration de source vidéo d'un profil	843
11.2.14	Suppression de configuration de codeur vidéo d'un profil	844
11.2.15	Suppression de configuration de source audio d'un profil	845
11.2.16	Suppression de configuration de codeur audio d'un profil	846
11.2.17	Suppression de configuration PTZ d'un profil	847
11.2.18	Suppression de configuration d'analyse vidéo d'un profil	848
11.2.19	Suppression de configuration de métadonnées d'un profil	849
11.2.20	Suppression de configuration de sortie audio	850
11.2.21	Suppression de configuration de décodeur audio	851
11.2.22	Suppression de profil multimédia	852
11.3	Source vidéo	853
11.3.1	Généralités	853
11.3.2	GetVideoSources	853
11.4	Configuration de source vidéo	854
11.4.1	Obtention de configurations de source vidéo	854
11.4.2	Obtention de configuration de source vidéo	855
11.4.3	Obtention de configurations de source vidéo compatibles	855
11.4.4	Obtention des options de configuration de source vidéo	856
11.4.5	Modification d'une configuration de source vidéo	857
11.5	Configuration de codeur vidéo	858
11.5.1	Obtention de configurations de codeur vidéo	859
11.5.2	Obtention de configuration de codeur vidéo	859
11.5.3	Obtention de configurations de codeur vidéo compatibles	860
11.5.4	Obtention d'options de configuration de codeur vidéo	861
11.5.5	Modification d'une configuration de codeur vidéo	862

11.5.6	Obtention d'un nombre garanti d'instances de codeur vidéo	863
11.6	Source audio	864
11.6.1	Généralités	864
11.6.2	Obtention de sources audio	864
11.7	Configuration de source audio	865
11.7.1	Obtention de configurations de source audio	865
11.7.2	Obtention de configuration de source audio	866
11.7.3	Obtention de configurations de source audio compatibles	867
11.7.4	Obtention d'options de configuration de source audio	868
11.7.5	Modification d'une configuration de source audio	869
11.8	Configuration de codeur audio	870
11.8.1	Obtention de configurations de codeur audio	871
11.8.2	Obtention de configuration de codeur audio	871
11.8.3	Obtention de configurations de codeur audio compatibles	872
11.8.4	Obtention d'options de configuration de codeur audio	873
11.8.5	Modification de configurations de codeur audio	874
11.9	Configuration d'analyse vidéo	875
11.9.1	Obtention de configurations d'analyse vidéo	876
11.9.2	Obtention de configuration d'analyse vidéo	876
11.9.3	Obtention de configurations d'analyse vidéo compatibles	877
11.9.4	Modification d'une configuration d'analyse vidéo	878
11.10	Configuration de métadonnées	879
11.10.1	Obtention de configurations de métadonnées	880
11.10.2	Obtention de configuration de métadonnées	880
11.10.3	Obtention de configurations de métadonnées compatibles	881
11.10.4	Obtention d'options de configuration de métadonnées	882
11.10.5	Modification d'une configuration de métadonnées	883
11.11	Sorties audio	884
11.11.1	Généralités	884
11.11.2	Obtention de sorties audio	884
11.12	Configuration de sortie audio	885
11.12.1	Obtention de configurations de sortie audio	885
11.12.2	Obtention d'une configuration de sortie audio	886
11.12.3	Obtention de configurations de sortie audio compatibles	887
11.12.4	Obtention d'options de configuration de sortie audio	887
11.12.5	Modification de configuration de sortie audio	888
11.13	Configuration de décodeur audio	889
11.13.1	Obtention de configurations de décodeur audio	890
11.13.2	Obtention d'une configuration de décodeur audio	890
11.13.3	Obtention de configurations de décodeur audio compatibles	891
11.13.4	Obtention d'options de configuration de décodeur audio	892
11.13.5	Modification d'une configuration de décodeur audio	893
11.14	Modes de voie audio	894
11.15	URI de flux	895
11.15.1	Généralités	895
11.15.2	Demande d'URI de flux	895
11.16	Instantané	897
11.16.1	Généralités	897
11.16.2	Demande d'URI d'instantané	897

11.17	Multidiffusion	898
11.17.1	Démarrage de transmission continue en multidiffusion	898
11.17.2	Arrêt de transmission continue en multidiffusion	898
11.18	Points de synchronisation	899
11.18.1	Généralités	899
11.18.2	Définition de point de synchronisation	899
11.19	Codes de défaut spécifiques au service	900
12	Transmission continue en temps réel	901
12.1	Protocole de flux multimédia	902
12.1.1	Format de transport	902
12.1.2	Transport de support	902
12.1.3	Point de synchronisation	907
12.1.4	JPEG via RTP	908
12.2	Protocole de contrôle multimédia	911
12.2.1	Contrôle de flux	911
12.3	Connexion de la voie de retour	916
12.3.1	Balise RTSP Require	916
12.3.2	Configuration des connexions pour une connexion bidirectionnelle	917
12.3.3	Transmission continue en multidiffusion	919
12.4	Gestion des erreurs	919
13	Configuration du récepteur	919
13.1	Persistance	920
13.2	Modes du récepteur	920
13.3	Commandes du récepteur	920
13.3.1	Obtention des récepteurs	920
13.3.2	Obtention du récepteur	920
13.3.3	Création de récepteur	921
13.3.4	Suppression de récepteur	921
13.3.5	Configuration de récepteur	922
13.3.6	SetReceiverMode	923
13.3.7	GetReceiverState	923
13.4	Événements	923
13.4.1	ChangeState	924
13.4.2	Échec de la connexion	924
13.5	Codes de défaut spécifiques au service	924
14	Service d'affichage	924
14.1	Panneaux	925
14.1.1	GetPaneConfigurations	926
14.1.2	GetPaneConfiguration	926
14.1.3	SetPaneConfigurations	927
14.1.4	SetPaneConfiguration	928
14.1.5	CreatePaneConfiguration	929
14.1.6	DeletePaneConfiguration	930
14.2	Présentation	931
14.2.1	GetLayout	931
14.2.2	SetLayout	932
14.3	DisplayOptions	933
14.3.1	Généralités	933
14.3.2	LayoutOptions	933

14.3.3	CodingCapabilities.....	933
14.3.4	GetDisplayOptions.....	933
14.4	Evénements	934
14.4.1	Généralités.....	934
14.4.2	Décodage d'événement d'erreur	934
14.5	Codes de défaut spécifiques au service.....	935
15	Traitement des événements.....	935
15.1	Interface de notification de base.....	936
15.1.1	Généralités.....	936
15.1.2	Exigences.....	937
15.2	Interface de notification Real-time Pull-Point	938
15.2.1	Création d'abonnement de point d'extraction	939
15.2.2	Messages Pull	940
15.3	Interface de transmission en continu de notification.....	941
15.4	Propriétés	941
15.4.1	Exemple de propriété.....	942
15.5	Structure des notifications	942
15.5.1	Informations de notification.....	943
15.5.2	Format de message	944
15.5.3	Exemple de propriété (suite).....	945
15.5.4	Langage de description de message.....	946
15.5.5	Filtre de contenu de message.....	947
15.6	Point de synchronisation	949
15.7	Structure de rubrique.....	949
15.7.1	Espaces de noms de rubrique ONVIF	950
15.7.2	Informations de type de rubrique.....	950
15.7.3	Filtre de rubrique	951
15.8	Obtention de propriétés d'événement	952
15.9	Messages de défaut SOAP	953
15.10	Exemple de notification	953
15.10.1	GetEventPropertiesRequest	954
15.10.2	GetEventPropertiesResponse.....	954
15.10.3	CreatePullPointSubscription	955
15.10.4	CreatePullPointSubscriptionResponse	956
15.10.5	PullMessagesRequest	956
15.10.6	PullMessagesResponse	957
15.10.7	UnsubscribeRequest	958
15.11	Codes de défaut spécifiques au service.....	958
16	Contrôle PTZ	958
16.1	Modèle PTZ.....	959
16.2	Noeud PTZ.....	960
16.2.1	GetNodes	961
16.2.2	GetNode.....	961
16.3	Configuration PTZ	962
16.3.1	GetConfigurations.....	963
16.3.2	GetConfiguration	964
16.3.3	GetConfigurationOptions	965
16.3.4	SetConfiguration.....	965
16.4	Opérations de mouvement.....	966

16.4.1	AbsoluteMove.....	967
16.4.2	RelativeMove.....	968
16.4.3	ContinuousMove.....	970
16.4.4	Stop	971
16.4.5	GetStatus	972
16.5	Opérations de préréglage.....	973
16.5.1	SetPreset	973
16.5.2	GetPresets	975
16.5.3	GotoPreset.....	975
16.5.4	RemovePreset.....	976
16.6	Opérations de position de départ.....	977
16.6.1	GotoHomePosition.....	977
16.6.2	SetHomePosition	978
16.7	Opérations auxiliaires.....	979
16.7.1	Généralités.....	979
16.7.2	SendAuxiliaryCommand.....	979
16.8	Espaces PTZ prédéfinis.....	980
16.8.1	Espaces de position absolue	980
16.8.2	Espaces de translation relative.....	982
16.8.3	Espaces de vitesse continue.....	983
16.8.4	Espaces de vitesse.....	984
16.9	Codes de défaut spécifiques au service.....	985
17	Analyse vidéo.....	987
17.1	Interface de description de scène.....	988
17.1.1	Présentation.....	988
17.1.2	Contenu associé aux trames.....	988
17.1.3	Éléments de scène.....	991
17.2	Interface de règles.....	996
17.2.1	Représentation des règles.....	996
17.2.2	Langage de description de règles.....	996
17.2.3	Règles normalisées.....	997
17.2.4	Opérations sur les règles.....	999
17.3	Interface de module d'analyse.....	1003
17.3.1	Configuration de module d'analyse.....	1003
17.3.2	Langage de description de module d'analyse.....	1004
17.3.3	Opérations sur les modules d'analyse.....	1004
17.4	Codes de défaut spécifiques au service.....	1008
18	Dispositif d'analyse	1009
18.1	Présentation.....	1010
18.2	Entrée de moteur d'analyse.....	1010
18.2.1	GetAnalyticsEngineInputs.....	1010
18.2.2	GetAnalyticsEngineInput.....	1011
18.2.3	SetAnalyticsEngineInput.....	1011
18.2.4	CreateAnalyticsEngineInputs	1012
18.2.5	DeleteAnalyticsEngineInputs	1013
18.3	Configuration d'analyse vidéo.....	1014
18.3.1	GetVideoAnalyticsConfiguration	1014
18.3.2	SetVideoAnalyticsConfiguration.....	1015
18.4	Moteurs d'analyse	1016

18.4.1	GetAnalyticsEngines.....	1016
18.4.2	GetAnalyticsEngine	1017
18.5	Contrôle de moteur d'analyse	1017
18.5.1	GetAnalyticsEngineControls	1018
18.5.2	GetAnalyticsEngineControl	1018
18.5.3	SetAnalyticsEngineControl	1019
18.5.4	CreateAnalyticsEngineControl	1020
18.5.5	DeleteAnalyticsEngineControl.....	1021
18.6	GetAnalyticsState.....	1022
18.7	Configuration de flux de sortie en continu	1023
18.7.1	Généralités.....	1023
18.7.2	Demande d'URI de flux.....	1023
19	Contrôle d'enregistrement.....	1024
19.1	Généralités.....	1024
19.2	Exigences générales	1026
19.3	Structures de données	1026
19.3.1	RecordingConfiguration	1026
19.3.2	TrackConfiguration	1027
19.3.3	RecordingJobConfiguration.....	1027
19.4	CreateRecording	1028
19.5	DeleteRecording.....	1029
19.6	GetRecordings	1030
19.7	SetRecordingConfiguration.....	1030
19.8	GetRecordingConfiguration.....	1031
19.9	CreateTrack	1032
19.10	DeleteTrack.....	1032
19.11	GetTrackConfiguration	1033
19.12	SetTrackConfiguration.....	1034
19.13	CreateRecordingJob.....	1035
19.14	DeleteRecordingJob	1036
19.15	GetRecordingJobs.....	1037
19.16	SetRecordingJobConfiguration	1037
19.17	GetRecordingJobConfiguration.....	1038
19.18	SetRecordingJobMode	1039
19.19	GetRecordingJobState	1040
19.20	Événements	1041
19.20.1	Enregistrement des modifications d'état de travail	1041
19.20.2	Modifications de configuration	1042
19.20.3	Suppression de données	1042
19.20.4	Enregistrement et création et suppression de piste	1043
19.21	Exemples	1043
19.21.1	Exemple 1: Configuration de l'enregistrement d'une seule caméra	1043
19.21.2	Exemple 2: Enregistrement de plusieurs flux d'une caméra vers un seul enregistrement.....	1044
20	Recherche d'enregistrement.....	1045
20.1	Généralités.....	1045
20.2	Concepts.....	1046
20.2.1	Direction de recherche	1046
20.2.2	Événement d'enregistrement	1046

20.2.3	Session de recherche	1047
20.2.4	Etendue de la recherche	1047
20.2.5	Filtres de recherche	1047
20.3	Structures de données	1047
20.3.1	Structure RecordingInformation	1047
20.3.2	Structure RecordingSourceInformation	1048
20.3.3	Structure TrackInformation	1048
20.3.4	Énumération SearchState	1048
20.3.5	Structure MediaAttributes	1048
20.3.6	Structure FindEventResult	1049
20.3.7	Structure FindPTZPositionResult	1049
20.3.8	Structure PTZPositionFilter	1049
20.3.9	Structure MetadataFilter	1049
20.3.10	Structure FindMetadataResult	1049
20.4	GetRecordingSummary	1049
20.5	GetRecordingInformation	1050
20.6	GetMediaAttributes	1050
20.7	FindRecordings	1051
20.8	GetRecordingSearchResults	1052
20.9	FindEvents	1053
20.10	GetEventSearchResults	1054
20.11	FindPTZPosition	1055
20.12	GetPTZPositionSearchResults	1056
20.13	FindMetadata	1057
20.14	GetMetadataSearchResults	1058
20.15	GetSearchState	1060
20.16	EndSearch	1061
20.17	Descriptions d'événement d'enregistrement	1062
20.18	Dialecte XPath	1064
21	Contrôle de lecture	1065
21.1	Utilisation de RTSP	1065
21.2	Extension d'en-tête RTP	1066
21.2.1	Horodatages NTP	1067
21.2.2	Compatibilité avec l'extension d'en-tête JPEG	1067
21.3	Balise de caractéristique RTSP	1067
21.4	Lancement de la restitution	1068
21.4.1	Champ d'en-tête Range	1068
21.4.2	Champ d'en-tête Rate-Control	1069
21.4.3	Champ d'en-tête Frames	1069
21.4.4	Points de synchronisation	1070
21.5	Lecture inversée	1070
21.5.1	Ordre de transmission de paquet	1070
21.5.2	Numéros de séquence RTP	1071
21.5.3	Horodatages RTP	1071
21.6	RTSP keepalive	1071
21.7	Enregistrement du métrage en cours	1071
21.8	Fin de métrage	1072
21.9	Go to time	1072
21.10	Utilisation de RTCP	1072

21.11 Commandes de service de lecture.....	1072
21.11.1 Demande d'URI de lecture	1072
21.11.2 ReplayConfiguration	1073
21.11.3 SetReplayConfiguration.....	1073
21.11.4 GetReplayConfiguration	1074
21.11.5 Codes de défaut spécifiques au service	1074
22 Sécurité.....	1075
22.1 Sécurité niveau transport.....	1075
22.1.1 Suites de chiffrement prises en charge	1076
22.1.2 Authentification du serveur	1076
22.1.3 Authentification de client	1076
22.2 Sécurité niveau message	1077
22.3 IEEE 802.1X.....	1077
Annexe A (informative) Rubriques de notification	1078
Annexe B (informative) Descriptions de scène	1081
Annexe C (normative) Schéma XML d'interface réseau IP vidéo	1083
Bibliographie.....	1310
Figure 1 – Principes de développement des services Web.....	686
Figure 2 – Profil multimédia	691
Figure 3 – Configuration de profil complète.....	692
Figure 4 – Structure de couche	694
Figure 5 – Modèle de contrôle PTZ	696
Figure 6 – Architecture d'analyse vidéo.....	698
Figure 7 – Modèle de stockage avec pistes.....	701
Figure 8 – Dispositif (NVT, par exemple) dans un domaine administratif (privé) et client (NVC) dans un réseau public	725
Figure 9 – Dispositif (NVT, par exemple) dans un réseau public et client (NVC) dans un domaine administratif (privé).....	725
Figure 10 – Dispositif (NVT, par exemple) dans un domaine administratif (privé) et client (NVC) dans un autre domaine administratif (privé).....	726
Figure 11 – Dispositif (NVT, par exemple) et client (NVC) dans un réseau public.....	726
Figure 12 – Schéma d'échange de message de découverte à distance entre un dispositif (NVT, par exemple) et un DP domestique	728
Figure 13 – Séquence de messages pour des clients (NVC) préconfigurés avec l'adresse de DP domestique	729
Figure 14 – En-tête RTP	903
Figure 15 – Séquence RTCP	906
Figure 16 – RTCP Sender Report	907
Figure 17 – Synchronisation multimédia.....	907
Figure 18 – Structure de paquet RTP/JPEG (seul le contenu type est présenté pour la charge utile d'extension)	909
Figure 19 – Contrôle de flux.....	912
Figure 20 – Méthode "Keep Alive".....	914
Figure 21 – Schéma de séquence de l'interface de notification de base	937
Figure 22 – Schéma de séquence pour l'interface de notification Real-time Pull-Point	939

Figure 23 – Système de coordonnées de trame par défaut.....	990
Figure 24 – Exemple d'enregistrements et de pistes	1025
Figure 25 – Schéma des éléments RecordingJobConfiguration	1027
Figure 26 – Schéma des éléments RecordingJobStateInformation	1040
Tableau 1 – Exigences des services en fonction des types de dispositif.....	705
Tableau 2 – Espaces de nom définis dans la présente norme	706
Tableau 3 – Espaces de nom référencés (avec préfixe)	707
Tableau 4 – Espaces de nom référencés (sans préfixe)	708
Tableau 5 – Description des opérations utilisée dans la présente norme.....	710
Tableau 6 – Défauts généraux	714
Tableau 7 – Erreurs HTTP	715
Tableau 8 – Paramètres de domaine d'application	721
Tableau 9 – Commande Get WSDL URL.....	730
Tableau 10 – Commande GetCapabilities	731
Tableau 11 – Fonctionnalités dans la commande GetCapabilities.....	732
Tableau 12 – Commande GetHostname.....	734
Tableau 13 – Commande SetHostname.....	735
Tableau 14 – Commande GetDNS	736
Tableau 15 – Commande SetDNS.....	737
Tableau 16 – Commande GetNTP.....	738
Tableau 17 – Commande SetNTP	739
Tableau 18 – Commande GetDynamicDNS.....	740
Tableau 19 – Commande SetDynamicDNS	741
Tableau 20 – Commande GetNetworkInterfaces	741
Tableau 21 – Commande SetNetworkInterfaces.....	742
Tableau 22 – Commande GetNetworkProtocols	744
Tableau 23 – Commande SetNetworkProtocols.....	745
Tableau 24 – Commande GetNetworkDefaultGateway	745
Tableau 25 – Commande SetNetworkDefaultGateway.....	746
Tableau 26 – Commande GetZeroConfiguration.....	747
Tableau 27 – Commande SetZeroConfiguration	748
Tableau 28 – Commande GetIPAddressFilter.....	749
Tableau 29 – Commande SetIPAddressFilter	750
Tableau 30 – Commande AddIPAddressFilter	751
Tableau 31 – Commande RemoveIPAddressFilter	752
Tableau 32 – Fonctionnalités GetDot11	755
Tableau 33 – Fonctionnalités IEEE 802.11.....	755
Tableau 34 – GetDot11Status	756
Tableau 35 – ScanAvailable802.11Networks.....	757
Tableau 36 – Commande GetDeviceInformation	758
Tableau 37 – Commande GetSystemUri.....	759
Tableau 38 – Commande GetSystemBackup	759

Tableau 39 – Commande RestoreSystem	760
Tableau 40 – Commande StartSystemRestore	761
Tableau 41 – Commande GetSystemDateAndTime	762
Tableau 42 – Commande SetSystemDateAndTime	763
Tableau 43 – Commande SetSystemFactoryDefault	764
Tableau 44 – Commande UpgradeSystemFirmware	765
Tableau 45 – Commande StartFirmwareUpgrade	766
Tableau 46 – Commande GetSystemLog	767
Tableau 47 – Commande GetSystemSupportInformation	768
Tableau 48 – Commande SystemReboot	768
Tableau 49 – Commande GetScopes	769
Tableau 50 – Commande SetScopes	770
Tableau 51 – Commande AddScopes	770
Tableau 52 – Commande RemoveScopes	771
Tableau 53 – Commande GetDiscoveryMode	772
Tableau 54 – Commande SetDiscoveryMode	772
Tableau 55 – Commande GetRemoteDiscoveryMode	773
Tableau 56 – Commande SetRemoteDiscoveryMode	773
Tableau 57 – Commande GetDPAddresses	774
Tableau 58 – Commande SetDPAddresses	774
Tableau 59 – Commande GetAccessPolicy	775
Tableau 60 – Commande SetAccessPolicy	775
Tableau 61 – Commande GetUsers	776
Tableau 62 – Commande CreateUsers	777
Tableau 63 – Commande DeleteUsers	778
Tableau 64 – Commande SetUser	779
Tableau 65 – Commande CreateDot1XConfiguration	781
Tableau 66 – Commande SetDot1XConfigurationRequest	782
Tableau 67 – Commande GetDot1XConfiguration	783
Tableau 68 – Commande GetDot1XConfigurations	783
Tableau 69 – Commande DeleteDot1XConfigurations	784
Tableau 70 – Commande CreateCertificate	785
Tableau 71 – Commande GetCertificates	786
Tableau 72 – Commande GetCACertificates	786
Tableau 73 – Commande GetCertificatesStatus	787
Tableau 74 – Commande SetCertificatesStatus	787
Tableau 75 – Commande GetPkcs10Request	788
Tableau 76 – Commande GetClientCertificateMode	789
Tableau 77 – Commande SetClientCertificateMode	789
Tableau 78 – Commande LoadCertificates	790
Tableau 79 – Commande LoadCertificateWithPrivateKey	791
Tableau 80 – Commande GetCertificateInformation	792
Tableau 81 – Commande LoadCACertificates	793

Tableau 82 – Commande DeleteCertificates	794
Tableau 83 – Commande GetRemoteUser	795
Tableau 84 – Commande SetRemoteUser	796
Tableau 85 – Commande GetEndpointReference	796
Tableau 86 – Commande GetRelayOutputs	797
Tableau 87 – Commande SetRelayOutputSettings	798
Tableau 88 – Commande SetRelayOutputState	799
Tableau 89 – Commande Send auxiliary	800
Tableau 90 – Codes de défaut spécifiques au service de dispositif	801
Tableau 91 – Commande GetVideoOutputs	807
Tableau 92 – Commande GetVideoOutputConfiguration	807
Tableau 93 – Commande SetVideoOutputConfiguration	808
Tableau 94 – Commande GetVideoOutputConfigurationOptions	809
Tableau 95 – Commande GetVideoSources	810
Tableau 96 – Commande GetVideoSourceConfiguration	810
Tableau 97 – Commande SetVideoSourceConfiguration	811
Tableau 98 – Commande GetVideoSourceConfigurationOptions	812
Tableau 99 – Commande GetAudioOutputs	813
Tableau 100 – Commande GetAudioOutputConfiguration	813
Tableau 101 – Commande SetAudioOutputConfiguration	814
Tableau 102 – Commande GetAudioOutputConfigurationOptions	815
Tableau 103 – Commande GetAudioSources	816
Tableau 104 – Commande GetAudioSourceConfiguration	816
Tableau 105 – Commande SetAudioSourceConfiguration	817
Tableau 106 – Commande GetAudioSourceConfigurationOptions	818
Tableau 107 – Commande GetRelayOutputs	819
Tableau 108 – Commande SetRelayOutputSettings	820
Tableau 109 – Commande SetRelayOutputState	821
Tableau 110 – Codes de défaut spécifiques au service DeviceIO	821
Tableau 111 – Commande GetImagingSettings	824
Tableau 112 – Commande SetImagingSettings	825
Tableau 113 – Commande GetOptions	826
Tableau 114 – Commande Move (mise au point)	827
Tableau 115 – Commande GetMoveOptions (mise au point)	828
Tableau 116 – Commande Stop (mise au point)	829
Tableau 117 – Commande GetStatus (mise au point)	830
Tableau 118 – Codes de défaut spécifiques d'imagerie	830
Tableau 119 – Commande CreateProfile	833
Tableau 120 – Commande GetProfiles	834
Tableau 121 – Commande GetProfile	834
Tableau 122 – Commande AddVideoSourceConfiguration	835
Tableau 123 – Commande AddVideoEncoderConfiguration	836
Tableau 124 – Commande AddAudioSourceConfiguration	837

Tableau 125 – Commande AddAudioEncoderConfiguration	838
Tableau 126 – Commande AddPTZConfiguration	839
Tableau 127 – Commande AddVideoAnalytics	840
Tableau 128 – Commande AddMetadataConfiguration	841
Tableau 129 – AddAudioOutputConfiguration	842
Tableau 130 – AddAudioDecoderConfiguration	843
Tableau 131 – Commande RemoveVideoSourceConfiguration	844
Tableau 132 – Commande RemoveVideoEncoderConfiguration	845
Tableau 133 – Commande RemoveAudioSourceConfiguration	846
Tableau 134 – Commande RemoveAudioEncoderConfiguration	847
Tableau 135 – Commande RemovePTZConfiguration	848
Tableau 136 – Commande RemoveVideoAnalyticsConfiguration	849
Tableau 137 – Commande RemoveMetadataConfiguration	850
Tableau 138 – RemoveAudioOutputConfiguration	851
Tableau 139 – RemoveAudioDecoderConfiguration	852
Tableau 140 – Commande DeleteProfile	853
Tableau 141 – Commande GetVideoSources	854
Tableau 142 – Commande GetVideoSourceConfigurations	854
Tableau 143 – Commande GetVideoSourceConfiguration	855
Tableau 144 – Commande GetCompatibleVideoSourceConfigurations	856
Tableau 145 – Commande GetVideoSourceConfigurationOptions	857
Tableau 146 – Commande SetVideoSourceConfiguration	858
Tableau 147 – Commande GetVideoEncoderConfigurations	859
Tableau 148 – Commande GetVideoEncoderConfiguration	860
Tableau 149 – Commande GetCompatibleVideoEncoderConfigurations	861
Tableau 150 – Commande GetVideoEncoderConfigurationOptions	862
Tableau 151 – Commande SetVideoEncoderConfiguration	863
Tableau 152 – Commande GetGuaranteedNumberOfVideoEncoderInstances	864
Tableau 153 – Commande GetAudioSources	865
Tableau 154 – Commande GetAudioSourceConfigurations	866
Tableau 155 – Commande GetAudioSourceConfiguration	867
Tableau 156 – Commande GetCompatibleAudioSourceConfigurations	868
Tableau 157 – Commande GetAudioSourceConfigurationOptions	869
Tableau 158 – Commande SetAudioSourceConfiguration	870
Tableau 159 – Commande GetAudioEncoderConfigurations	871
Tableau 160 – Commande GetAudioEncoderConfiguration	872
Tableau 161 – Commande GetCompatibleAudioEncoderConfigurations	873
Tableau 162 – Commande GetAudioEncoderConfigurationOptions	874
Tableau 163 – Commande SetAudioEncoderConfiguration	875
Tableau 164 – Commande GetVideoAnalyticsConfigurations	876
Tableau 165 – Commande GetVideoAnalyticsConfiguration	877
Tableau 166 – Commande GetCompatibleVideoAnalyticsConfigurations	878
Tableau 167 – Commande SetVideoAnalyticsConfiguration	879

Tableau 168 – Commande GetMetadataConfigurations	880
Tableau 169 – Commande GetMetadataConfiguration	881
Tableau 170 – Commande GetCompatibleMetadataConfigurations	882
Tableau 171 – Commande GetMetadataConfigurationOptions	883
Tableau 172 – Commande SetMetadataConfiguration	884
Tableau 173 – Commande GetAudioOutputs	885
Tableau 174 – GetAudioOutputConfiguration	886
Tableau 175 – GetAudioOutputConfiguration	886
Tableau 176 – GetCompatibleAudioOutputConfiguration	887
Tableau 177 – Commande GetAudioOutputConfigurationOptions	888
Tableau 178 – SetAudioOutputConfiguration	889
Tableau 179 – GetAudioDecoderConfigurations	890
Tableau 180 – GetAudioDecoderConfiguration	891
Tableau 181 – GetCompatibleAudioDecoderConfigurations	892
Tableau 182 – GetAudioDecoderConfigurationOptions	893
Tableau 183 – SetAudioDecoderConfiguration	894
Tableau 184 – Commande GetStreamUri	896
Tableau 185 – Commande GetSnapshotUri	897
Tableau 186 – Commande StartMulticastStreaming	898
Tableau 187 – Commande StopMulticastStreaming	899
Tableau 188 – Commande SetSynchronizationPoint	900
Tableau 189 – Codes de défaut spécifiques au service multimédia	901
Tableau 190 – Valeur d'en-tête RTP	903
Tableau 191 – Méthodes RTSP	913
Tableau 192 – Commande GetReceivers	920
Tableau 193 – Commande GetReceiver	921
Tableau 194 – Commande CreateReceiver	921
Tableau 195 – Commande DeleteReceiver	922
Tableau 196 – Commande ConfigureReceiver	922
Tableau 197 – Commande SetReceiverMode	923
Tableau 198 – Commande GetReceiverState	923
Tableau 199 – Codes de défaut spécifiques au service	924
Tableau 200 – GetPaneConfigurations	926
Tableau 201 – GetPaneConfiguration	927
Tableau 202 – SetPaneConfigurations	928
Tableau 203 – SetPaneConfiguration	929
Tableau 204 – CreatePaneConfiguration	930
Tableau 205 – DeletePaneConfiguration	931
Tableau 206 – GetLayout	932
Tableau 207 – SetLayout	932
Tableau 208 – GetDisplayOptions	934
Tableau 209 – Codes de défaut spécifiques au service	935
Tableau 210 – Commande CreatePullPointSubscription	940

Tableau 211 – Commande PullMessages.....	941
Tableau 212 – Commande SetSynchronizationPoint	949
Tableau 213 – Commande GetEventProperties.....	953
Tableau 214 – Commande GetNodes.....	961
Tableau 215 – Commande GetNode	962
Tableau 216 – Commande GetConfigurations	964
Tableau 217 – Commande GetConfiguration.....	964
Tableau 218 – Commande GetConfigurationOptions	965
Tableau 219 – Commande SetConfiguration	966
Tableau 220 – Commande AbsoluteMove	968
Tableau 221 – Commande RelativeMove	969
Tableau 222 – Commande ContinuousMove	971
Tableau 223 – Commande Stop (PTZ)	972
Tableau 224 – Commande GetStatus (PTZ).....	973
Tableau 225 – Commande SetPreset.....	974
Tableau 226 – Commande GetPresets.....	975
Tableau 227 – Commande GotoPreset.....	976
Tableau 228 – Commande RemovePreset.....	977
Tableau 229 – Commande GotoHomePosition.....	978
Tableau 230 – Commande SetHomePosition	979
Tableau 231 – Commande SendAuxiliary.....	980
Tableau 232 – Codes de défaut spécifiques au service PTZ	986
Tableau 233 – Commande GetSupportedRules.....	999
Tableau 234 – Commande GetRules.....	1000
Tableau 235 – Commande CreateRules.....	1001
Tableau 236 – Commande ModifyRules.....	1002
Tableau 237 – Commande DeleteRules	1003
Tableau 238 – Commande GetSupportedAnalyticsModules.....	1005
Tableau 239 – Commande GetAnalyticsModules.....	1005
Tableau 240 – Commande CreateAnalyticsModules.....	1006
Tableau 241 – Commande ModifyAnalyticsModules.....	1007
Tableau 242 – Commande DeleteAnalyticsModules	1008
Tableau 243 – Codes de défaut spécifiques au service d'analyse	1009
Tableau 244 – Commande GetAnalyticsEngineInputs	1011
Tableau 245 – Commande GetAnalyticsEngineInput	1011
Tableau 246 – Commande SetAnalyticsEngineInput	1012
Tableau 247 – Commande CreateAnalyticsEngineInputs	1013
Tableau 248 – Commande DeleteAnalyticsEngineInputs.....	1014
Tableau 249 – Commande GetVideoAnalyticsConfiguration	1015
Tableau 250 – Commande SetVideoAnalyticsConfiguration	1016
Tableau 251 – Commande GetAnalyticsEngines	1017
Tableau 252 – Commande GetAnalyticsEngine.....	1017
Tableau 253 – Commande GetAnalyticsEngineControls.....	1018

Tableau 254 – Commande GetAnalyticsEngineControl.....	1019
Tableau 255 – Commande SetAnalyticsEngineControl.....	1020
Tableau 256 – Commande CreateAnalyticsEngineControl.....	1021
Tableau 257 – Commande DeleteAnalyticsEngineControl.....	1022
Tableau 258 – GetAnalyticsState.....	1023
Tableau 259 – Commande GetAnalyticsDeviceStreamUri.....	1024
Tableau 260 – Commande CreateRecording.....	1029
Tableau 261 – Commande DeleteRecording.....	1030
Tableau 262 – Commande GetRecordings.....	1030
Tableau 263 – Commande SetRecordingConfiguration.....	1031
Tableau 264 – Commande GetRecordingConfiguration.....	1031
Tableau 265 – Commande CreateTrack.....	1032
Tableau 266 – Commande DeleteTrack.....	1033
Tableau 267 – Commande GetTrackConfiguration.....	1034
Tableau 268 – Commande SetTrackConfiguration.....	1035
Tableau 269 – Commande CreateRecordingJob.....	1036
Tableau 270 – Commande DeleteRecordingJob.....	1037
Tableau 271 – Commande GetRecordingJobs.....	1037
Tableau 272 – Commande SetRecordingJobConfiguration.....	1038
Tableau 273 – Commande GetRecordingJobConfiguration.....	1039
Tableau 274 – Commande SetRecordingJobMode.....	1039
Tableau 275 – Commande GetRecordingJobState.....	1040
Tableau 276 – Commande GetRecordingSummary.....	1050
Tableau 277 – Commande GetRecordingInformation.....	1050
Tableau 278 – Commande GetMediaAttributes.....	1051
Tableau 279 – Commande FindRecordings.....	1052
Tableau 280 – Commande GetRecordingSearchResults.....	1053
Tableau 281 – Commande FindEvents.....	1054
Tableau 282 – Commande GetEventSearchResults.....	1055
Tableau 283 – Commande FindPTZPosition.....	1056
Tableau 284 – Commande GetPTZPositionSearchResults.....	1057
Tableau 285 – Commande FindMetadata.....	1058
Tableau 286 – Commande GetMetadataSearchResults.....	1060
Tableau 287 – Commande GetSearchState.....	1061
Tableau 288 – Commande EndSearch.....	1061
Tableau 289 – Présentation d'un paquet RTP.....	1066
Tableau 290 – Présentation de paquet RTP avec présentation d'en-tête JPEG.....	1067
Tableau 291 – Commande GetReplayUri.....	1073
Tableau 292 – Commande SetReplayConfiguration.....	1074
Tableau 293 – Commande GetReplayConfiguration.....	1074
Tableau 294 – Codes de défaut spécifiques au service de lecture.....	1075

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

**SYSTÈMES DE VIDÉOSURVEILLANCE DESTINÉS
À ÊTRE UTILISÉS DANS LES APPLICATIONS DE SÉCURITÉ –****Partie 2-3: Protocoles de transmission vidéo –
Mise en œuvre de l'interopérabilité IP en fonction des services Web**

AVANT-PROPOS

- 1) La Commission Electrotechnique Internationale (CEI) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de la CEI). La CEI a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, la CEI – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de la CEI"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec la CEI, participent également aux travaux. La CEI collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
- 2) Les décisions ou accords officiels de la CEI concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de la CEI intéressés sont représentés dans chaque comité d'études.
- 3) Les Publications de la CEI se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de la CEI. Tous les efforts raisonnables sont entrepris afin que la CEI s'assure de l'exactitude du contenu technique de ses publications; la CEI ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
- 4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de la CEI s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de la CEI dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de la CEI et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
- 5) La CEI elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de la CEI. La CEI n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
- 6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
- 7) Aucune responsabilité ne doit être imputée à la CEI, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de la CEI, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de la CEI ou de toute autre Publication de la CEI, ou au crédit qui lui est accordé.
- 8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
- 9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de la CEI peuvent faire l'objet de droits de brevet. La CEI ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets et de ne pas avoir signalé leur existence.

La Norme internationale CEI 62676-2-3 a été établie par le comité d'études 79 de la CEI: Systèmes d'alarme et de sécurité électroniques.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
79/437/FDIS	79/450/RVD

Le rapport de vote indiqué dans le tableau ci-dessus donne toute information sur le vote ayant abouti à l'approbation de cette norme.

Cette publication a été rédigée selon les Directives ISO/CEI, Partie 2.

Une liste de toutes les parties de la série CEI 62676, publiées sous le titre général *Systèmes de vidéosurveillance destinés à être utilisés dans les applications de sécurité*, peut être consultée sur le site web de la CEI.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant la date de stabilité indiquée sur le site web de la CEI sous "<http://webstore.iec.ch>" dans les données relatives à la publication recherchée. A cette date, la publication sera

- reconduite,
- supprimée,
- remplacée par une édition révisée, ou
- amendée.

IMPORTANT – Le logo "colour inside" qui se trouve sur la page de couverture de cette publication indique qu'elle contient des couleurs qui sont considérées comme utiles à une bonne compréhension de son contenu. Les utilisateurs devraient, par conséquent, imprimer cette publication en utilisant une imprimante couleur.

INTRODUCTION

Le Comité d'études 79 de la CEI en charge des systèmes d'alarme et de sécurité électroniques ainsi que de nombreuses organisations gouvernementales, de laboratoires d'essai et de fabricants de matériel ont défini un cadre commun pour la transmission vidéosurveillance afin de permettre l'interopérabilité entre les produits.

La série de normes CEI 62676 dédiées aux systèmes de vidéosurveillance est divisée en 4 parties indépendantes:

- Partie 1: Exigences systèmes
- Partie 2: Protocoles de transmission vidéo
- Partie 3: Interfaces vidéo analogiques et numériques
- Partie 4: Directives d'application (à publier)

Chaque partie propose ses propres articles relatifs au domaine d'application, ainsi qu'aux références, définitions et exigences.

La série CEI 62676-2 comprend 3 sous-parties, respectivement numérotées 2-1, 2-2 et 2-3:

CEI 62676-2-1, *Protocoles de transmission vidéo – Exigences générales*

CEI 62676-2-2, *Protocoles de transmission vidéo – Mise en œuvre de l'interopérabilité IP en fonction des services HTTP et REST*

CEI 62676-2-3, *Protocoles de transmission vidéo – Mise en œuvre de l'interopérabilité IP en fonction des services Web*

Cette troisième sous-partie de la série CEI 62676-2 traite de la mise en œuvre de l'interopérabilité IP sur la base des services Web. Elle repose sur les exigences relatives aux Protocoles de transmission vidéo IP spécifiées dans la CEI 62676-2-1, qui définit les exigences de protocole à satisfaire par une interface de dispositif vidéo IP de haut niveau.

SYSTÈMES DE VIDÉOSURVEILLANCE DESTINÉS À ÊTRE UTILISÉS DANS LES APPLICATIONS DE SÉCURITÉ –

Partie 2-3: Protocoles de transmission vidéo – Mise en œuvre de l'interopérabilité IP en fonction des services Web

1 Domaine d'application

La présente partie 2-3 de la CEI 62676 définit les procédures de communication entre les clients vidéo en réseau et les dispositifs de transmission vidéo sur la base des services Web. Ce nouvel ensemble de spécifications permet de concevoir des systèmes vidéo en réseau avec des dispositifs et des récepteurs de différents fournisseurs utilisant des interfaces communes et bien définies. Ces interfaces couvrent des fonctions telles que la gestion de dispositif, la transmission en temps réel de contenu audio et vidéo, le traitement d'événement, le contrôle panoramique, horizontal/vertical et zoom (PTZ), l'analyse vidéo et le contrôle, la recherche et la lecture des enregistrements.

Les interfaces de gestion et de contrôle définies dans la présente Norme sont décrites sous forme de services Web. La présente Norme internationale contient également les définitions complètes du schéma XML et du langage de description de services web WSDL pour les services vidéo présentés.

Afin d'offrir une interopérabilité complète et prête à l'emploi, la norme définit les procédures de découverte de dispositif. Les mécanismes de découverte de dispositif de la norme reposent sur la spécification WS-Discovery, avec des extensions. Ces extensions ont été introduites afin de satisfaire aux besoins spécifiques de découverte vidéo en réseau.

La présente Norme n'est pas limitée aux fonctions de découverte, de configuration et de contrôle, mais elle définit les formats précis de transmission en continu multimédia et de métadonnées dans les réseaux IP au moyen d'une adaptation des normes de l'IETF. De plus, des extensions de protocole appropriées ont été introduites afin de permettre aux fabricants de vidéo en réseau d'offrir des solutions de transfert vidéo totalement normalisées à leurs clients et intégrateurs.

Un dispositif de vidéotransmission satisfaisant aux exigences de la présente Norme sur la base des services Web selon la spécification de cette partie est déclaré comme compatible avec l'interopérabilité des services Web de la CEI 62676-2.

L'objet de la présente Norme est de réaliser une mise en œuvre vidéo en réseau avec interopérabilité complète, constituée de produits provenant de différents fournisseurs de vidéo en réseau. La présente Norme décrit le modèle vidéo en réseau, les interfaces, les types de données et les schémas d'échange de données. La norme réutilise les normes existantes pertinentes lorsqu'elles sont disponibles et présente de nouvelles spécifications, uniquement lorsque cela est nécessaire, permettant de prendre en charge les exigences spécifiques de la vidéosurveillance en réseau. Il s'agit de la principale spécification de l'Open Network Video Interface Forum (Forum ONVIF). De plus, l'ONVIF a publié les spécifications connexes suivantes:

- ONVIF Schema [voir C.15]
- ONVIF Analytics Service WSDL [voir C.1]
- ONVIF Analytics Device Service [voir C.2]
- ONVIF Device Service WSDL [voir C.4]
- ONVIF DeviceIO Service WSDL [voir C.3]

- ONVIF Display Service WSDL [voir C.5]
- ONVIF Event Service WSDL [voir C.6]
- ONVIF Imaging Service WSDL [voir C.7]
- ONVIF Media Service WSDL [voir C.8]
- ONVIF PTZ Service WSDL [voir C.9]
- ONVIF Receiver Service WSDL [voir C.10]
- ONVIF Recording Service WSDL [voir C.11]
- ONVIF Remote Discovery WSDL [voir C.12]
- ONVIF Replay Service WSDL [voir C.13]
- ONVIF Search Service WSDL [voir C.14]
- ONVIF Topic Namespace XML [voir C.16]

La présente Norme a pour objet de définir le cadre de la spécification de l'ONVIF. Elle comporte les sections suivantes:

Présentation de la spécification: Donne une vue générale des différentes parties de la spécification et leurs relations les unes aux autres.

Interopérabilité des services Web: Offre une brève présentation des services Web et de la base de services Web pour les spécifications de l'ONVIF.

Configuration IP: Définit les exigences de configuration de réseau vidéo IP du réseau ONVIF.

Découverte de dispositif: Décrit comment les dispositifs sont découverts dans les réseaux locaux et à distance.

Gestion de dispositif: Définit les commandes de gestion des transmetteurs vidéo du réseau.

DeviceIO: Définit les commandes de gestion des entrées et sorties physiques.

Affichage: Définit les commandes de gestion des dispositifs d'affichage.

Image et multimédia: Définit les commandes de configuration relatives aux paramètres d'image et multimédia.

Transmission continue en temps réel: Donne les exigences pour la transmission en continu interopérable de vidéo, audio et de métadonnées.

Traitement des événements: Définit comment s'abonner et recevoir des données d'événements vidéo de réseau (notifications).

Contrôle PTZ: Donne les commandes pour le contrôle panoramique, horizontal/vertical et zoom.

Analyse vidéo: Définit le modèle d'analyse, la description d'objet d'analyse et les configurations de règles analytiques de l'ONVIF.

Dispositif d'analyse vidéo: Définit les commandes de gestion d'un dispositif d'analyse vidéo.

Contrôle d'enregistrement: Définit les mécanismes de configuration des enregistrements.

Recherche d'enregistrement et contrôle de lecture: Indique les commandes d'extraction de média enregistrés, y compris les métadonnées.

Section sécurité: Définit les exigences de sécurité du transport et au niveau du message des mises en œuvre satisfaisant à l'ONVIF.

2 Références normatives

Les documents suivants sont cités en référence de manière normative, en intégralité ou en partie, dans le présent document et sont indispensables pour son application. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

UIT-T G.711, *Modulation par impulsions et codage (MIC) des fréquences vocales*

< http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.711-198811-!!PDF-F&type=items >

[X.680] Recommandation UIT-T X.680 (1997) | ISO/CEI 8824-1:2008, *Technologies de l'information – Notation de syntaxe abstraite numéro un (ASN.1): Spécification de la notation de base* (disponible en anglais seulement)

[X.681] Recommandation UIT-T X.681 (1997) | ISO/CEI 8824-2:2008, *Technologies de l'information – Notation de syntaxe abstraite numéro un (ASN.1): Spécification des objets informationnels* (disponible en anglais seulement)

[X.682] Recommandation UIT-T X.682 (1997) | ISO/CEI 8824-3:2008, *Technologies de l'information – Notation de syntaxe abstraite numéro un (ASN.1): Spécification des contraintes* (disponible en anglais seulement)

[X.683] Recommandation UIT-T X.683 (1997) | ISO/CEI 8824-4:2008, *Technologies de l'information – Notation de syntaxe abstraite numéro un (ASN.1): Paramétrage des spécifications de la notation de syntaxe abstraite numéro un* (disponible en anglais seulement)

[X.690] Recommandation UIT-T X.690 (1997) | ISO/CEI 8825-1:2008, *Technologies de l'information – Règles de codage ASN.1: Spécification des règles de codage de base (BER), des règles de codage canoniques (CER) et des règles de codage distinctives (DER)* (disponible en anglais seulement)

NIST FIPS 180-2, *SECURE HASH STANDARD* (disponible en anglais seulement)

<<http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>>

RFC1305, *Network Time Protocol (Version 3), Specification, Implementation and Analysis* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc1305.txt>>

IETF RFC 2131, *Dynamic Host Configuration Protocol* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc2131.txt>>

IETF RFC 2136, *Dynamic Updates in the Domain Name System (DNS UPDATE)* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc2136.txt>>

IETF RFC 2246, *The TLS Protocol Version 1.0* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc2246.txt>>

IETF RFC 2326, *Real Time Streaming Protocol (RTSP)* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc2326.txt>>

IETF RFC 2435, *RTP Payload Format for JPEG-compressed Video* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc2435.txt>>

IETF RFC 2616, *Hypertext Transfer Protocol – HTTP/1.1* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc2616.txt>>

IETF RFC 2617, *HTTP Authentication: Basic and Digest Access Authentication*

<<http://www.ietf.org/rfc/rfc2617.txt>>

IETF RFC 2782, *A DNS RR for specifying the location of services (DNS SRV)*

<<http://www.ietf.org/rfc/rfc2782.txt>>

IETF RFC 3268, *Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc3268.txt>>

IETF RFC 3315, *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc3315.txt>>

IETF RFC 3550, *RTP: A Transport Protocol for Real-Time Applications* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc3550.txt>>

IETF RFC 3551, *RTP Profile for Audio and Video Conferences with Minimal Control* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc3551.txt>>

IETF RFC 3927, *Dynamic Configuration of IPv4 Link-Local Addresses* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc3927.txt>>

IETF RFC 3984, *RTP Payload Format for H.264 Video* (disponible en anglais seulement)

<<http://www.ietf.org/rfc/rfc3984>>

IETF RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax* (disponible en anglais seulement)

[<http://www.ietf.org/rfc/rfc3986.txt>](http://www.ietf.org/rfc/rfc3986.txt)

IETF RFC 4514, *Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names* (disponible en anglais seulement)

[<http://www.ietf.org/rfc/rfc4514.txt>](http://www.ietf.org/rfc/rfc4514.txt)

IETF RFC 4566, *SDP: Session Description Protocol* (disponible en anglais seulement)

[<http://www.ietf.org/rfc/rfc4566.txt>](http://www.ietf.org/rfc/rfc4566.txt)

IETF RFC 4571, *Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport* (disponible en anglais seulement)

[<http://www.ietf.org/rfc/rfc4571.txt>](http://www.ietf.org/rfc/rfc4571.txt)

IETF RFC 4702, *The Dynamic Host Configuration Protocol (DHCP) Client Fully Qualified Domain Name (FQDN) Option* (disponible en anglais seulement)

[<http://www.ietf.org/rfc/rfc4702.txt>](http://www.ietf.org/rfc/rfc4702.txt)

IETF RFC 4861, *Neighbor Discovery for IP version 6 (IPv6)* (disponible en anglais seulement)

[<http://www.ietf.org/rfc/rfc4861.txt>](http://www.ietf.org/rfc/rfc4861.txt)

IETF RFC 4862, *IPv6 Stateless Address Autoconfiguration* (disponible en anglais seulement)

[<http://www.ietf.org/rfc/rfc4862.txt>](http://www.ietf.org/rfc/rfc4862.txt)

W3C SOAP 1.2, Part 1, *Messaging Framework* (disponible en anglais seulement)

[<http://www.w3.org/TR/soap12-part1/>](http://www.w3.org/TR/soap12-part1/)

W3C SOAP Version 1.2 Part 2: Adjuncts (Second Edition) (disponible en anglais seulement)

[<http://www.w3.org/TR/2007/REC-soap12-part2-20070427/>](http://www.w3.org/TR/2007/REC-soap12-part2-20070427/)

OASIS Web Services Base Notification 1.3 (disponible en anglais seulement)

[<http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf>](http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf)

OASIS Web Services Security UsernameToken Profile 1.0 (disponible en anglais seulement)

[<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf>](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf)

W3C XML Schema Part 1: Structures Second Edition

[<http://www.w3.org/TR/xmlschema-1/>](http://www.w3.org/TR/xmlschema-1/)

W3C XML Schema Part 2: Datatypes Second Edition (disponible en anglais seulement)

[<http://www.w3.org/TR/xmlschema-2/>](http://www.w3.org/TR/xmlschema-2/)

W3C XML-binary Optimized Packaging (disponible en anglais seulement)

[<http://www.w3.org/TR/2005/REC-xop10-20050125/>](http://www.w3.org/TR/2005/REC-xop10-20050125/)

W3C XML Path Language (XPath) Version 1.0 (disponible en anglais seulement)

[<http://www.w3.org/TR/xpath/>](http://www.w3.org/TR/xpath/)

IEEE 802.11:2007, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications* (disponible en anglais seulement)

[<http://standards.ieee.org/getieee802/download/802.11-2007.pdf>](http://standards.ieee.org/getieee802/download/802.11-2007.pdf)

IEEE 802.1X, Port-Based Network Access Control (disponible en anglais seulement)

[<http://standards.ieee.org/getieee802/download/802.1X-2004.pdf>](http://standards.ieee.org/getieee802/download/802.1X-2004.pdf)

[UDDI API ver2] “*UDDI Version 2.04 API Specification UDDI Committee Specification*, 19 July 2002”, OASIS standard, 19 July 2002 (disponible en anglais seulement)

[<http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf>](http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf)

[UDDI Data Structure ver2] “*UDDI Version 2.03 Data Structure Reference UDDI Committee Specification*”, OASIS standard, 19 July 2002 (disponible en anglais seulement)

[<http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf>](http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf)

[WS-X.509Token] “*Web Services Security X.509 Certificate Token Profile 1.1*”, OASIS Standard, 1 February 2006. (disponible en anglais seulement)

[<http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf>](http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf)

3 Termes, définitions et abréviations

3.1 Termes et définitions

Pour les besoins du présent document, les termes et définitions suivants s'appliquent.

3.1.1

réseau ad-hoc

ensemble de services de base indépendants

[SOURCE: IEEE 802.11:2007]

3.1.2

ensemble de services de base

ensemble de stations IEEE 802.11 qui participent à un réseau commun

[SOURCE: IEEE 802.11:2007]

3.1.3

fonctionnalité

capacité d'un dispositif permettant à un client de demander les services offerts par un dispositif

3.1.4

entité de configuration

composant multimédia abstrait de dispositif vidéo en réseau utilisé pour générer un flux multimédia sur le réseau

Note 1 à l'article: Le flux multimédia est un flux vidéo et/ou audio.

3.1.5

plan de contrôle

plan composé de fonctions de contrôle multimédia

Note 1 à l'article: Les commandes de contrôle de dispositif, de configuration multimédia et PTZ sont des fonctions de contrôle multimédia.

3.1.6

PTZ numérique

fonction qui diminue ou coupe une image pour régler sa position et son rapport

3.1.7

entrée/sortie

ES

connecteurs externes

EXEMPLES: Entrées numériques, ports de relais et entrées/sorties vidéo/audio.

3.1.8

présentation

disposition des zones d'affichage (écrans) sur un moniteur

3.1.9

plan multimédia

plan composé de flux multimédia

Note 1 à l'article: Le flux multimédia est un flux vidéo, audio et de métadonnées.

3.1.10

profil multimédia

ensemble de configurations définissant un sous-ensemble de source audio/vidéo, de paramètres de codeur et de décodeur, y compris les paramètres PTZ et d'analyse

3.1.11

métadonnées

ensemble des données de transmission en continu, sauf la vidéo et l'audio, comprenant les résultats d'analyse vidéo, les données de position PTZ et d'autres métadonnées

Note 1 à l'article: Les autres métadonnées comprennent des données textuelles provenant d'applications POS.

3.1.12

transmetteur vidéo réseau

serveur vidéo en réseau qui envoie des données multimédia à un client sur un réseau IP

Note 1 à l'article: Un serveur vidéo en réseau est une caméra de réseau IP ou dispositif de codage.

3.1.13**affichage vidéo en réseau**

récepteur vidéo en réseau qui reçoit les données multimédia, provenant par exemple d'un NVD, sur un réseau IP

EXEMPLE: Un moniteur vidéo de réseau IP.

3.1.14**stockage vidéo en réseau**

dispositif qui enregistre le média et les métadonnées reçus d'un dispositif de transmission en continu (un NVT, par exemple) sur un réseau IP vers un support de stockage définitif

Note 1 à l'article: Le NVS permet aux clients de consulter les données stockées.

3.1.15**analyses vidéo en réseau**

dispositif qui procède à une analyse des données reçues d'un dispositif de transmission en continu (un NVT, par exemple) ou d'un dispositif de stockage (un NVS, par exemple)

3.1.16**panneau**

zone de l'affichage physique

3.1.17**normes de cryptographie à clé publique**

groupe de normes de cryptographie à clé publique inventé et publié par RSA Security

3.1.18**clé prépartagée**

clé statique distribuée au dispositif

3.1.19**nœud PTZ**

entité PTZ de niveau inférieur avec mappage pour le dispositif PTZ et ses fonctionnalités

3.1.20**pullpoint**

ressource permettant d'extraire des messages

Note 1 à l'article: Lors de l'extraction des messages, les notifications ne sont pas bloquées par les pare-feu.

3.1.21**enregistrement**

le média (le cas échéant) et les métadonnées actuellement stockés sur le NVS à partir d'une seule source de données

Note 1 à l'article: Un enregistrement est composé d'une ou de plusieurs pistes. Il peut s'agir de plusieurs pistes de même type (deux pistes vidéo enregistrées en parallèle avec différents paramètres, par exemple).

3.1.22**événement d'enregistrement**

événement associé à un enregistrement, représenté par un message de notification dans les API

3.1.23**travail d'enregistrement**

travail permettant de procéder au transfert des données entre une source de données et un enregistrement particulier à l'aide d'une configuration particulière

3.1.24

proxy de découverte à distance

DP à distance

discovery proxy à distance

proxy de découverte (DP) à distance qui permet à un NVT de s'enregistrer sur le DP à distance et sur le NVC pour trouver les NVT enregistrés par son intermédiaire, même si le NVC et le NVT résident dans des domaines administratifs différents du réseau

3.1.25

description de scène

sortie de métadonnées par analyse vidéo décrivant l'emplacement et le comportement d'un objet

3.1.26

route

voie de données individuelle composée de données vidéo, de données audio et de métadonnées

3.1.27

analyse vidéo

algorithmes ou programmes utilisés pour analyser les données vidéo et pour générer des données décrivant l'emplacement et le comportement d'objets

3.1.28

accès protégé WiFi

programme de certification créé par la Wi-Fi Alliance afin d'indiquer la conformité au protocole de sécurité couvert par le programme

3.2 Abréviations

AAC	Codage audio avancé (Advanced Audio Coding)
API	Interface de programmation d'applications (Application Programming Interface)
ASCII	Codage en caractères de 7 bits
ASN	Notation de syntaxe abstraite (Abstract Syntax Notation)
AVP	Profil audio/vidéo
AVPF	Profil audio/vidéo pour retour RTCP (Audio/Video Profile for rtcp Feedback)
BLC	Compensation de lumière noire (Back Light Compensation)
BSSID	Identification réseau sans fil de base (Basic Service Set Identifier)
CA	Autorité de certification (Certificate Authority)
CBC	Chaînage de bloc de chiffrement
CCMP	Méthode de chiffrement par bloc (Counter mode with Cipher-block chaining Message authentication code Protocol)
DER	Règles de codage distinctives (Distinguished Encoding Rules)
DHCP	Protocole DHCP (Dynamic Host Configuration Protocol)
DHT	Table de Huffman de définition (Define Huffman Table)
DM	Gestion de dispositif (Device Management)
DNS	Serveur de noms de domaine (Domain Name Server)
DQT	Table de quantification de définition (Define Quantization Table)
DP	Proxy de découverte (Discovery Proxy)
DRI	Intervalle de redémarrage de définition (Define Restart Interval)
EAP	Protocole d'authentification EAP (Extensible Authentication Protocol)
EOI	Fin d'image (End Of Image)
FIPS	Norme de traitement des informations fédérales (Federal Information Processing Standard)
FOV	Champ de vision (Field Of View)
GW	Passerelle (Gateway)
HMAC	Code d'authentification de message par hachage (Hash-based Message Authentication Code)
HTTP	Protocole HTTP (HyperText Transfer Protocol)
HTTPS	Protocole HTTPS (HyperText Transfer Protocol over Secure Socket Layer)
ES; E/S	Entrée/Sortie
IP	Protocole IP (Internet Protocol)
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
Ir	Infrarouge
JFIF	Format d'échange de fichiers JPEG (JPEG File Interchange Format)
JPEG	Groupe JPEG (Joint Photographic Expert Group)
LAN	Réseau local (Local Area Network)
MPEG-4	Groupe MPEG – 4 (Moving Picture Experts Group - 4)
MTOM	Mécanisme d'optimisation de la transmission de message (Message Transmission Optimization Mechanism)
NAT	Traduction d'adresse de réseau (Network Address Translation)
NFC	Communication en champ proche (Near Field Communication)

NTP	Protocole NTP (Network Time Protocol)
NVA	Dispositif d'analyse vidéo en réseau (Network Video Analytics)
NVC	Client vidéo réseau (Network Video Client)
NVD	Affichage vidéo en réseau (Network Video Display)
NVT	Transmetteur vidéo réseau (Network Video Transmitter)
NVS	Dispositif de stockage vidéo en réseau (Network Video Storage Device)
OASIS	Organisme de normalisation Groupe OASIS (Organization for the Advancement of Structured Information Standards)
ONVIF	Forum ONVIF (Open Network Video Interface Forum)
PLI	Indicateur de perte d'image (Picture Loss Indication)
POSIX	Interface de système d'exploitation portable
PKCS	Normes de cryptographie à clé publique (Public Key Cryptography Standards)
PSK	Clé prépartagée (Pre Shared Key)
PTZ	Panoramique/horizontal/vertical/zoom (Pan/Tilt/Zoom)
QVGA	Quarter Video Graphics Array (320 x 240 pixels)
REL	Langage REL (Rights Expression Language)
RSA	Rivest, Shamir et Adleman
RTCP	Protocole de contrôle en temps réel, protocole RTCP (RTP Control Protocol)
RTP	Protocole de transmission en temps réel, protocole RTP (Real-Time Transport Protocol)
RTSP	Protocole de flux en temps réel, protocole RTSP (Real Time Streaming Protocol)
SAML	Langage SAML (Security Assertion Markup Language)
SDP	Protocole de description de session, protocole SDP (Session Description Protocol)
SHA	Algorithme de hachage sécurisé (Secure Hash Algorithm)
SOAP	Protocole SOAP (Simple Object Access Protocol)
SOI	Début d'image (Start Of Image)
SOF	Début de trame (Start Of Frame)
SOS	Début de balayage (Start Of Scan)
SR	Rapport expéditeur (Sender Report)
SSID	Identifiant d'ensemble de services (Service Set ID)
TCP	Protocole TCP (Transmission Control Protocol)
TLS	Sécurité de couche transport (Transport Layer Security)
TKIP	Protocole de communication pour protection réseau sans fil (Temporal Key Integrity Protocol)
TTL	Durée de vie (Time-To-Live)
UDDI	Description, découverte et intégration universelles (Universal Description, Discovery and Integration)
UDP	Protocole UDP (User Datagram Protocol)
URI	Identifiant unique de ressource (Uniform Resource Identifier)
URN	Nom URN (Uniform Resource Name, nom sur Internet)
USB	Bus série universel (Universal Serial Bus)
UDDI	Description, découverte et intégration universelles (Universal Description, Discovery and Integration)

TUC	Temps universel coordonné
UTF	Format de transformation Unicode (Unicode Transformation Format)
UUID	Identifiant unique universel (Universally Unique Identifier)
WDR	Plage dynamique large (Wide Dynamic Range)
WPA	Accès protégé WiFi
WS	Services web (Web Services)
WSDL	Langage de description de services web (Web Services Description Language)
WS-I	Interopérabilité des services web (Web Services Interoperability)
XML	Langage XML (eXtensible Markup Language)
XPath	Langage de chemin XML

4 Présentation

La présente Norme repose sur des cas d'utilisation de vidéo en réseau couvrant des scénarios de réseaux locaux comme étendus. La spécification commence par un ensemble de fonctions d'interface pour la configuration et le fonctionnement de dispositifs vidéo en réseau en définissant leurs interfaces côté serveur. L'ensemble de dispositifs vidéo en réseau est composé du NVT (Transmetteur vidéo réseau - Network Video Transmitter), du NVD (Affichage vidéo en réseau - Network Video Display), du NVS (Dispositif de stockage vidéo en réseau - Network Video Storage) et du NVA (Dispositif d'analyse vidéo en réseau - Network Video Analytics). Le cadre d'application est conçu pour être étendu et amélioré dans les versions ultérieures.

Le cadre d'application couvre les procédures allant du déploiement du dispositif vidéo en réseau et de la phase de configuration à la phase de transmission en temps réel pour ces différents scénarios de vidéo en réseau.

La présente Norme aborde la découverte de dispositif, la configuration de dispositif, les événements, le contrôle PTZ, les analyses vidéo et les fonctions de transmission en temps réel de vidéo en direct, ainsi que la recherche, la lecture et l'enregistrement de vidéos enregistrées.

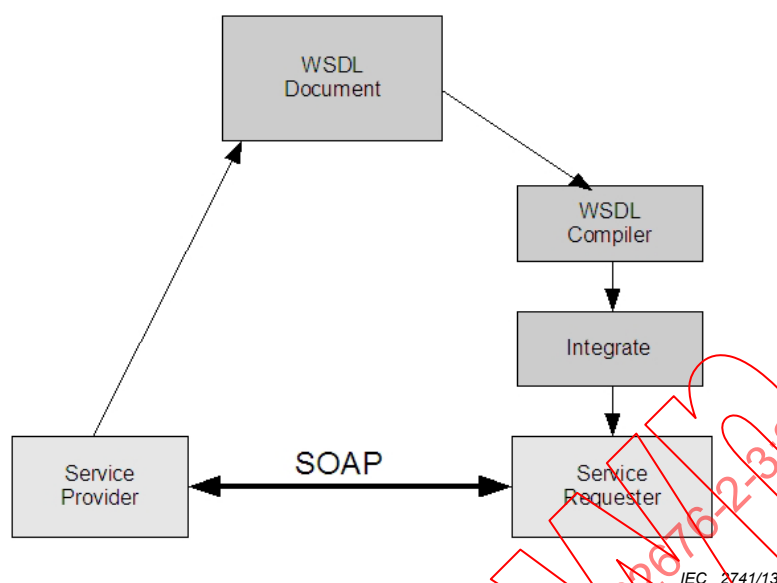
Tous les services partagent un schéma XML commun et tous les types de données sont fournis dans [ONVIF Schema]. Les différents services sont définis dans les sections et documents WSDL de service respectifs.

4.1 Services web

Le terme Services Web est le nom d'une méthode normalisée d'intégration d'applications s'appuyant sur des normes de services Web ouvertes et indépendantes des plateformes (XML SOAP 1.2, [Partie 1] et WSDL 1.1 sur un réseau IP, par exemple). XML est utilisé comme syntaxe de description des données, SOAP est utilisé pour le transfert de messages et WSDL est utilisé pour décrire les services.

Ce cadre d'application repose sur les normes des services Web. Tous les services de configuration définis dans la norme sont exprimés sous forme d'opérations de services Web, et sont définis en WSDL, avec HTTP comme mécanisme de transport sous-jacent.

La Figure 1 présente les principes de base du développement basé sur les services Web. Le fournisseur de services (dispositif) met en œuvre le ou les services ONVIF.



Légende

Anglais	Français
WSDL document	Document WSDL
WSDL Compiler	Compilateur WSDL
Integrate	Intégration
Service Requester	Demandeur de services
Service Provider	Fournisseur de services

Figure 1 – Principes de développement des services Web

Le service est décrit au moyen du WSDL basé sur XML. Ensuite, le WSDL est utilisé comme base de mise en œuvre/d'intégration du demandeur de services (client). L'intégration côté client est simplifiée par l'utilisation d'outils de compilation WSDL qui génèrent un code spécifique à la plateforme, et qui peut être utilisé par le développeur côté client pour intégrer le service Web dans une application.

Le fournisseur et le demandeur de services Web peuvent communiquer à l'aide du protocole d'échange de messages SOAP. SOAP est un protocole de messagerie léger basé sur XML utilisé pour coder les informations dans une requête de service Web et dans un message de réponse avant de les envoyer sur un réseau. Les messages SOAP sont indépendants du système d'exploitation ou du protocole et peuvent être transportés au moyen de divers protocoles Internet. La présente Norme ONVIF définit les protocoles de transport conformes des messages SOAP pour les services Web décrits.

La section de présentation des services Web définit les différents services ONVIF, la syntaxe de définition des commandes dans la spécification, les principes de traitement des erreurs et les mécanismes de sécurités adoptés pour les services Web.

Afin de garantir l'interopérabilité, tous les services définis suivent les recommandations de profil de base 2.0 WS-I et utilisent le schéma littéral/enveloppant.

4.2 Configuration IP

La section Configuration IP définit les exigences et recommandations en matière de conformité de la configuration IP. La configuration IP comprend:

- la capacité de communication du réseau IP;

- la configuration IP statique;
- la configuration IP dynamique.

4.3 Découverte de dispositif

Les interfaces de configuration définies dans la présente Norme sont des interfaces de services Web reposant sur la norme WS-Discovery. L'utilisation de la présente Norme permet de réutiliser un cadre d'application de découverte de services Web adapté existant plutôt que d'avoir à définir un service ou un adressage de service complètement nouveau.

La présente Norme présente un comportement de découverte spécifique adapté à la vidéosurveillance. Par exemple, une découverte pleinement interopérable nécessite une définition de service et des critères de recherche de service bien définis. Dans ce but, la spécification couvre le type de dispositif et les définitions des domaines d'application.

Une découverte réussie fournit l'adresse de service du dispositif. Dès que le client obtient l'adresse de service du dispositif, il peut recevoir des informations détaillées sur le dispositif par l'intermédiaire du service de dispositif (voir 4.5 ci-dessous).

Outre le protocole de découverte de services Web normalisés, la présente norme prend en charge les proxies de découverte à distance afin de rechercher les dispositifs enregistrés par leur intermédiaire, même si le client et le dispositif résident dans des domaines de réseau administratif différents.

4.4 Types de dispositifs

Le type de dispositif indique la fonction principale d'un dispositif. La présente norme précise l'ensemble de types de dispositif suivants:

- transmetteur vidéo réseau (NVT);
- affichage vidéo en réseau (NVD);
- stockage vidéo en réseau (NVS);
- analyses vidéo en réseau (NVA).

À chaque type de dispositif correspond un nombre de services obligatoires définis en 5.1.2. Un dispositif peut prendre en charge d'autres services facultatifs, signalant leur disponibilité grâce à la découverte de dispositif.

4.5 Gestion de dispositif

Les fonctions de gestion de dispositif sont traitées par l'intermédiaire du service de dispositif. Le service de dispositif est le point d'entrée vers tous les autres services proposés par un dispositif. Le WSDL pour le service de dispositif est fourni dans le fichier WSDL de gestion de dispositif. Les interfaces de gestion de dispositif sont constituées des sous-catégories suivantes:

- fonctionnalités;
- réseau;
- système;
- sécurité.

4.5.1 Fonctionnalités

Les commandes de fonctionnalité permettent à un client de demander les services offerts par un dispositif et de déterminer les services généraux et spécifiques au fournisseur offerts par le dispositif. Les fonctionnalités sont structurées sous forme de différents services du dispositif et sont en outre divisées en catégories secondaires (le cas échéant), comme suit:

- analyses,
- dispositif,
 - fonctionnalités,
 - réseau,
 - système,
 - E/S,
 - sécurité,
- événement,
- imagerie,
- média,
- PTZ,
- dispositif ES,
- affichage,
- enregistrement,
- recherche,
- lecture,
- dispositif d'analyse.

Les fonctionnalités correspondant aux différentes catégories indiquent les commandes et les paramètres disponibles pour le service particulier ou la sous-catégorie de service particulière.

4.5.2 Réseau

L'ensemble de commandes réseau qui suit permet une gestion normalisée des fonctions:

- obtenir et définir le nom d'hôte;
- obtenir et définir les configurations DNS;
- obtenir et définir les configurations NTP;
- obtenir et définir le DNS dynamique;
- obtenir et définir les configurations d'interface de réseau;
- activer/désactiver et énumérer les protocoles de réseau;
- obtenir et définir la passerelle par défaut;
- obtenir et définir la configuration zéro;
- obtenir, définir, ajouter et supprimer un filtre d'adresses IP.

4.5.3 Système

Les commandes système permettent de gérer les paramètres systèmes suivants du dispositif:

- obtenir des informations sur le dispositif;
- effectuer des sauvegardes du système;
- obtenir et régler la date et l'heure du système;
- réinitialisation par défaut;
- mettre à niveau le micrologiciel;
- obtenir le journal du système;
- obtenir des données de diagnostic sur le dispositif (information d'assistance);
- redémarrer;

- obtenir et définir les paramètres de découverte du dispositif.

4.5.4 Obtention des informations sur le système

Les informations sur le système (journaux système, informations de support spécifiques au fournisseur et images de sauvegarde de la configuration, par exemple) peuvent être obtenues à l'aide de MTOM ou HTTP.

La méthode MTOM est prise en charge par les commandes GetSystemLog, GetSystemSupportInformation et GetSystemBackup. La méthode HTTP est prise en charge par la commande GetSystemUri; cela permet d'extraire les URI à partir desquels les fichiers peuvent être téléchargés à l'aide d'une opération HTTP GET.

4.5.5 Mise à niveau de micrologiciel

Deux mécanismes permettent de mettre à niveau le micrologiciel sur un dispositif. Le premier utilise la commande UpgradeSystemFirmware pour envoyer la nouvelle image de micrologiciel à l'aide de MTOM.

Le deuxième est un processus en deux étapes: en premier lieu, le client envoie la commande StartFirmwareUpgrade pour demander au dispositif de se préparer à la mise à niveau, puis envoie l'image du micrologiciel à l'aide de HTTP POST.

La méthode HTTP s'adresse aux dispositifs dont les ressources sont limitées et qui ne peuvent pas recevoir l'image du nouveau micrologiciel dans leur état de fonctionnement.

4.5.6 Restauration du système

La fonctionnalité de restauration du système permet de restaurer la configuration d'un dispositif à partir d'une image de sauvegarde. Deux mécanismes sont également proposés. Le premier utilise la commande RestoreSystem pour envoyer l'image de sauvegarde à l'aide de MTOM. Le deuxième utilise la commande StartSystemRestore suivie d'une opération HTTP POST pour envoyer l'image de sauvegarde.

4.5.7 Sécurité

Les opérations de sécurité suivantes permettent de gérer les configurations de sécurité du dispositif:

- obtenir et définir la politique de sécurité d'accès;
- traiter les justificatifs et les paramètres utilisateur;
- traiter les certificats de serveur HTTPS;
- activer/désactiver l'authentification client HTTPS;
- fonctions de génération de clé et de téléchargement de certificats;
- traiter les certificats de supplication IEEE 802.1X;
- traiter les certificats de CA IEEE 802.1X;
- configuration IEEE 802.1X.

4.6 DeviceIO

Le service DeviceIO propose des commandes permettant d'extraire et de configurer les paramètres d'entrées et de sorties physiques d'un dispositif.

Le service DeviceIO prend en charge la configuration des interfaces de dispositif suivantes:

- VideoOutputs;
- VideoSources;

- AudioOutputs;
- AudioSources;
- RelayOutputs.

Les commandes suivantes répertorient les interfaces existantes:

- GetVideoOutputs – obtient toutes les sorties vidéo existantes du dispositif;
- GetVideoSources – obtient toutes les sources vidéo existantes du dispositif;
- GetAudioOutputs – obtient toutes les sorties audio existantes du dispositif;
- GetAudioSources – obtient toutes les sources audio existantes du dispositif;
- GetRelayOutputs – obtient toutes les sorties de relais existantes du dispositif.

Pour VideoOutputs, VideoSources, AudioOutputs et AudioSources, les commandes suivantes sont prises en charge:

- *Set<nom_dispositif>Configuration* – modifie la configuration d'une interface spécifique;
- *Get<nom_dispositif>Configuration* – permet d'obtenir la configuration d'une interface spécifique;
- *Get<nom_dispositif>ConfigurationOptions* – permet d'obtenir les valeurs de propriété prises en charge d'une interface spécifique.

RelayOutputs prend en charge les commandes suivantes:

- SetRelayOutputSettings – modifie la configuration d'une sortie de relais;
- SetRelayOutputState – définit le statut logique.

Le WSDL du service DeviceIO est spécifié en [C.3].

4.7 Configuration d'imagerie

Le service d'imagerie fournit les données de configuration et de contrôle pour les propriétés spécifiques d'imagerie. WSDL fait partie du cadre d'application et est fourni dans le fichier WSDL d'imagerie.

Le service comprend les opérations suivantes:

- obtenir et définir les configurations d'image (temps d'exposition, gain et équilibre des blancs, par exemple);
- obtenir les options de configuration d'image (plages valides pour les paramètres d'image);
- déplacer l'objectif de mise au point;
- stopper le déplacement en cours de mise au point;
- obtenir la position courante et modifier le statut de mise au point.

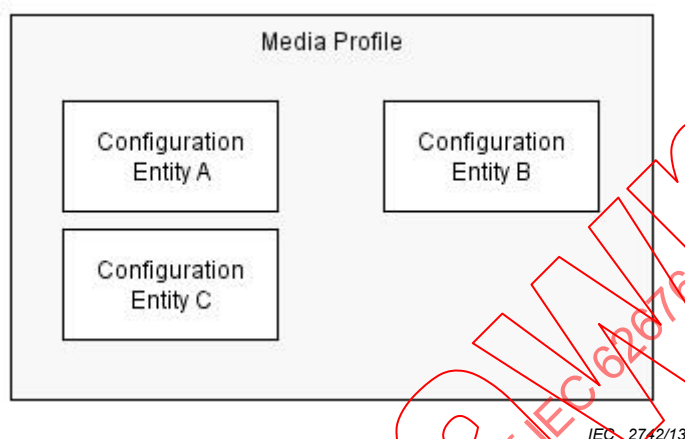
4.8 Configuration multimédia

4.8.1 Généralités

Les configurations multimédia sont traitées par l'intermédiaire du service multimédia. Les configurations multimédia servent à déterminer les propriétés de transmission en continu des flux multimédia demandés tels que définis dans la présente norme. Le dispositif permet la configuration multimédia par l'intermédiaire du service multimédia. Le WSDL du service multimédia est fourni dans le fichier WSDL multimédia.

4.8.2 Profils multimédia

Les configurations de transmission vidéo et audio en temps réel sont contrôlées à l'aide de profils multimédia. Un profil multimédia mappe une source vidéo et/ou audio à un codeur vidéo et/ou audio, des configurations PTZ et analytiques (voir Figure 2). Le NVT présente différents profils disponibles en fonction de ses fonctionnalités (l'ensemble de profils disponibles peut changer de manière dynamique).



Légende

Anglais	Français
Media profile	Profil multimédia
Configuration Entity A	Entité de configuration A
Configuration Entity B	Entité de configuration B
Configuration Entity C	Entité de configuration C

Figure 2 – Profil multimédia

Un dispositif doté d'un service de configuration multimédia offre au moins un profil multimédia au démarrage. Un dispositif peut offrir des profils "prêts à l'emploi" pour les configurations multimédia les plus courantes qu'offre le dispositif.

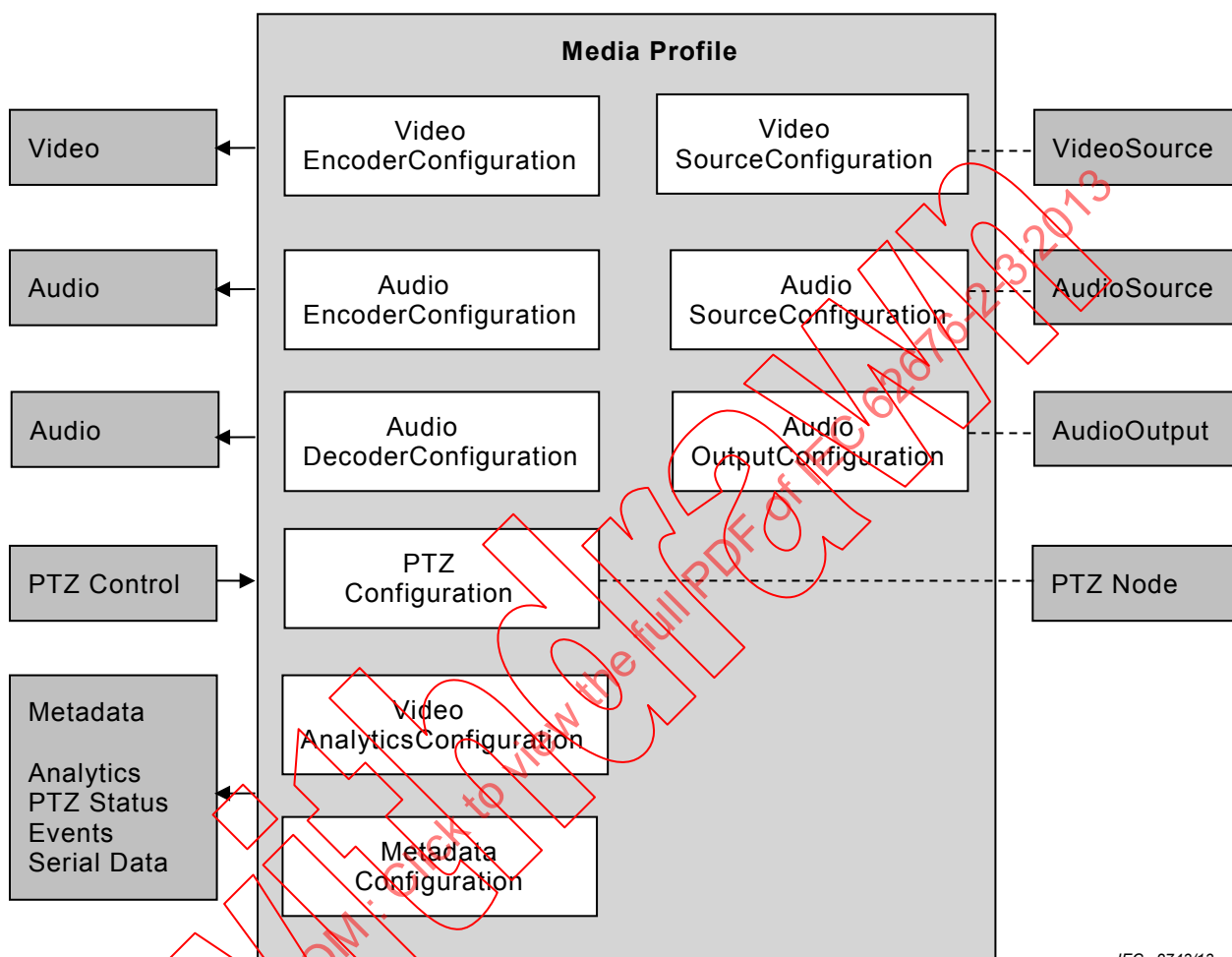
Le profil contient un attribut "fixe" qui indique si un profil peut être supprimé ou pas. Le caractère fixe ou pas d'un profil est défini par le NVT.

Un profil est composé d'un ensemble d'*entités de configuration* interconnectées. Les configurations sont fournies par le NVT et peuvent être statiques ou créées de manière dynamique par le NVT. Par exemple, les configurations dynamiques peuvent être créées par le NVT en fonction des ressources de codage disponibles actuelles. Une entité de configuration est d'un des types suivants:

- configuration de source vidéo;
- configuration de source audio;
- configuration de codeur vidéo;
- configuration de codeur audio;
- configuration PTZ;
- configuration d'analyse vidéo;
- configuration de métadonnées;
- configuration de sortie audio;
- configuration de décodeur audio.

Un profil est composé de l'ensemble ou d'un sous-ensemble de ces entités de configuration. Suivant les fonctionnalités du NVT, une entité de configuration particulière peut faire partie ou non d'un profil. Par exemple, un profil avec une source audio et une configuration de codeur audio peut exister uniquement dans un dispositif avec support audio.

Un exemple de configuration de profil complète est illustré à la Figure 3.



IEC 2743/13

Légende

Anglais	Français
Media profile	Profil multimédia
Video	Vidéo
Video EncoderConfiguration	Vidéo EncoderConfiguration
Video SourceConfiguration	Vidéo SourceConfiguration
PTZ Control	Contrôle PTZ
PTZ Configuration	Configuration PTZ
PTZ Node	Nœud PTZ
Metadata Analytics PTZ Status Events Serial Data	Métadonnées Analytics Statut PTZ Événements Données en série
Video AnalyticsConfiguration	Vidéo AnalyticsConfiguration
Metadata Configuration	Métadonnées Configuration

Figure 3 – Configuration de profil complète

Un profil multimédia décrit les éléments à présenter au client dans un flux multimédia, et comment les présenter, ainsi que la manière de traiter les entrées PTZ et analytiques.

Les commandes qui suivent répertorient les sources existantes:

- *GetVideoSources* – obtient toutes les sources vidéo existantes du dispositif;
- *GetAudioSources* – obtient toutes les sources audio existantes du dispositif;
- *GetAudioOutputs* – obtient toutes les sorties audio existantes du dispositif.

Les commandes qui suivent gèrent les profils multimédia:

- *CreateProfile* – crée un profil multimédia;
- *GetProfiles* – obtient tous les profils multimédia existants;
- *GetProfile* – obtient un profil multimédia spécifique;
- *DeleteProfile* – supprime un profil multimédia spécifique;
- *Add<entité_configuration>* – ajoute une entité de configuration spécifique au profil multimédia;
- *Remove<entité_configuration>* – supprime une entité de configuration spécifique d'un profil multimédia.

Les commandes qui suivent gèrent les entités de configuration:

- *Get<entité_configuration>Options* – obtient les valeurs de propriété valides pour une entité de configuration spécifique;
- *Set<entité_configuration>* – définit la configuration d'une entité de configuration;
- *Get<entité_configuration>s* – obtient toutes les entités de configuration existantes du type;
- *Get<entité_configuration>* – obtient une entité de configuration spécifique;
- *GetCompatible<entité_configuration>s* – obtient toutes les entités de configuration compatibles avec un profil multimédia spécifique,

où *<entité_configuration>* est le type d'entité de configuration. Par exemple, la commande complète permettant d'obtenir une configuration de codeur vidéo est:

GetVideoEncoderConfiguration.

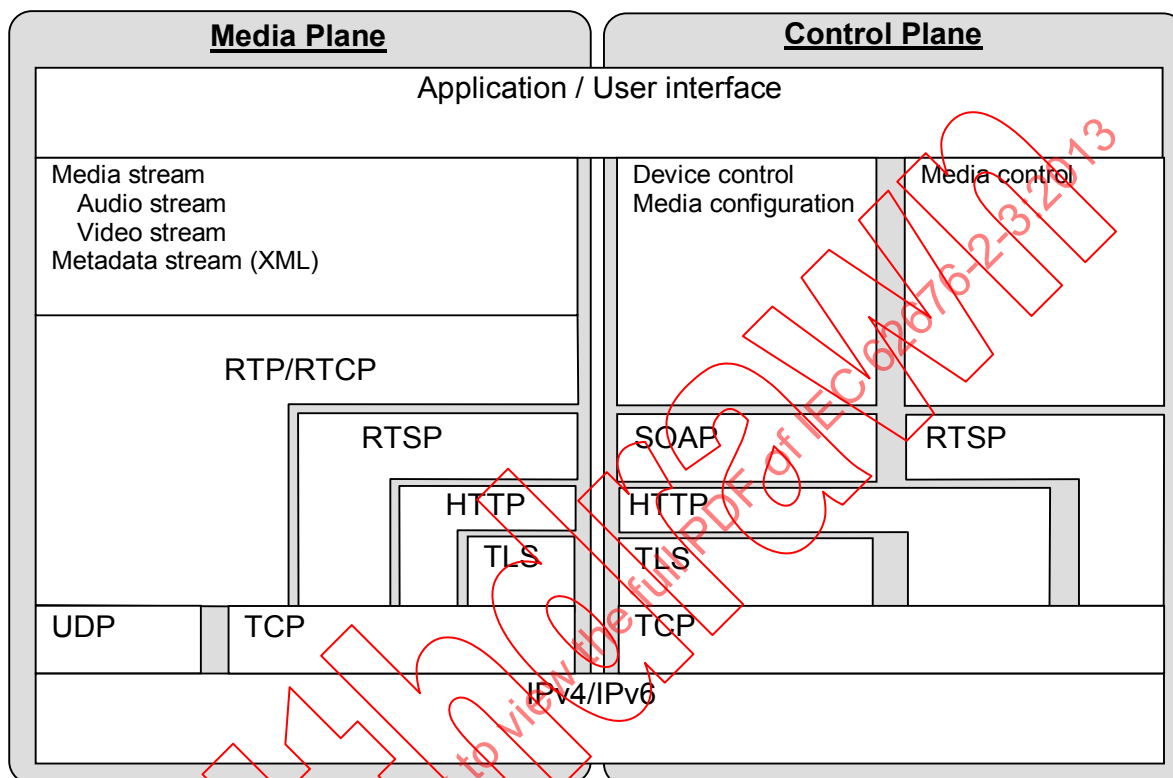
Les commandes qui suivent initient et manipulent un flux vidéo/audio:

- *GetStreamUri* – demande un URI de flux RTSP ou HTTP valide pour un profil multimédia et un protocole spécifiques;
- *StartMulticastStreaming* – commence la transmission continue en multidiffusion à l'aide d'un profil multimédia spécifié;
- *StopMulticastStreaming* – arrête un flux de multidiffusion;
- *SetSynchronizationPoint* – insère un point de synchronisation (trame I, etc.) dans les flux actifs;
- *GetSnapshotUri* – demande un URI HTTP valide pour un profil multimédia spécifique qui peut être utilisé pour obtenir un instantané JPEG.

Voir l'Article 5 pour des exemples d'utilisation de profils dans une mise en œuvre client.

4.9 Transmission en temps réel

La présente Norme définit les options et les formats de transmission multimédia. Une distinction est faite entre *plan multimédia* et *plan de contrôle*, comme illustré à la Figure 4. Un ensemble d'options de transmission multimédia (audio, vidéo et métadonnées), toutes basées sur RTP [RFC 3550], est décrit afin d'assurer l'interopérabilité des services de transmission multimédia.



IEC 2744/13

Légende

Anglais	Français
Media plane	Plan multimédia
Control Plane	Plan de contrôle
Media stream	Flux multimédia
Audio stream	Flux audio
Video stream	Flux vidéo
Metadata stream	Flux de métadonnées
Device control	Contrôle de dispositif
Media configuration	Configuration multimédia
Media control	Contrôle multimédia
Application / User interface	Application/Interface utilisateur

Figure 4 – Structure de couche

Le format de transmission de métadonnées permet une transmission en temps réel bien définie des données analytiques, de statut PTZ et de notification.

La configuration multimédia est exécutée via SOAP/HTTP et est couverte par le service de configuration multimédia (voir 4.8).

Le contrôle multimédia est assuré sur RTSP (voir RFC 2326). La présente Norme utilise le profilage RTP, RTCP et RTSP, ainsi que JPEG sur des extensions RTP et des mécanismes de contrôle multidiffusion.

La norme introduit des extensions à la norme RTSP pour permettre les connexions de transmission en continu bidirectionnelles.

Les configurations de transmission en continu des codecs vidéo suivants sont proposées:

- JPEG (via RTP), voir 12.1.3;
- MPEG-4, Simple Profile (SP) [ISO/CEI 14496-2];
- MPEG-4, Advanced Simple Profile (ASP) [ISO/CEI 14496-2];
- H.264, baseline [ISO/CEI 14496-10];
- H.264, main [ISO/CEI 14496-10];
- H.264, extended [ISO/CEI 14496-10];
- H.264, high [ISO/CEI 14496-10];

et pour les codecs audio suivants:

- G.711 [UIT-T G.711];
- G.726 [UIT-T G.726];
- AAC [ISO/CEI 14496-3].

4.10 Traitement des événements

Le traitement des événements repose sur les spécifications OASIS WS-BaseNotification et WS-Topics. Ces spécifications permettent de réutiliser un cadre de notification riche sans avoir à redéfinir les principes du traitement des événements, ses formats de base et ses schémas de communication.

Selon WS-BaseNotification, la traversée de pare-feu est traitée par l'intermédiaire d'un schéma de notification *PullPoint*. Ce schéma ne permet toutefois pas la notification en temps réel. La présente norme définit donc un schéma de communication *PullPoint* et une interface de service alternatifs. Le schéma *PullPoint* permet à un client résidant derrière un pare-feu de recevoir des notifications en temps réel tout en utilisant le cadre d'application WS-BaseNotification.

Un événement totalement normalisé requiert des notifications normalisées. Toutefois, les sujets de notification dépendent dans une large mesure des besoins de l'application. La présente norme définit un ensemble de rubriques de notification de base; il est recommandé qu'un dispositif les prenne en charge (voir Annexe A). De plus, pour certains services, la présente norme étend les rubriques de notification de base sans événement obligatoire.

Le WSDL du service d'événement comprenant les extensions est fourni dans le fichier WSDL d'événement.

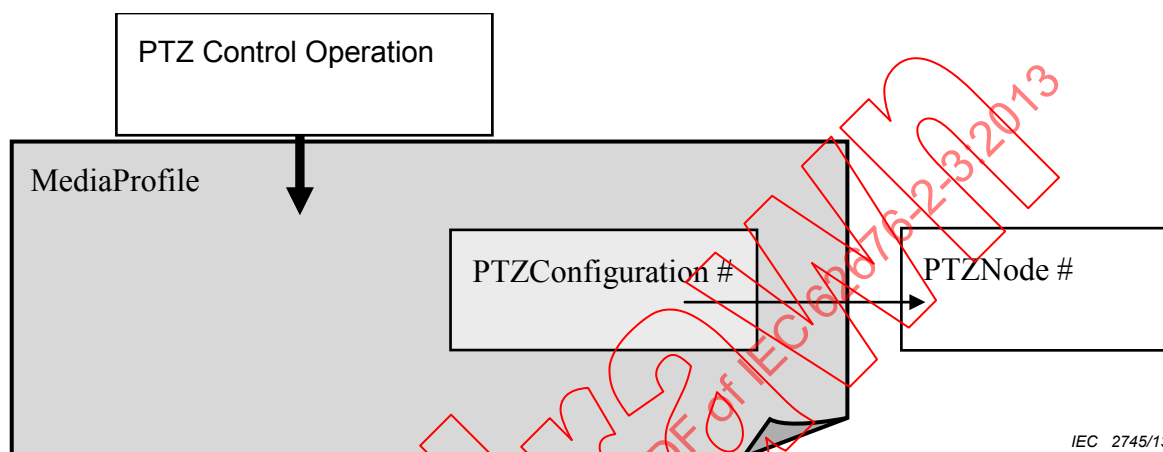
4.11 Contrôle PTZ

Le service PTZ permet de contrôler un dispositif de codage vidéo avec panoramique/horizontal/vertical/zoom (PTZ). Le WSDL du service PTZ est fourni dans le fichier WSDL PTZ.

Le principe du contrôle PTZ suit le modèle *MediaProfile* (voir 4.8) et est composé de trois blocs principaux:

- Nœud PTZ – Entité PTZ de niveau inférieur mappée avec le dispositif PTZ et ses fonctionnalités;
- Configuration PTZ – Contient la configuration PTZ pour un nœud PTZ spécifique;
- Opération de contrôle PTZ – Opérations de préréglage et de statut PTZ.

La relation entre les trois éléments mentionnés ci-dessus est décrite en Figure 5.



IEC 2745/13

Légende

Anglais	Français
PTZ Control Operation	Opération de contrôle PTZ
Media profile	Profil multimédia
PTZ configuration #	Configuration PTZ
PTZ node #	N° de nœud PTZ

Figure 5 – Modèle de contrôle PTZ

Un NVT compatible PTZ peut être doté d'un ou de plusieurs nœuds PTZ. Le nœud PTZ peut être doté d'un pilote PTZ mécanique, d'un pilote PTZ chargé sur un codeur vidéo ou d'un pilote PTZ numérique. Le nœud PTZ est l'entité de niveau le plus bas du contrôle PTZ, et il spécifie les fonctionnalités PTZ prises en charge.

Les configurations PTZ sont définies *par profil multimédia* et traitées par l'intermédiaire de ces commandes de configuration:

- obtenir et définir les configurations de panoramique, horizontal/vertical et zoom;
- obtenir les options de configuration de panoramique, horizontal/vertical et zoom.

La présente Norme définit les opérations de contrôle PTZ suivantes:

- opérations de déplacement PTZ absolu, relatif et continu;
- opération d'arrêt;
- obtenir l'information de statut PTZ (position, erreur et statut de déplacement, par exemple);
- obtenir, définir, supprimer et déplacer à la position préétablie;

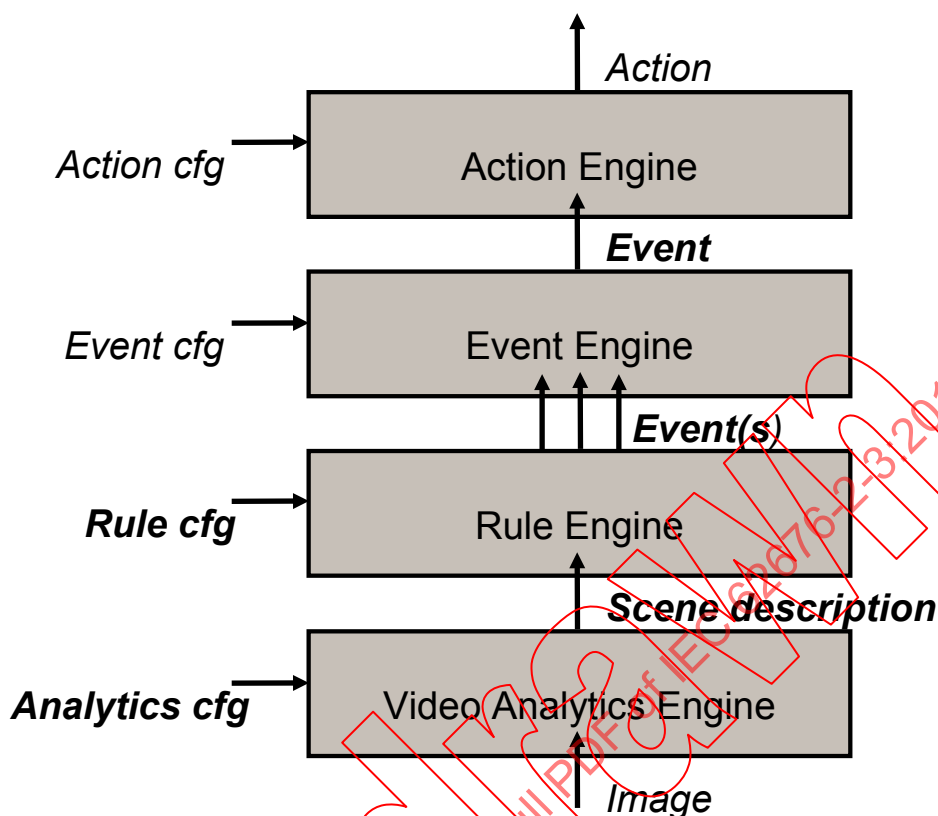
- obtenir, définir et déplacer à la position initiale.

4.12 Analyse vidéo

Les applications d'analyse vidéo sont divisées en parties d'analyse d'image et spécifiques à l'application. L'interface entre ces deux parties produit une abstraction qui décrit la scène sur la base des objets présents. Les applications d'analyse vidéo sont réduites à une comparaison des descriptions et des règles de la scène (comme les lignes virtuelles qu'il est interdit de franchir ou les polygones qui définissent une zone protégée). D'autres règles peuvent représenter un comportement à l'intérieur des objets, comme des objets suivant d'autres objets (pour former une détection par talonnage). Ces règles peuvent également permettre de décrire un mouvement d'objet interdit, qui peut être utilisé pour établir une limite de vitesse.

Ces deux parties séparées, appelées "moteur d'analyse vidéo" et "moteur de règles", forment avec les événements et les actions l'architecture d'analyse vidéo selon la présente norme (voir Figure 6).

L'architecture d'analyse vidéo est composée d'éléments et d'interfaces. Chaque élément donne une fonctionnalité correspondant à une entité unique d'un point de vue sémantique de la solution complète d'analyse vidéo. Les interfaces sont unidirectionnelles et définissent une entité d'information avec un contenu unique. Seules les interfaces sont soumises à cette spécification. Le point central de cette architecture est la capacité à répartir les éléments ou ensembles d'éléments adjacents vers un dispositif du réseau.



IEC 2746/13

Légende

Anglais	Français
Action Engine	Moteur d'actions
Event engine	Moteur d'événements
Rule engine	Moteur de règles
Video Analytics Engine	Moteur d'analyse vidéo
Action cfg	Cfg d'action
Event cfg	Cfg d'événement
Rule cfg	Cfg de règles
Analytics cfg	Cfg d'analyses
Event	Événement
Event(s)	Événement (s)
Scene description	Description de scène

Figure 6 – Architecture d'analyse vidéo

Les interfaces suivantes sont définies dans la présente Norme:

- interface de configuration d'analyses;
- description de scène;
- interface de configuration de règles;
- interface d'événement.

La norme définit un cadre de configuration pour le moteur d'analyse vidéo. Ce cadre permet à un client de demander au dispositif les modules analytiques pris en charge responsables de

leurs configurations. Les configurations de ces modules peuvent être ajoutées, supprimées ou modifiées de manière dynamique par un client, lui permettant d'exécuter plusieurs modules d'analyse vidéo en parallèle, s'ils sont pris en charge par le dispositif.

La sortie du moteur d'analyse vidéo est appelée *Description de scène*. La Description de scène représente l'abstraction de la scène en termes d'objets, statiques ou dynamiques, qui font partie de la scène. La présente norme définit une interface de description de scène XML comprenant les types de données et les mécanismes de transport de données.

Les règles décrivent la manière d'interpréter la description de scène et de réagir à cette information. La présente norme définit la syntaxe de règle et les méthodes normalisées pour communiquer ces règles de l'application au dispositif.

Un événement signale l'état de l'analyse de la description de scène et des règles associées. L'interface d'événement est à la fois l'entrée et la sortie de l'élément de moteur d'événements. L'interface d'événement est traitée par l'intermédiaire de la notification générale et du cadre d'application des sujets (voir 4.10).

Le WSDL du service d'analyse vidéo fait partie du cadre d'application et est fourni dans le fichier WSDL d'analyse.

4.13 Dispositif d'analyse

Il faut utiliser le service de dispositif d'analyse pour les dispositifs d'analyse autonomes qui exécutent des processus d'évaluation sur les flux multimédia ou les flux multimédia améliorés par métadonnées. Les évaluations peuvent concerner plusieurs flux multimédia ou flux multimédia améliorés par métadonnées à la fois.

Le service de dispositif d'analyse reçoit les flux multimédia ou flux multimédia améliorés par métadonnées de la part de dispositifs de génération active ou de stockage. Il peut être doté de fonctionnalités de décodeur si l'analyse est réalisée sur des données non compressées.

Le service de dispositif d'analyse est utilisé par le client pour configurer les propriétés et fonctionnalités d'un dispositif d'analyse autonome. Les fonctionnalités de voie de retour ne sont pas fournies par les dispositifs d'analyse autonomes.

La sortie du service de dispositif d'analyse peut être obtenue à l'aide du service d'événements, la commande `GetStreamUri` étant également prise en charge.

4.14 Affichage

Le service d'affichage offre des fonctions permettant à un client de contrôler et configurer les dispositifs d'affichage. Le service présente des panneaux, chacun d'eux occupant une zone de l'affichage physique. La configuration du panneau mappe les entrées et sorties audio à une sortie vidéo. La configuration fait également référence à un objet de réception qui reçoit les données à afficher. Les fonctions d'extraction et de configuration d'un panneau sont fournies.

Une présentation définit dans quelle mesure ces panneaux sont visibles à l'écran (une seule vue ou quatre vues, par exemple). Le service introduit des commandes d'extraction de la présentation en cours d'un affichage et modifie la présentation.

Il introduit également des commandes pour demander les fonctionnalités de codage et de décodage d'une sortie vidéo ainsi que les options de présentation.

4.15 Récepteur

4.15.1 Généralités

Un récepteur est un objet qui agit comme un point terminal de client RTSP. Les récepteurs sont utilisés par d'autres services qui utilisent des flux multimédia (les services de dispositif d'affichage, d'enregistrement et d'analyse, par exemple). La configuration d'un récepteur détermine le point terminal RTSP vers lequel il convient de se connecter, ainsi que les paramètres de connexion qu'il convient d'utiliser.

Un récepteur peut fonctionner en trois modes distincts:

- Toujours connecté. Le récepteur tente de maintenir une connexion permanente au point terminal configuré.
- Jamais connecté. Le récepteur ne tente pas de se connecter.
- Connexion automatique. Le récepteur se connecte à la demande des utilisateurs des flux multimédia.

Un seul récepteur peut être utilisé par plusieurs utilisateurs. Par exemple, pour enregistrer un flux et l'analyser, un travail d'enregistrement et un moteur d'analyses peuvent être associés au même récepteur. Si le récepteur utilise le mode "Connexion automatique", il se connecte à chaque fois que le travail d'enregistrement ou le moteur d'analyses est actif, et se déconnecte lorsqu'ils sont inactifs.

Les récepteurs peuvent être créés et supprimés manuellement, en appelant les opérations CreateReceiver et DeleteReceiver du service de récepteur ou automatiquement par d'autres services. Par exemple, si un travail d'enregistrement est créé avec l'option "AutoCreateReceiver", il est automatiquement créé et associé à un récepteur. La suppression d'un travail d'enregistrement supprime également le récepteur.

4.15.2 Points de synchronisation

Étant donné que les récepteurs utilisent des adresses RTSP pour spécifier la source du flux, ils n'ont pas nécessairement accès à l'interface des services Web de l'émetteur. Cela signifie qu'ils ne peuvent pas utiliser la commande SetSynchronizationPoint décrite en 11.18.2.

Il convient que les récepteurs utilisent plutôt le message PLI décrit dans la norme [RFC 4585] pour demander un point de synchronisation.

4.16 Stockage

La présente Norme fournit un ensemble d'interfaces permettant de prendre en charge les dispositifs de stockage en réseau interopérables, tels que les enregistreurs vidéo en réseau (NVR), les enregistreurs vidéo numériques (DVR) et les caméras à stockage intégré.

Les fonctions suivantes sont prises en charge:

- contrôle d'enregistrement;
- recherche;
- lecture.

Ces fonctions sont offertes par trois services liés:

Le **service d'enregistrement**, qui permet à un client de gérer les enregistrements et de configurer le transfert de données entre les sources de données et les enregistrements. La gestion des enregistrements comprend la création et la suppression des enregistrements et des pistes.

Le **service de recherche**, qui permet à un client de rechercher des informations relatives à des enregistrements sur le dispositif de stockage (pour concevoir une vue "chronologique", par exemple) et rechercher des données dans un ensemble d'enregistrements. Il s'agit de rechercher des événements inclus dans l'enregistrement de piste de métadonnées.

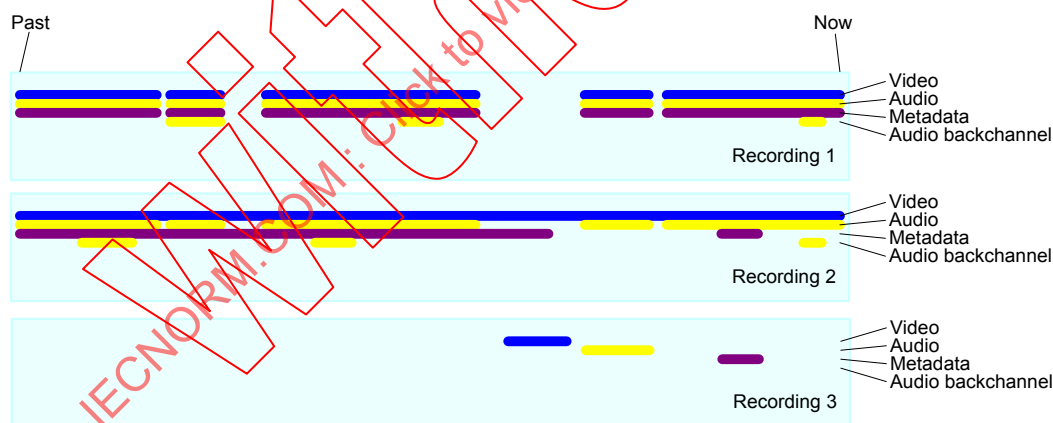
Le **service de lecture** permet à un client de lire les données enregistrées, y compris les données vidéo, les données audio et les métadonnées. Des fonctions permettent de démarrer et d'arrêter la restitution et de modifier la vitesse et le sens du flux restitué. Le service permet également à un client de télécharger des données à partir du dispositif de stockage, de manière à pouvoir offrir une fonction d'export.

4.16.1 Modèle de stockage

Les interfaces de stockage de la présente Norme présentent une vue logique des données sur le dispositif de stockage. Cette vue est totalement indépendante de la manière dont les données peuvent être physiquement stockées sur le disque.

Le concept clé du modèle de stockage est l'*enregistrement*. Le terme *enregistrement* est utilisé dans la présente norme pour indiquer un conteneur d'un ensemble de *pistes* audio, vidéo et de métadonnées associées, provenant en général de la même source de données (une caméra, par exemple). Un *enregistrement* peut comporter un certain nombre de pistes. Une *piste* est perçue comme une chronologie infinie contenant des données à certains moments.

Un enregistrement est au moins capable de contenir trois pistes: une piste audio, une piste vidéo et une piste de métadonnées. Certaines mises en œuvre du service d'enregistrement peuvent prendre en charge plusieurs pistes de chaque type. Par exemple, le même enregistrement peut contenir deux pistes vidéo, l'une contenant un flux basse résolution ou bas débit, et l'autre un flux haute résolution ou haut débit. La Figure 7 donne un exemple de trois enregistrements et des informations de piste correspondantes au cours du temps.



IEC 2747/13

Légende

Anglais	Français
Past	Passé
Now	Présent
Video	Vidéo
Audio	Audio
Metadata	Métadonnées
Audio backchannel	Voie de retour audio
Recording	Enregistrement

Figure 7 – Modèle de stockage avec pistes

Il est important de noter que les interfaces de stockage n'exposent pas les interfaces de stockage internes sur le dispositif. En particulier, un enregistrement n'est pas destiné à représenter un seul fichier du disque même si, dans de nombreuses mises en œuvre de dispositif de stockage, un enregistrement est physiquement stocké dans une série de fichiers. Par exemple, certaines mises en œuvre de caméra enregistrent des alarmes en créant un fichier distinct pour chaque alarme déclenchée. Même si chaque fichier peut être représenté comme étant un *enregistrement* différent, le modèle de la présente Norme a pour objet de regrouper tous ces fichiers dans un seul enregistrement.

Dans un enregistrement, les régions dans lesquelles les données sont réellement enregistrées sont représentées par des paires d'événements, chacune d'elles étant composée d'un événement de début d'enregistrement et d'un événement de fin d'enregistrement. Un client peut concevoir la vue logique des enregistrements à l'aide des méthodes FindRecordings et FindEvents du service de recherche.

Si les métadonnées sont enregistrées, la piste de métadonnées peut contenir tous les événements générés par la source de données (voir l'article relatif au traitement d'événements et l'objet MetadataConfiguration). De plus, d'un point de vue conceptuel, un dispositif enregistre également les événements d'historique définis par l'ONVIF (voir Enregistrement des descriptions d'événement dans le service de recherche), cela incluant des informations telles que le début et la fin d'une plage de données enregistrée. D'un point de vue conceptuel, un dispositif peut également enregistrer les événements d'historique spécifiques au fournisseur. Les événements générés par le dispositif ne sont pas insérés dans les pistes de métadonnées existantes des enregistrements. La méthode FindEvents du service de recherche peut trouver tous les événements enregistrés.

4.16.2 Enregistrement

Le service d'enregistrement permet à un client de gérer les enregistrements et de configurer le transfert de données entre les sources de données et les enregistrements. La gestion des enregistrements comprend la création et la suppression des enregistrements et des pistes.

L'enregistrement des travaux permet de transférer des données d'une source d'enregistrement vers un enregistrement. Une source d'enregistrement peut être un objet de réception créé avec le service de récepteur, ou un profil multimédia qui code les données sur un dispositif local. Le profil multimédia peut être utilisé comme une source sur une caméra à stockage intégré.

Pour sauvegarder des données sur un enregistrement, un client crée en premier lieu un enregistrement et vérifie que l'enregistrement dispose des pistes nécessaires. Ensuite, le client crée un travail d'enregistrement qui extrait les données d'une ou de plusieurs sources et les stocke sur les pistes de l'enregistrement.

Les clients peuvent configurer plusieurs travaux d'enregistrement dans le même enregistrement. Si plusieurs travaux d'enregistrement sont actifs, le dispositif utilise un schéma de priorité pour faire un choix parmi les pistes définies dans les travaux d'enregistrement. Les clients peuvent à tout moment changer le mode des travaux d'enregistrement, en permettant la mise en œuvre de caractéristiques telles que l'enregistrement d'alarme ou l'enregistrement manuel.

Le travail d'enregistrement s'appuie sur le service de récepteur pour recevoir des données provenant d'autres dispositifs grâce aux objets de récepteur identifiés par ReceiverTokens.

4.16.3 Recherche

Le service de recherche permet à un client de rechercher des informations relatives à des enregistrements sur le dispositif de stockage (pour concevoir une vue "chronologique", par exemple) et rechercher des données dans un ensemble d'enregistrements. Il s'agit de

rechercher des événements et d'autres informations inclus dans l'enregistrement de piste de métadonnées.

Le service de recherche offre les fonctionnalités suivantes:

- recherche d'enregistrements et d'informations relatives à chacun d'eux;
- recherche d'événements dans les métadonnées et parmi les événements historiques;
- recherche de positions PTZ dans les métadonnées;
- recherche d'autres informations dans les métadonnées (texte issu de systèmes de point de vente électronique, par exemple).

La recherche réelle, qui est asynchrone, a lieu sous la forme d'opérations couplées de recherche et de résultats. Chaque opération de recherche lance une session de recherche. Le client peut alors obtenir les résultats issus de la session de recherche par incrément ou en totalité, selon la mise en œuvre et l'étendue de la recherche. Il existe quatre paires d'opérations de recherche pour les enregistrements, les événements d'enregistrement, les positions PTZ et les métadonnées.

FindRecordings et GetRecordingSearchResults

FindEvents et GetEventSearchResults

FindPTZPosition et GetPTZPositionSearchResults

FindMetadata et GetMetadataSearchResults

4.16.4 Lecture

Le service de lecture offre un mécanisme de lecture des données vidéo, des données audio et des métadonnées. Le service peut également être utilisé pour télécharger des données à partir du dispositif de stockage, de manière à pouvoir offrir une fonction d'export.

Le protocole de lecture repose sur RTSP [RFC 2326]. Toutefois, RTSP ne prenant pas directement en charge toutes les exigences en matière de lecture, plusieurs extensions ont été ajoutées au protocole. En particulier, une extension d'en-tête RTP est définie pour pouvoir associer un horodatage absolu à chaque unité d'accès (une trame vidéo, par exemple) et acheminer des informations relatives à la continuité du flux.

La commande GetReplayUri du service de lecture renvoie l'URL RTSP d'un enregistrement afin de pouvoir assurer la lecture à l'aide de RTSP.

4.17 Sécurité

Le présent paragraphe décrit les exigences de sécurité vidéo en réseau. La présente norme définit le mécanisme de sécurité à deux niveaux de communication différents:

- sécurité au niveau transport;
- sécurité au niveau du message.

La présente norme définit également la sécurité réseau basée sur le port de la manière suivante.

- IEEE 802.1X

Les exigences générales de sécurité, les définitions et les exigences de sécurité du transport sont spécifiées à l'Article 0. Les exigences de sécurité au niveau du message sont spécifiées en 5.12. Les exigences IEEE 802.1X sont spécifiées en 0. La gestion de la sécurité est traitée grâce au service de gestion du dispositif (voir 4.5.7 ci-dessous).

5 Interopérabilité des services Web

Toutes les commandes de gestion et de configuration reposent sur les services Web.

Pour les besoins de la présente Norme:

- le dispositif (NVT, NVD, NVS, NVA) est un fournisseur de services;
- le client est un demandeur de services.

Un système vidéo en réseau classique ne dispose pas d'un seul client qui traite toutes les opérations de configuration et de gestion du dispositif pour un seul dispositif. Il peut en effet exister une distinction entre les fonctionnalités de contrôle et de *récepteur* vidéo en réseau. Un dispositif fournissant des services peut également agir comme un client. Les éditions ultérieures de la présente norme peuvent introduire des entités et interfaces supplémentaires dans le système.

Les services Web nécessitent également une manière commune de découvrir les fournisseurs de services. Cette découverte est obtenue au moyen des spécifications de Description, découverte et intégration universelles (UDDI) [UDDI API ver2], [UDDI Data Structure ver2]. Les spécifications UDDI utilisent des courtiers de service pour la découverte de service. Cette spécification cible les dispositifs, alors que le modèle UDDI n'est pas orienté dispositif. Par conséquent, l'UDDI et les courtiers de service *n'entrent pas dans le domaine d'application* de la présente norme.

Selon la présente norme, les dispositifs (fournisseurs de services) sont découverts à l'aide de techniques basées sur WS-Discovery [WS-Discovery]. Les principes de découverte de service sont décrits à l'Article 7.

Les services Web donnent aux développeurs la liberté de définir des échanges de services et de messages, ce qui peut poser des problèmes d'interopérabilité. L'organisme d'interopérabilité des services Web (WS-I) développe des profils normalisés et des directives permettant de créer des services Web interopérables. Les dispositifs et les clients doivent suivre les directives du profil de base WS-I 2.0 [WS-I BP 2.0]. Les descriptions de services de la présente norme suivent les recommandations du profil de base WS-I 2.0.

5.1 Présentation des services

5.1.1 Généralités

Un dispositif satisfaisant à l'ONVIF doit prendre en charge un certain nombre de services Web définis dans la présente norme. Des exemples de services Web définis par la présente norme sont indiqués ci-dessous:

- service de dispositif;
- service multimédia;
- service d'événement.

Le service de dispositif est le *service cible* d'un dispositif satisfaisant à l'ONVIF et le *point d'entrée* de tous les autres services du dispositif.

Le point d'entrée pour la gestion du dispositif est fixé à:

`http://onvif_host/onvif/device_service`

5.1.2 Exigences des services

Un dispositif doit offrir des services de gestion de dispositif et des services d'événement. Un dispositif PEUT prendre en charge l'un des autres services en fonction des fonctionnalités du dispositif. Selon le type de dispositif (NVT, NVD, NVS, NVA), des services supplémentaires

sont nécessaires. Les exigences exactes de conformité sont définies dans les différentes définitions de service de la présente norme.

Si un dispositif prend en charge un certain service, il doit répondre à toutes les commandes définies dans le WSDL du service correspondant. Si la commande spécifique n'est pas requise pour ce service et que le dispositif ne prend pas en charge la commande, le dispositif doit répondre à une requête par les codes d'erreur:

env:Receiver,

ter:ActionNotSupported,

voir 5.11.2 pour la définition des codes d'erreur.

Le Tableau 1 présente les services requis en fonction des différents types de dispositif. Les services obligatoires sont marqués d'un "M", les services obligatoires si une caractéristique connexe est prise en charge par le dispositif étant marqués d'un "C".

Tableau 1 – Exigences des services en fonction des types de dispositif

	NVT	NVS	NVD	NVA
Dispositif	M	M	M	M
Événement	M	M	M	M
Multimédia	M			
PTZ	C			
Imagerie				
Analyses				M
Contrôle d'enregistrement		C		
Recherche d'enregistrement		M		
Contrôle de lecture		M		
Dispositif ES	M		M	
Récepteur		C	M	M
Affichage			M	
Dispositif d'analyse				M

5.2 Présentation de WSDL

"WSDL est un format XML de description des services réseau, se présentant sous la forme d'un ensemble de points terminaux opérant sur des messages contenant des informations orientées document ou procédure. Les opérations et les messages sont décrits de manière abstraite, puis associés à un protocole de réseau et un format de message concrets afin de définir un point terminal. Les points terminaux concrets associés sont combinés en points terminaux abstraits (services). WSDL est extensible, afin de permettre la description de points terminaux et de leurs messages indépendamment des formats de message ou des protocoles de réseau utilisés pour communiquer" [WSDL1.1].

La présente norme suit la spécification WSDL 1.1 et utilise le schéma littéral/enveloppant.

Un document WSDL est composé des sections suivantes:

- types – définition des types de données à l'aide de définitions de schéma XML;
- message – définition du contenu des messages d'entrée et de sortie;

- opération – définition de la manière dont les messages d'entrée et de sortie sont associés à une opération logique;
- type de port – regroupe un ensemble d'opérations;
- liaison – spécification des protocoles utilisés pour l'échange de message pour un type de port particulier;
- port – spécifie une adresse pour une liaison;
- service – utilisé pour regrouper un ensemble de ports associés.

5.3 Espaces de nom

Les préfixes et les espaces de nom utilisés dans la présente Norme figurent dans le Tableau 2. Ces préfixes ne font pas partie de la norme, une mise en œuvre pouvant utiliser le préfixe de son choix.

Tableau 2 – Espaces de nom définis dans la présente norme

Préfixe	URI d'espace de nom	Description
tt	http://www.onvif.org/ver10/schema	Descriptions de schéma XML de la présente norme.
tds	http://www.onvif.org/ver10/device/wsdl	Espace de nom du service de dispositif WSDL.
trt	http://www.onvif.org/ver10/media/wsdl	Espace de nom du service multimédia WSDL.
timg	http://www.onvif.org/ver20/imaging/wsdl	Espace de nom du service d'image WSDL.
tev	http://www.onvif.org/ver10/events/wsdl	Espace de nom du service d'événement WSDL.
tptz	http://www.onvif.org/ver20/ptz/wsdl	Espace de nom du service de contrôle PTZ WSDL.
tan	http://www.onvif.org/ver20/analytics/wsdl	Espace de nom du service d'analyse WSDL.
ter	http://www.onvif.org/ver10/error	Espace de nom des pannes définies par l'ONVIF.
dn	http://www.onvif.org/ver10/network/wsdl	Espace de nom utilisé pour le service de découverte de dispositif à distance de la présente norme.
tns1	http://www.onvif.org/ver10/topics	Espace de nom de l'espace de nom de rubrique de l'ONVIF.
tad	http://www.onvif.org/ver10/analyticsdevice/wsdl	Espace de nom du service de dispositif d'analyse WSDL.
tmd	http://www.onvif.org/ver10/deviceIO/wsdl	Espace de nom du service de dispositif ES WSDL.
tls	http://www.onvif.org/ver10/display/wsdl	Espace de nom du service d'affichage WSDL.
trv	http://www.onvif.org/ver10/receiver/wsdl	Espace de nom du service de récepteur WSDL.
trc	http://www.onvif.org/ver10/recording/wsdl	Espace de nom du service d'enregistrement WSDL.
trp	http://www.onvif.org/ver10/replay/wsdl	Espace de nom du service de lecture WSDL.
tse	http://www.onvif.org/ver10/search/wsdl	Espace de nom du service de recherche WSDL.

Les espaces de nom figurant dans le Tableau 3 sont référencés par la présente Norme.

Tableau 3 – Espaces de nom référencés (avec préfixe)

Préfixe	URI d'espace de nom	Description
wSDL	http://schemas.xmlsoap.org/wSDL/	Espace de nom WSDL pour le cadre d'application WSDL.
soap12	http://schemas.xmlsoap.org/wSDL/soap12/	Espace de nom WSDL pour la liaison WSDL SOAP 1.2.
http	http://schemas.xmlsoap.org/wSDL/http/	Espace de nom WSDL pour la liaison WSDL GET & POST HTTP.
soapenc	http://www.w3.org/2003/05/soap-encoding	Espace de nom de codage défini par SOAP 1.2 [SOAP 1.2, Part 2]
soapenv	http://www.w3.org/2003/05/soap-envelope	Espace de nom d'enveloppe défini par SOAP 1.2 [SOAP 1.2, Part 1]
xs	http://www.w3.org/2001/XMLSchema	Espace de nom d'instance défini par XS [Schéma XML, Partie 1] et [Schéma XML, Partie 2]
xsi	http://www.w3.org/2001/XMLSchema-instance	Espace de nom d'instance de schéma XML
d	http://schemas.xmlsoap.org/ws/2005/04/discovery	Espace de nom de découverte de dispositif défini par [WS-Discovery]
wsadis	http://schemas.xmlsoap.org/ws/2004/08/addressing	Espace de nom d'adressage de dispositif référencé dans [WS-Discovery]
wsa	http://www.w3.org/2005/08/addressing	Espace de nom d'adressage de dispositif défini par [WS-Addressing]
wstop	http://docs.oasis-open.org/wsn/t-1	Espace de nom de schéma de la spécification [WS-Topics]
wsnt	http://docs.oasis-open.org/wsn/b-2	Espace de nom de schéma de la spécification [WS-BaseNotification]
xop	http://www.w3.org/2004/08/xop/include	Espace de nom d'encapsulation optimisée binaire XML défini par [XOP]

De plus, la présente Norme se réfère aux espaces de nom sans préfixe figurant dans le Tableau 4.

Tableau 4 – Espaces de nom référencés (sans préfixe)

URI d'espace de nom	Description
http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete	Dialecte d'expression de rubrique défini pour les expressions de rubrique.
http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet	Dialecte ONVIF pour les expressions de rubrique.
http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter	Dialecte de filtrage ONVIF utilisé pour le filtrage de contenu de message.
http://www.onvif.org/ver10/tpz/ZoomSpaces/PositionGenericSpace	Espace de position de zoom normalisé ONVIF pour le contrôle PTZ.
http://www.onvif.org/ver10/tpz/PanTiltSpaces/PositionGenericSpace	Espace de position de basculement horizontal/vertical normalisé ONVIF pour le contrôle PTZ.
http://www.onvif.org/ver10/tpz/ZoomSpaces/TranslationGenericSpace	Espace de translation de zoom normalisé ONVIF pour le contrôle PTZ.
http://www.onvif.org/ver10/tpz/PanTiltSpaces/TranslationGenericSpace	Espace de translation de basculement horizontal/vertical normalisé ONVIF pour le contrôle PTZ.
http://www.onvif.org/ver10/tpz/ZoomSpaces/VelocityGenericSpace	Espace de vitesse de zoom normalisé ONVIF pour le contrôle PTZ.
http://www.onvif.org/ver10/tpz/PanTiltSpaces/VelocityGenericSpace	Espace de vitesse de basculement horizontal/vertical normalisé ONVIF pour le contrôle PTZ.
http://www.onvif.org/ver10/tpz/ZoomSpaces/SpeedGenericSpace	Espace de vitesse de zoom normalisé ONVIF pour le contrôle PTZ.
http://www.onvif.org/ver10/tpz/PanTiltSpaces/SpeedGenericSpace	Espace de vitesse de basculement horizontal/vertical normalisé ONVIF pour le contrôle PTZ.

5.4 Types

Les types de données sont définis à l'aide de descriptions de schéma XML Partie 1 et Partie 2. Tous les types de données définis dans la présente norme sont inclus dans [ONVIF Schema] et peuvent être téléchargés depuis l'adresse suivante:

- <http://www.onvif.org/onvif/ver10/schema/onvif.xsd>

5.5 Messages

Selon WSDL 1.1, les opérations sont décrites au moyen de messages d'entrée et de sortie au format XML. La section de message contient le contenu du message.

Dans la présente spécification, un message contient deux éléments principaux:

- nom de message;
- parties du message.

Le nom de message spécifie le nom de l'élément. Il est utilisé dans la définition d'opération du document WSDL. Le nom de message définit le nom du message.

L'élément parties du message WSDL permet de définir le format réel du message. Bien qu'un message WSDL puisse être composé de plusieurs parties, la spécification suit le profil de base WS-I [WS-I BP 2.0] et n'autorise pas plus d'un élément de partie dans un message. Le même nom est donc toujours utilisé ("paramètres") pour le nom de partie du message.

La notation WSDL qui suit est utilisée pour les messages de la présente Norme:

```
<message name="" Operation_Name' Request">  
  
    <part name="parameters" element="" prefix':' Operation_Name'"/>  
  
</message>
```

respectivement,

```
<message name="" Operation_Name' Response">  
  
    <part name="parameters" element="" prefix':' Operation_Name' Response"/>  
  
</message>
```

où "prefix" est le préfixe de l'espace de nom dans lequel est défini le message.

Cette spécification utilise des types spécifiques de message qui encapsulent de multiples parties pour autoriser de multiples arguments (ou données) dans les messages.

5.6 Opérations

Les opérations sont définies dans la déclaration WSDL portType (type de port). Une opération peut être de l'un de ces deux types suivants:

- One-way – unidirectionnel, le fournisseur de services reçoit un message.
- Request-response – demande-réponse, le fournisseur de services reçoit un message et envoie un message correspondant.

Selon l'opération, différents types de port peuvent être utilisés.

Le nom d'opération définit le nom de l'opération.

Les opérations de la spécification sont définies à l'aide du format de table suivant illustré dans le Tableau 5.

Tableau 5 – Description des opérations utilisée dans la présente norme

Operation_Name		Type d'opération
Nom du message	Description	
'Operation_Name'Request	<i>Description du message de requête.</i> $Type_{r1} \text{ Nom}_{r1} [a_{r1}][b_{r1}]$ $Type_{r2} \text{ Nom}_{r2} [a_{r2}][b_{r2}]$: $Type_{rn} \text{ Nom}_{rn} [a_{rn}][b_{rn}]$	
'Operation_Name'Response	<i>Description du message de réponse.</i> $Type_{s1} \text{ Nom}_{s1} [a_{s1}][b_{s2}]$ $Type_{s2} \text{ Nom}_{s2} [a_{s2}][b_{s2}]$: $Type_{sn} \text{ Nom}_{sn} [a_{sn}][b_{sn}]$	
'FaultMessage_Name'	<i>Si des défauts spécifiques à cette opération sont définis, ce champ décrit la structure du message de défaut défini.</i>	
Codes de défaut	Description	
Code	Description du défaut spécifique à l'opération.	
Sous-code		
Sous-code		

La colonne de description comprend une liste des éléments (le cas échéant) inclus respectivement dans les messages de requête et de réponse. La valeur entre crochets définit les limites supérieure et inférieure du nombre d'occurrences qui peuvent être prévues pour l'élément du type spécifié. Par exemple, Nom_{s2} du tableau ci-dessus se produit a_{s2} fois au minimum et b_{s2} fois au maximum.

La plupart des commandes ne définissent *aucun* message de défaut spécifique. Si un message est défini, il figure dans le tableau immédiatement après le message de réponse.

Les codes de défaut figurant dans les tableaux sont les codes de *défaut spécifiques* que l'on peut attendre de la commande (voir 5.11.2.2). *Chaque commande peut générer un défaut générique* (voir 5.11.2.2).

5.6.1 Type d'opération unidirectionnelle

Un type d'opération unidirectionnelle est utilisé lorsque le fournisseur de services reçoit un message de contrôle et *n'envoie aucun* message explicite d'accusé de réception ni de confirmation. La présente norme utilise les opérations unidirectionnelles pour la découverte et les événements uniquement.

Ce type d'opération est défini par un seul message d'entrée.

5.6.2 Type d'opération demande-réponse (request-response)

Une opération demande-réponse est utilisée lorsque le fournisseur de services reçoit un message et répond par un message correspondant.

Ce type d'opération est défini par un message d'entrée, un message de sortie et plusieurs messages de défaut.

5.7 Types de port

Un type de port est un ensemble nommé d'opérations abstraites et de messages abstraits impliqués. Un seul type de port est un ensemble de plusieurs opérations différentes.

Tous les noms d'opérations de la présente norme sont classés en catégories. Chaque catégorie d'opérations contient une ou plusieurs opérations. Chaque catégorie contient *un seul type* d'opération et est regroupée en un seul *type de port*. Une opération unidirectionnelle et une opération demande-réponse ne peuvent jamais exister pour le même type de port.

5.8 Liaison

Une liaison définit une spécification concrète de format de données de transport et de protocole pour un type de port particulier. Il peut exister n'importe quel nombre de liaisons pour un type de port donné.

"Port_type" est un type défini au préalable et "Liaison" est une chaîne de caractères commençant par une lettre majuscule qui définit le nom de la liaison.

Les définitions de liaison pour un dispositif conforme à la présente norme doivent suivre les exigences contenues dans [WS-I BP 2.0]. Cela implique que les liaisons WSDL SOAP 1.2 doivent être utilisées.

La liaison SOAP peut avoir différents styles. Un dispositif doit utiliser le style "document" spécifié au niveau opérationnel.

Les liaisons sont définies dans les spécifications WSDL pour les services respectifs.

5.9 Ports

Le point terminal individuel est spécifié par une adresse unique correspondant à une liaison. Un nom unique doit être attribué à chaque port. Une définition de port contient un nom et un attribut de liaison.

La présente norme n'impose aucun principe de nomination des ports.

5.10 Services

Un service est un ensemble de ports associés. La présente norme n'impose aucun principe de nomination des services.

5.11 Traitement des erreurs

Comme pour tout autre protocole, des erreurs peuvent se produire lors des communications ou du traitement des messages ou du protocole.

La spécification classe le traitement des erreurs selon les catégories suivantes:

- erreurs de protocole;
- erreurs SOAP;

- erreurs d'application.

5.11.1 Erreurs de protocole

Les *erreurs de protocole* sont le résultat d'un message de protocole formé de manière incorrecte, qui peut contenir des valeurs d'en-tête illégales, être reçu de manière intempestive ou faire l'objet d'un retard du connecteur. Pour indiquer et interpréter les erreurs de protocole, les protocoles HTTP et RTSP ont défini un ensemble de codes de statut normalisés [ex. 1xx, 2xx, 3xx, 4xx, 5xx]. Conformément à la présente Norme, le dispositif et le client doivent utiliser les codes de statut appropriés définis par les protocoles RTSP et HTTP pour les rapports d'erreur, et les gérer correctement lorsqu'ils en reçoivent.

5.11.2 Erreurs SOAP

Les *erreurs SOAP* sont générées suite à des erreurs de fonctionnement des services Web ou lors du traitement d'un message SOAP. Toutes ces erreurs SOAP doivent être signalées et traitées par l'intermédiaire de messages de défaut SOAP. La spécification SOAP donne un cadre d'application commun bien défini pour traiter les erreurs par l'intermédiaire de SOAP.

Un message de défaut SOAP est un message SOAP normal dont le corps contient un seul élément bien connu (soapenv:Fault). Pour mieux comprendre l'erreur, SOAP a défini une structure de message de défaut SOAP contenant divers composants.

- code de défaut;
- sous-code;
- raison;
- nœud et rôle;
- détails de défaut.

Les éléments d'information **Sous-code et Détail de défaut** sont destinés à transporter les informations d'erreur spécifiques à l'information.

La présente Norme utilise un espace de nom séparé pour les défauts spécifiques (voir 5.11.2.2):

ter = "http://www.onvif.org/ver10/error".

Les messages de défaut SOAP des différents services Web sont définis dans les définitions des différents services Web. Le serveur et le client doivent utiliser le traitement de message de défaut SOAP 1.2, comme spécifié dans la présente norme, et doivent suivre les recommandations de traitement des défauts du profil de base WS-I 2.0.

L'exemple qui suit est un message d'erreur (message de défaut SOAP 1.2 via HTTP). Les valeurs en italique sont des valeurs fictives remplaçant les valeurs réelles.

```
HTTP/1.1 500 Internal Server Error
CONTENT-LENGTH: bytes in body
CONTENT-TYPE: application/soap+xml; charset="utf-8"
DATE: when response was generated
<?xml version="1.0" ?>
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-
envelope"
    xmlns:ter="http://www.onvif.org/ver10/error"
    xmlns:xs="http://www.w3.org/2000/10/XMLSchema">
<soapenv:Body>
<soapenv:Fault>
<soapenv:Code>
<soapenv:Value>fault code</soapenv:Value>
<soapenv:Subcode>
```

```

    <soapenv:Value>ter:fault subcode</soapenv:Value>
    <soapenv:Subcode>
      <soapenv:Value>ter:fault subcode</soapenv:Value>
    </soapenv:Subcode>
  </soapenv:Subcode>
</soapenv:Code>
<soapenv:Reason>
  <soapenv:Text xml:lang="en">fault reason</soapenv:Text>
</soapenv:Reason>
<soapenv:Node>http://www.w3.org/2003/05/soap-
envelope/node/ultimateReceiver</soapenv:Node>
<soapenv:Role>http://www.w3.org/2003/05/soap-
envelope/role/ultimateReceiver</soapenv:Role>
<soapenv:Detail>
  <soapenv:Text>fault detail</soapenv:Text>
</soapenv:Detail>
</soapenv:Fault>
</soapenv:Body>
</soapenv:Envelope>

```

Le Tableau 6 résume les codes d'erreur SOAP généraux (les codes d'erreur sont définis dans SOAP version 1.2 Part 1: Messaging Framework). Le serveur et le client PEUVENT définir des sous-codes de défaut supplémentaires pour les applications.

La distinction est faite entre les défauts généraux et les défauts spécifiques. Une commande peut générer un défaut général. Les défauts spécifiques sont associés à une commande ou un jeu de commandes spécifique. Les défauts spécifiques qui s'appliquent à une commande particulière sont définis dans le tableau de définition des commandes.

Dans les tableaux de défauts de la présente norme, le Fault Code, le Subcode et le Fault Reason sont définis comme des valeurs normatives. La colonne de description est ajoutée à titre d'information.

5.11.2.1 Défauts généraux

Le Tableau 6 répertorie les codes de défaut général et, le cas échéant, les sous-codes. Toutes les mises en œuvre de serveur et de client doivent traiter toutes les défauts énumérés ci-dessous. Une commande de service Web peut retourner un ou plusieurs des défauts généraux.

Les défauts indiqués sans *sous-code* n'ont aucune valeur *subcode*.

Tableau 6 – Défauts généraux

Code de défaut	Sous-code	Raison de défaut	Description
env:VersionMismatch		Défaut de correspondance de version SOAP	Le dispositif a trouvé un élément d'information non valide à la place de l'élément d'information attendu <i>Envelope</i> .
env:MustUnderstand		Blocs d'en-tête SOAP incompris	Un ou plusieurs blocs d'en-tête SOAP n'ont pas été compris.
env:DataEncodingUnknown		Codage de données SOAP non pris en charge	Un bloc d'en-tête SOAP ou un élément d'information enfant dans le corps SOAP a un codage de données non pris en charge par le dispositif.
env:Sender	ter:WellFormed	Erreur bien formée	Une violation d'erreur bien formée XML s'est produite.
env:Sender	ter:TagMismatch	Défaut de correspondance de balise	Il y a un défaut de correspondance de nom de balise ou d'espace de nom.
env:Sender	ter:Tag	Aucune balise	Il manque une balise XML.
env:Sender	ter:Namespace	Erreur d'espace de nom	Une erreur d'espace de nom SOAP s'est produite.
env:Sender	ter:MissingAttr	Attribut requis absent	Il manque un attribut obligatoire.
env:Sender	ter:ProhibAttr	Attribut interdit	Un attribut interdit est présent.
env:Sender	ter:InvalidArgs	Arguments non valides	Une erreur due à l'un des cas suivants: argument manquant arguments trop nombreux arguments de type de données incorrect.
env:Sender	ter:InvalidArgVal	Valeur argument invalide	La valeur de l'argument est invalide.
env:Sender	ter:UnknownAction	Action inconnue	Une action inconnue est spécifiée.
env:Sender	ter:OperationProhibited	Opération non autorisée	L'opération demandée n'est pas autorisée par le dispositif.
env:Sender	ter:NotAuthorized	Expéditeur non autorisé	L'action demandée requiert une autorisation et l'expéditeur n'est pas autorisé.
env:Receiver	ter:ActionNotSupported	Action optionnelle non mise en œuvre	L'action demandée est optionnelle et n'est pas mise en œuvre par le dispositif.
env:Receiver	ter:Action	Échec action	L'action SOAP demandée a échoué.
env:Receiver	ter:OutOfMemory	Hors mémoire	Le dispositif n'a pas de mémoire suffisante pour terminer l'action.
env:Receiver	ter:CriticalError	Erreur critique	Le dispositif a rencontré une condition d'erreur qu'il ne peut pas régler lui-même et il a besoin d'une réinitialisation ou d'une réalimentation.

5.11.2.2 Défauts spécifiques

Les défauts spécifiques s'appliquent uniquement à une commande ou un ensemble de commandes spécifique. Les défauts spécifiques sont déclarés comme faisant partie intégrante des définitions de service dans la présente Norme.

5.11.2.3 Erreurs HTTP

Si le serveur attend le début du message entrant et qu'aucun message SOAP n'est reçu, il ne doit PAS générer un défaut SOAP et plutôt envoyer une réponse d'erreur HTTP selon le Tableau 7.

Tableau 7 – Erreurs HTTP

Erreur HTTP	Code d'erreur HTTP	Raison HTTP
Demande mal formulée	400	Demande incorrecte
Requiert une autorisation	401	Non autorisé
La méthode HTTP n'est pas POST ou GET	405	Méthode non autorisée
Méthode d'encapsulation de message non prise en charge	415	Support non pris en charge

Il convient qu'un serveur évite de rapporter les erreurs internes, étant donné que cela peut exposer les failles de sécurité qui peuvent faire l'objet d'une utilisation frauduleuse.

5.12 Sécurité

Les services définis dans la présente Norme doivent être protégés à l'aide du cadre WS-Security. La spécification WS-Security définit un ensemble normalisé d'extensions SOAP qui peut être utilisé pour assurer l'intégrité et la confidentialité des messages des services Web. Ce cadre permet de mettre en œuvre différents modèles de sécurité utilisant des jetons. Les jetons suivants sont actuellement définis.

- Profil de jeton de nom d'utilisateur [WS-UsernameToken];
- Profil de jeton de sécurité X.509 [WS-X.509Token];
- Profil de jeton SAML [WS-SAMLToken];
- Profil de jeton Kerberos [WS-KerberosToken];
- Profil de jeton REL (Rights Expression Language: Langue d'expression des droits) [WS-RELTOKEN].

Un serveur et un client doivent prendre en charge le profil de jeton de nom d'utilisateur comme indiqué dans WS-Security et en 5.12.2, et PEUVENT prendre en charge l'un des autres profils WS-Security définis.

Le profil de jeton de nom d'utilisateur *ne confère qu'un* niveau de sécurité *rudimentaire*. Dans un système dans lequel la sécurité est importante, il est recommandé de toujours configurer le dispositif pour un accès TLS (voir 22.1). La sécurité de niveau message de jeton de nom d'utilisateur combinée avec la sécurité de niveau transport protégé par TLS, avec authentification de client et de serveur, produit un niveau de sécurité acceptable dans un grand nombre de systèmes.

Un dispositif satisfaisant à l'ONVIF doit, lors de l'authentification dans RTSP et HTTP, utiliser des justificatifs provenant du même ensemble de justificatifs définis pour la partie de service Web. Pour les utilisateurs définis avec le profil de jeton de nom d'utilisateur, l'authentification Digest [RFC 2617] doit être utilisée pour RTSP et HTTP.

Il convient qu'un dispositif satisfaisant à l'ONVIF authentifie une requête WS au niveau WS, il convient que HTTP fasse uniquement office de protocole de transport, et le dispositif ne doit pas authentifier une requête WS à ce niveau.

Il convient qu'un dispositif satisfaisant à l'ONVIF authentifie une requête RTSP au niveau RTSP, si HTTP est utilisé pour formuler une requête RTSP, le dispositif ne doit pas authentifier à ce niveau.

Un dispositif satisfaisant à l'ONVIF doit, avec les méthodes d'authentification RTSP et HTTP, utiliser un utilisateur/des justificatifs issus du même ensemble d'utilisateurs/de justificatifs qui sont utilisés pour la partie WS. Pour les utilisateurs définis avec le profil de jeton de nom d'utilisateur, l'authentification Digest [RFC 2617] doit être utilisée pour RTSP et HTTP.

5.12.1 Contrôle d'accès à base d'utilisateurs

Le cadre WS-Security permet la protection et l'authentification au niveau des messages SOAP. Ces mécanismes d'authentification sont utilisés pour construire une règle de sécurité d'accès pour un service ONVIF. La présente spécification permet de configurer une règle de sécurité basée sur quatre niveaux d'utilisateur différents:

- 1) administrateur;
- 2) opérateur;
- 3) utilisateur multimédia;
- 4) anonyme.

Une règle d'accès détaillée pour différentes classes d'utilisateur peut être définie en utilisant ces catégories. Les utilisateurs non authentifiés sont placés dans la catégorie anonyme, et un dispositif ne doit pas ajouter des utilisateurs à la catégorie de niveau d'utilisateur anonyme.

La *politique de sécurité d'accès exacte* doit pouvoir être définie par l'utilisateur du dispositif ou par un administrateur système. Le format exact du fichier de configuration de règle est *hors du domaine d'application* de la présente norme.

Les commandes pour obtenir et définir une politique de sécurité d'accès dans un format arbitraire sont définies en 8.4.

5.12.2 Profil de jeton d'utilisateur

5.12.2.1 Généralités

Le seul profil de jeton WS-Security obligatoire est le profil de jeton d'utilisateur [WS-UsernameToken].

Un client doit utiliser à la fois les pointeurs et les horodatages comme définis dans [WS-UsernameToken]. Le serveur doit rejeter un jeton de nom d'utilisateur n'utilisant pas à la fois un pointeur et des horodatages de *création*.

La présente Norme définit un ensemble de commandes pour gérer les justificatifs de profil de jeton de nom d'utilisateur (voir 0). Ces commandes permettent d'associer des utilisateurs aux différents niveaux d'utilisateur définis en 5.12.1.

5.12.2.2 Déduction de mot de passe

L'utilisation des mêmes justificatifs sur plusieurs dispositifs présente un certain risque en matière de sécurité. Demander à un utilisateur de fournir un justificatif unique pour chaque dispositif n'est pas faisable. Il convient plutôt qu'un client mette en œuvre l'algorithme de déduction de mot de passe ci-dessous.

Soit UA, un utilisateur aléatoire. Soit P-UA, la valeur du mot de passe utilisé par l'utilisateur UA pour accéder aux dispositifs du système. Soit NEP, la valeur de référence du point de service du dispositif final d'un dispositif particulier du système. Enfin, soit PE-UA, l'équivalent du mot de passe utilisé par le client pour accéder à un dispositif particulier du système. Il convient que le client calcule PE-UA comme suit:

$$PE\_UA = \text{base64}(\text{HMAC\_SHA-1}(UA + P\_UA, NEP + \text{"mot de passe ONVIF"})),$$

où "+" est la concaténation et "mot de passe ONVIF" une chaîne ASCII. Il convient de l'inclure sous la forme exacte dans laquelle il est présenté, sans octet de longueur ni caractère nul de fin, c'est-à-dire la valeur hexadécimale suivante: 4F 4E 56 49 46 20 70 61 73 73 77 6F 72 64.

HMAC_SHA-1 est l'algorithme spécifié dans la norme [RFC 2104] utilisant SHA-1 [FIPS 180-2] comme algorithme sous-jacent. La valeur clé à utiliser pour la fonction HMAC est le mot de passe utilisateur, P-UA, directement mappé à son équivalent binaire. De même, il convient de mapper la valeur PE-UA à son équivalent ASCII avant de la transmettre au dispositif.

base64 est décrit dans la norme [RFC 3548], à noter que le résultat de l'opération base64 est le mot de passe réel équivalent et qui doit être utilisé en l'état.

Exemple

Soit l'identifiant et le mot de passe suivants utilisés par le client (ASCII): "user" et "VRxuNzpqR", c'est-à-dire

UA = 75 73 65 72

P-UA = 56 52 78 75 4E 7A 70 71 72 58

Ensuite, supposons que le dispositif dispose de la valeur de référence de point terminal de service de dispositif suivante:

Urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6.

Ensuite, le mot de passe équivalent à utiliser est calculé comme suit:

PE-UA = base64(HMAC_SHA-1(P-UA, NEP+"mot de passe ONVIF")) =
 base64(HMAC_SHA-1(75736572565278754E7A70717258,
 F81D4FAE7DEC11D0A76500A0C91E6BF6+4F4E5649462070617373776F7264)) =
 base64(16 E5 C5 A9 4D DE 8A 97 6D D7 2F 55 78 5F C2 D0 6B DA 53 4A)=
 FuXFqU3eipdt1y9VeF/C0GvaU0o=

L'équivalence de mot de passe obtenue "FuXFqU3eipdt1y9VeF/C0GvaU0o=" est le mot de passe qui doit être utilisé par un client pour configurer les justificatifs client sur le dispositif particulier, puis pour accéder au dispositif.

6 Configuration IP

Le dispositif et le client communiquent sur un réseau IP ouvert ou fermé. La présente Norme n'impose aucune restriction ou exigence générale sur le type de réseau. Cependant, des liaisons de communication doivent pouvoir être établies entre les entités conformément au cadre architectural spécifié à l'Article 4. La configuration IP du dispositif comprend des paramètres tels que des adresses IP et une passerelle par défaut.

Le dispositif doit avoir au moins une interface réseau qui lui donne une connectivité de réseau IP. De manière similaire, le dispositif doit avoir au moins une interface réseau qui lui assure une connectivité IP et permet la communication de données entre le dispositif et le client.

Le dispositif et le client doivent prendre en charge la communication réseau basée sur IPv4. Il convient que le dispositif et le client prennent en charge la communication réseau basée sur IPv6.

Une configuration IP statique doit pouvoir être effectuée sur le dispositif en utilisant une interface de configuration réseau ou locale.

Il convient que le dispositif prenne en charge la configuration IP dynamique d'adresses locales-liaison conformément à la norme [RFC 3927]. Un dispositif qui prend en charge IPv6 doit prendre en charge la configuration IP sans état conformément à la norme [RFC 4862] et la découverte de voisin conformément à la norme [RFC 4861].

Le dispositif doit prendre en charge la configuration IP dynamique conformément à la norme [RFC 2131]. Un dispositif qui prend en charge IPv6 doit prendre en charge la configuration IP avec état conformément à la norme [RFC 3315].

Le dispositif PEUT prendre en charge un mécanisme de configuration IP supplémentaire.

La configuration réseau d'un dispositif est effectuée par l'intermédiaire d'une interface réseau IP ou d'une interface locale (USB, port série, Bluetooth ou NFC, par exemple). La configuration IP par l'intermédiaire d'une interface locale est *hors* du domaine d'application de la présente norme. Des configurations IP de dispositif doivent pouvoir être effectuées en utilisant l'interface de configuration de paramètres spécifiée en 0. Un utilisateur de dispositif peut activer ou désactiver l'une des options de configuration d'adresse IP conformément à la présente norme à l'aide d'une interface de configuration réseau. Dans la configuration de dispositif par défaut, la configuration d'adresse DHCP et liaison-locale dynamique (sans état) doit être activée. Même si le dispositif est configuré par l'intermédiaire d'une configuration d'adresse statique, il convient que l'adresse de liaison-locale soit activée par défaut.

Lorsqu'un dispositif est connecté à un réseau IPv4, il convient que les priorités d'attribution d'adresse (adresse liaison-locale contre adresse acheminable) soient définies comme recommandé dans la norme [RFC 3927].

Les détails supplémentaires quant à la manière d'obtenir la connectivité IP sont *hors* du domaine d'application de la présente spécification.

7 Découverte de dispositif

7.1 Généralités

Un client recherche les dispositifs disponibles en utilisant le protocole de découverte de services Web dynamique [WS-Discovery].

Un dispositif satisfaisant à la présente norme doit mettre en œuvre le rôle Target Service tel que spécifié dans [WS-Discovery].

Un client satisfaisant à la présente norme doit mettre en œuvre le rôle Client tel que spécifié dans [WS-Discovery].

Le rôle Discovery Proxy *décrit dans* [WS-Discovery] NE doit PAS être pris en charge par un dispositif ou un client (un autre rôle Discovery Proxy est introduit dans la présente norme, voir 7.4). Un dispositif qui met en œuvre le rôle Client ignore le schéma d'interaction avec le Discovery Proxy comme décrit à l'Article 3 de [WS-Discovery]. Au lieu de cela, la présente norme définit un nouveau rôle Discovery Proxy qui permet la découverte à distance. La découverte à distance repose sur la présence d'un Discovery Proxy, et il convient qu'un fournisseur de système qui souhaite permettre la découverte à distance mette en œuvre le rôle Discovery Proxy tel que spécifié en 7.4.

[WS-Discovery] décrit l'identifiant UUID (Universally Unique Identifier): une recommandation de format URI pour un point terminal est référencée en 2.6, mais la présente spécification supprime cette recommandation. Au lieu de cela, le format Uniform Resource Name: Universally Unique Identifier (URN:UUID) est utilisé [RFC 4122] (voir 7.3.1).

7.2 Modes de fonctionnement

Le dispositif doit pouvoir fonctionner dans *deux* modes:

- découvrable;
- non découvrable.

Un dispositif en mode découvrable envoie des messages Hello en multidiffusion une fois qu'il est connecté au réseau ou envoie ses changements de statut conformément à [WS-Discovery]. De plus, il écoute toujours les messages Probe (sonde) et Resolve (Résoudre) et envoie des réponses en conséquence. Un dispositif en mode non découvrable ne doit pas écouter les messages [WS-Discovery] ni envoyer de tels messages.

Le mode découvrable doit être le comportement *par défaut* d'un dispositif. Afin de contrer les attaques de refus de service, un dispositif doit pouvoir être configuré en mode non découvrable grâce à l'opération définie en 0.

7.3 Définitions de découverte

7.3.1 Référence de point terminal

Il convient qu'un dispositif ou un point terminal qui joue le rôle de client utilise un URN:UUID [RFC 4122] comme propriété d'adresse de sa référence de point terminal.

Le dispositif ou un point terminal qui joue le rôle de client doit utiliser un identifiant stable, globalement unique et constant entre les interfaces réseau comme faisant partie de sa propriété de référence de point terminal. La combinaison d'un wsdis:Address et d'un wsdis:ReferenceProperties constitue un identifiant stable et globalement unique.

7.3.2 Hello

7.3.2.1 Types

Un dispositif doit inclure le type de port de service de gestion de dispositif, c'est-à-dire tds:Device, dans la déclaration <d:Types>.

Pour des raisons de compatibilité en amont, un dispositif satisfaisant à l'ONVIF doit également inclure ds:NetworkVideoTransmitter dans la déclaration <d:Types>.

L'exemple suivant illustre comment le type est codé dans le corps SOAP Hello:

```
<d:Types>tds:Device</d:Types>.
```

Le message Hello PEUT comprendre des types additionnels.

7.3.2.2 Domaines d'application

Un dispositif doit inclure l'attribut <d:Scopes> avec les domaines d'application du dispositif dans le message Hello.

Le domaine d'application du dispositif est défini à l'aide des URI de la norme [RFC 3986]. La présente spécification définit les attributs de domaine d'application comme suit:

L'attribut de schéma: onvif

L'attribut d'autorité: www.onvif.org

Cela implique que tous les URI de domaine d'application définis par l'ONVIF se présentent sous le format suivant:

```
onvif://www.onvif.org/<path>
```

Le dispositif PEUT avoir d'autres URI de domaine d'application. Ces URI ne sont pas limités aux domaines d'application définis par l'ONVIF.

Le Tableau 8 définit les fonctionnalités de base et d'autres propriétés du dispositif. Hormis ces paramètres normalisés, un paramètre de domaine d'application tel que défini par le propriétaire du dispositif doit pouvoir être défini. Les paramètres de domaine d'application peuvent être énumérés et définis à l'aide des commandes définies en 0. Les éditions ultérieures de la norme peuvent introduire des paramètres de domaine d'application normalisés supplémentaires.

Un dispositif PEUT avoir d'autres URI de domaine d'application. Ces URI ne sont pas limités aux domaines d'application définis par l'ONVIF.

Tableau 8 – Paramètres de domaine d'application

Catégorie	Valeurs définies	Description
type	video_encoder	Le paramètre video_encoder indique que ce dispositif est un dispositif codeur vidéo en réseau. La liste de domaines d'application d'un dispositif avec prise en charge de vidéo en réseau doit inclure le type video_encoder.
	Ptz	Un domaine d'application ptz indique qu'il s'agit d'un dispositif ptz. La liste de domaines d'application d'un dispositif avec prise en charge PTZ doit contenir une entrée de domaine d'application avec cette valeur.
	audio_encoder	Le domaine d'application audio_encoder indique que ce dispositif est un codeur audio, et que la liste de domaines d'application d'un dispositif avec prise en charge de codeur audio doit comprendre une entrée de domaine d'application avec cette valeur.
	video_analytics	Le domaine d'application video_analytics indique que ce dispositif prend en charge l'analyse vidéo telle que définie à l'Article 17. La liste de domaines d'application d'un dispositif avec prise en charge d'analyse vidéo doit contenir une entrée de domaine d'application avec cette valeur.
	Network_Video_Transmitter	Le domaine d'application network_video_transmitter indique s'il s'agit d'un dispositif satisfaisant à NVT. La liste de domaines d'application d'un NVT doit comprendre une entrée de domaine d'application avec cette valeur.
	Network_Video_Decoder	Le domaine d'application network_video_display indique s'il s'agit d'un dispositif satisfaisant à NVD. La liste de domaines d'application d'un NVD doit comprendre une entrée de domaine d'application avec cette valeur.
	Network_Video_Storage	Le domaine d'application network_video_storage indique s'il s'agit d'un dispositif satisfaisant à NVS. La liste de domaines d'application d'un NVS doit comprendre une entrée de domaine d'application avec cette valeur.
	Network_Video_Analytic	Le domaine d'application network_video_analytic indique s'il s'agit d'un dispositif satisfaisant à NVA. La liste de domaines d'application d'un NVA doit comprendre une entrée de domaine d'application avec cette valeur.
location	Une chaîne de caractères ou une valeur de chemin.	L'emplacement définit l'emplacement physique du dispositif. La valeur d'emplacement peut être une chaîne décrivant l'emplacement physique du dispositif. La liste de domaines d'application d'un dispositif doit comprendre au moins une entrée d'emplacement.
hardware	Une chaîne de caractères ou une valeur de chemin.	Chaîne ou valeur de chemin décrivant le matériel du dispositif. La liste de domaines d'application d'un dispositif doit comprendre au moins une entrée de matériel.
name	Une chaîne de caractères ou une valeur de chemin.	Nom ouvert à la recherche du dispositif. La liste de domaines d'application d'un dispositif doit comprendre au moins une entrée de nom.

La liste de domaines d'application d'un dispositif doit comprendre au moins une entrée des catégories type, location (emplacement), hardware (matériel) et name (nom), respectivement. La liste de domaines d'application d'un dispositif PEUT comprendre des attributs de domaine d'application supplémentaires.

La liste de domaines d'application d'un dispositif peut comprendre un nombre *aléatoire* de domaines d'application. Cela implique qu'une unité peut, par exemple, définir *plusieurs* domaines d'application d'emplacement *différents*. Une sonde est comparée à *tous* les domaines d'application de la liste.

Exemple

L'exemple suivant illustre l'utilisation de la valeur de domaine d'application. Ceci *n'est qu'un exemple*, et en aucune façon une indication du type de paramètre de domaine d'application

tenu de faire partie d'une configuration de NVT. Dans cet exemple, il est supposé que le NVT est configuré avec les domaines d'application suivants:

`onvif://www.onvif.org/type/Network_Video_Transmitter`

`onvif://www.onvif.org/type/video_encoder`

`onvif://www.onvif.org/type/ptz`

`onvif://www.onvif.org/type/audio_encoder`

`onvif://www.onvif.org/type/video_analytics`

`onvif://www.onvif.org/hardware/D1-566`

`onvif://www.onvif.org/location/country/china`

`onvif://www.onvif.org/location/city/beijing`

`onvif://www.onvif.org/location/building/headquarter`

`onvif://www.onvif.org/location/floor/R5`

`onvif://www.onvif.org/name/ARV-453`

Un client qui sonde le dispositif avec le domaine d'application `onvif://www.onvif.org` obtient une correspondance. De manière similaire, une sonde pour le dispositif avec le domaine d'application:

`onvif://www.onvif.org/location/country/china`

obtient une correspondance. Une sonde avec:

`onvif://www.onvif.org/hardware/D1`

n'obtient *aucune* correspondance.

7.3.2.3 Adresses

Un dispositif doit comprendre l'élément `<d:XAddrs>` avec la/les adresse(s) du service de dispositif dans le message Hello. Un URI doit être fourni pour chaque protocole (http, https), ainsi qu'une adresse IP disponible en externe.

Les principes de configuration d'adressage IP d'un dispositif sont définis en 5.12.2.2.

7.3.3 Sonde et correspondance de sonde

Pour les définitions de types de correspondance, de domaines d'application et d'adresses de sonde de dispositif, voir 7.3.2 Hello.

Le dispositif doit au moins prendre en charge la règle de mise en correspondance de domaine d'application `http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc3986`. Ces définitions de mise en correspondance de domaine d'application diffèrent légèrement de la définition de [WS-Discovery], la norme [RFC 2396] étant remplacée par la norme [RFC 3986].

Un dispositif doit inclure l'élément `<d:XAddrs>` avec les adresses du service de dispositif dans un message de correspondance de sonde. Dans la plupart des cas, l'élément

`<d:XAddr>` contient une adresse vers les interfaces de gestion et de configuration comme défini en 5.1.

7.3.4 Résolution et correspondance de résolution

La présente norme implique d'inclure des informations d'adresse de point terminal dans les messages Hello et Probe Match. Dans la plupart des cas, il n'est pas utile d'échanger des résolutions et des correspondances de résolution. Cependant, pour être compatible avec la spécification [WS-Discovery], il convient qu'un dispositif mette en œuvre la réponse de correspondance de résolution.

7.3.5 Bye

Il convient qu'un dispositif envoie un message Bye unilatéral lorsqu'il se prépare à quitter un réseau comme décrit dans WS-Discovery.

7.3.6 Messages de défaut SOAP

En cas d'erreur avec le paquet de multidiffusion, il convient que le dispositif et le client rejettent et ignorent la demande de façon silencieuse. Il n'est pas recommandé d'envoyer une réponse d'erreur en raison du risque de génération de grandes quantités de paquets si de nombreux dispositifs envoient une réponse d'erreur à la même demande. Afin d'être complet, le traitement d'un envoi de paquet à diffusion individuelle est décrit ci-dessous.

Si un dispositif reçoit un message Probe à diffusion individuelle et qu'il ne prend pas en charge la règle de mise en correspondance, il PEUT choisir de ne pas envoyer un message Probe Match, et générer plutôt un défaut SOAP conforme à SOAP 1.2, comme suit:

[Action] `http://schemas.xmlsoap.org/ws/2005/04/discovery/fault`

[Code] `s12:Sender`

[Sous-code] `d:MatchingRuleNotSupported`

[Raison] Par exemple, la règle de mise en correspondance spécifiée n'est pas prise en charge.

[Détail] `<d: SupportedMatchingRules>`

Liste de `xs:anyURI`

`<d: SupportedMatchingRules>`

Il convient que tous les défauts survenant dans une extension ou provenant de l'application soient générés conformément aux protocoles de message de défaut SOAP 1.2. Suite à la transmission d'un message de défaut SOAP à l'expéditeur, il convient d'informer l'application de la génération d'un défaut.

7.4 Extensions de découverte à distance

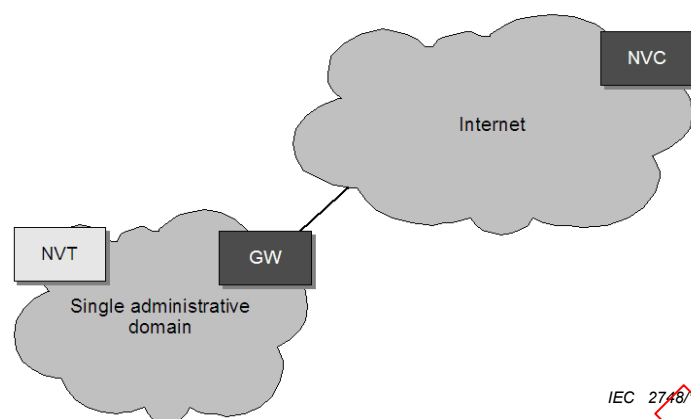
Le présent paragraphe décrit les extensions de découverte nécessaires pour couvrir des scénarios de réseau plus complexes. Ces extensions *ne sont pas* requises par les points terminaux satisfaisant à l'ONVIF. Un dispositif qui prend en charge la découverte de service à distance doit prendre en charge les extensions de découverte définies dans le présent paragraphe.

Les extensions de découverte à distance définies dans le présent paragraphe peuvent être utilisées *conjointement* avec le schéma de base de multidiffusion WS-Discovery ordinaire, comme défini dans la présente norme. Par exemple, les extensions de découverte à distance peuvent fonctionner en parallèle avec la découverte "locale" ordinaire.

7.4.1 Scénarios de réseau

Si le client et le dispositif ne résident *pas* dans le même domaine administratif, le client ne peut pas trouver *et* se connecter au dispositif à l'aide d'une sonde à multidiffusion. Par exemple, si le dispositif ou le client résident dans un réseau derrière un pare-feu ou un NAT (passerelle GW), il ne peut pas se connecter à une sonde à multidiffusion. D'autres méthodes sont alors nécessaires et la spécification utilise quatre scénarios différents:

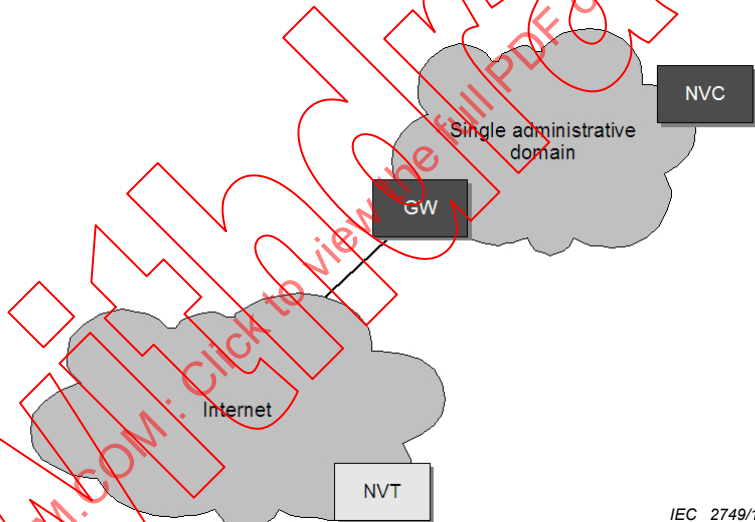
- 1) le dispositif réside dans un domaine administratif (privé) et le client dans un réseau public (voir Figure 8);
- 2) le dispositif réside dans un réseau public et le client dans un domaine administratif (privé) (voir Figure 9);
- 3) le dispositif réside dans un domaine administratif (privé) et le client dans un *autre* domaine administratif (privé) (voir Figure 10);
- 4) le dispositif et le client résident tous deux dans un réseau public (voir Figure 11).



Légende

Anglais	Français
Single administrative domain	Domaine administratif unique

Figure 8 – Dispositif (NVT, par exemple) dans un domaine administratif (privé) et client (NVC) dans un réseau public



Légende

Anglais	Français
Single administrative domain	Domaine administratif unique

Figure 9 – Dispositif (NVT, par exemple) dans un réseau public et client (NVC) dans un domaine administratif (privé)

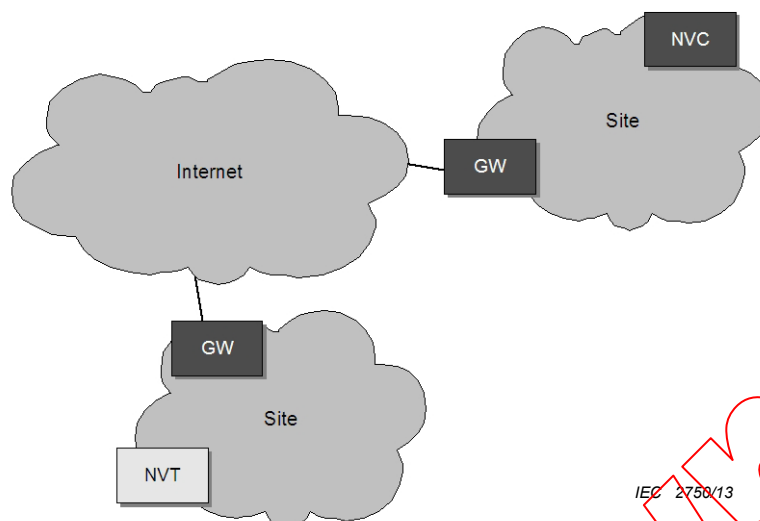


Figure 10 – Dispositif (NVT, par exemple) dans un domaine administratif (privé) et client (NVC) dans un autre domaine administratif (privé)

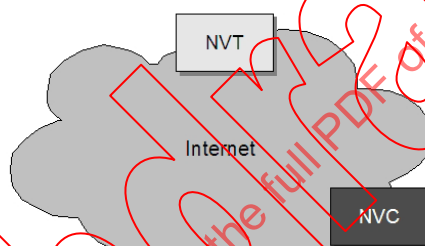


Figure 11 – Dispositif (NVT, par exemple) et client (NVC) dans un réseau public

La spécification [WS-Discovery] introduit un "proxy de découverte" (Discovery *Proxy*: DP) pour résoudre certains de ces scénarios. Cependant, la spécification [WS-Discovery] ne prend pas en charge tous les scénarios de réseau introduits dans la présente spécification. Cette spécification définit un DP qui permet le "plug and play" (prêt à l'emploi) pour les scénarios de réseau plus complexes mentionnés ci-dessus. Ce DP ne satisfait pas à la spécification [WS-Discovery].

7.4.2 Proxy de découverte (DP)

Il est nécessaire qu'un administrateur réseau configurant un réseau pour un réseau étendu de NVT couvrant plusieurs domaines administratifs introduise un point terminal DP dans le système. Le DP effectue les tâches suivantes:

- 1) écoute des messages Hello de dispositif et réponse comme défini en 7.4.3;
- 2) réponse aux requêtes de sonde au nom de dispositifs enregistrés provenant de clients.

Le DP peut résider dans le même domaine administratif que le dispositif. Afin de prendre en charge des scénarios de réseau dans lesquels le client et le dispositif résident dans des domaines différents sans connectivité multidiffusion, placer le DP dans un réseau publiquement disponible, de sorte que les points terminaux de dispositif et de client puissent y accéder. Le dispositif doit pouvoir trouver l'adresse réseau de son "DP domestique" afin d'annoncer sa présence avec un message Hello *directement* envoyé à son DP domestique. Conformément à la présente norme, l'adresse réseau du DP domestique peut être obtenue des manières suivantes:

- 1) configuration d'adresse directe;

2) découverte de DP en utilisant la recherche d'enregistrement de service (SRV) DNS.

Le dispositif tente de se connecter à un DP domestique après avoir obtenu une connectivité de réseau ou en cas de modification de l'adresse réseau du DP domestique par l'une de ces méthodes.

L'enregistrement de découverte de dispositif doit pouvoir être activé/désactivé à distance. Un dispositif prenant en charge la découverte à distance doit mettre en œuvre l'opération d'activation/de désactivation de message Hello comme défini en 0.

Un dispositif qui n'est pas configuré avec une adresse de DP domestique ou dont la fonction Hello à distance est désactivée NE doit PAS envoyer un message Hello à distance comme défini en 7.4.3.

7.4.2.1 Configuration d'adresse DP directe

La présente norme introduit une commande de gestion de dispositif pour la configuration d'adresse de DP domestique sur l'interface réseau (voir 0 et 0).

Un dispositif qui prend en charge la découverte à distance PEUT également assurer la configuration locale de l'adresse de DP domestique. Ces configurations sont effectuées par l'intermédiaire d'une interface locale choisie du dispositif (un port série ou une interface USB, par exemple). Ce type de configuration locale est *hors du domaine d'application* de la présente norme.

7.4.2.2 Recherche d'enregistrement de service DNS

Si la fonction de découverte à distance est activée pour un dispositif qui *ne dispose pas* de la configuration d'adresse de DP distant, le dispositif doit tenter d'effectuer une recherche de SRV DNS pour le DP domestique. La définition de nom d'enregistrement et de protocole suivante [RFC 2782] doit être utilisée:

`_onvifdiscover._tcp`

Afin d'éviter que le dispositif ne procède à une recherche de SRV DNS, une adresse DP doit être configurée en utilisant la configuration d'adresse directe avant d'activer la découverte à distance.

Afin que des dispositifs puissent effectuer une recherche de DP pour d'autres dispositifs, un administrateur doit entrer l'adresse, le port et la priorité de DP dans le DNS en utilisant des SRV. Il est nécessaire qu'un ou plusieurs serveurs de recrutement soient présents. Le nombre exact dépend de la charge du système et est *hors du domaine d'application* de la présente norme.

7.4.3 Comportement pour les messages Hello et Probe à distance

Le schéma de découverte locale défini dans [WS-Discovery] ne fonctionne pas pour les scénarios de découverte à distance. Si le dispositif réside derrière un NAT/pare-feu, comme dans les scénarios de la Figure 8 ou de la Figure 10, une sonde à diffusion unilatérale provenant du DP n'atteint pas automatiquement le dispositif si celui-ci ne retourne pas une adresse de réseau public. De plus, si le dispositif réside derrière un pare-feu, le dispositif suivant la diffusion unilatérale de message Probe Match ne pourrait pas retourner au DP. La spécification définit un schéma de communication légèrement différent pour la découverte à distance afin de résoudre ce problème.

Outre la multidiffusion de messages Hello lorsqu'il se connecte à un réseau ou que ses métadonnées sont modifiées, un dispositif configuré pour les messages Hello à distance envoie ce type de message à son DP domestique. Ce message est envoyé par le dispositif au DP sous la forme d'une opération de demande de services Web en utilisant la liaison HTTP

définie en [C.12]. La liste de domaines d'application du message Hello à distance doit être incluse dans le message Hello.

Lorsque le DP domestique reçoit un message Hello d'un dispositif, il envoie un message Hello de réponse confirmant l'enregistrement de dispositif par l'intermédiaire du message Hello. De même, si un dispositif se prépare à quitter un réseau, il convient qu'il envoie une demande Bye au DP distant. Le DP accuse réception de la demande Bye par un message de réponse Bye. Les messages Hello de DP, de réponse Hello, Bye et de réponse Bye sont fournis dans le cadre d'un service DP (voir [C.12] pour les définitions de WSDL). Grâce à ces extensions, les messages de découverte peuvent atteindre les points terminaux souhaités, comme indiqué à la Figure 12.

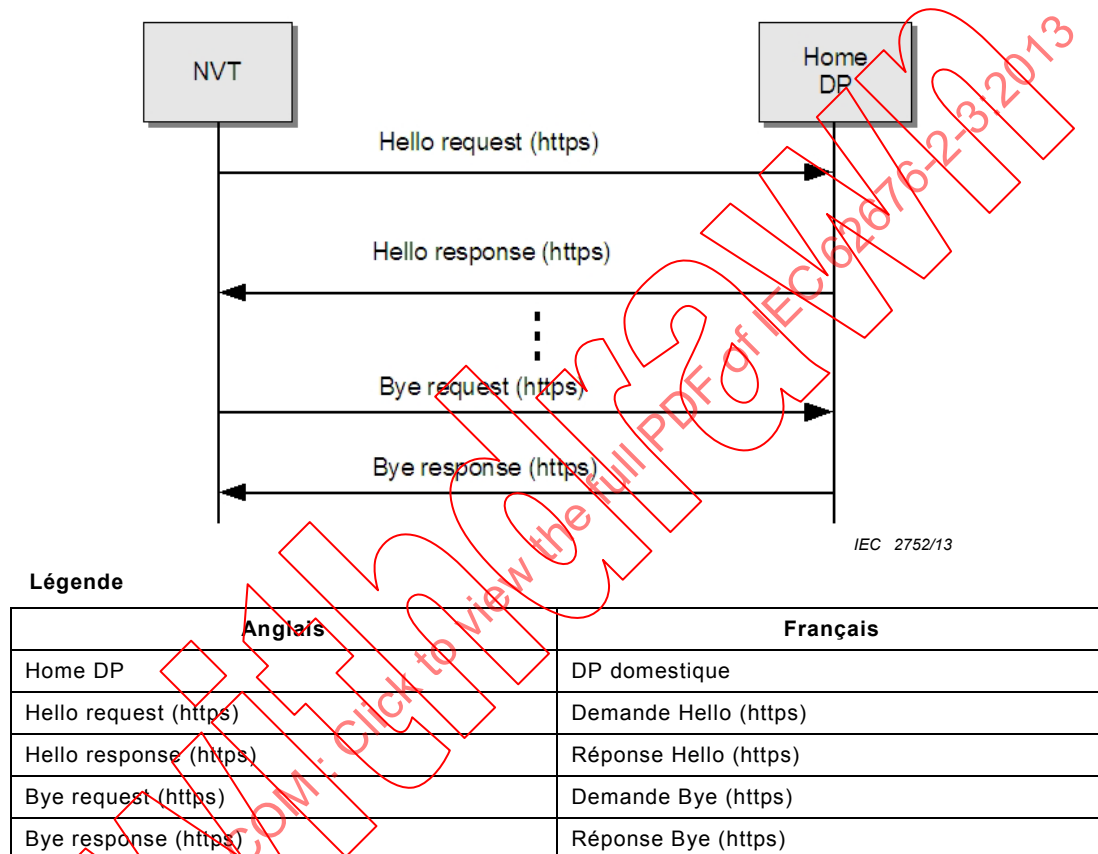


Figure 12 – Schéma d'échange de message de découverte à distance entre un dispositif (NVT, par exemple) et un DP domestique

7.4.4 Comportement du client

7.4.4.1 Généralités

Pour les scénarios de découverte à distance, il est nécessaire que le client envoie des messages de sonde au DP domestique. Il est alors nécessaire que le client soit configuré de manière à pouvoir se connecter directement au DP domestique.

7.4.4.2 Configuration de DP domestique de NVC

Le client peut être configuré pour rechercher directement de nouveaux dispositifs par l'intermédiaire du DP domestique. Dans ce cas, l'adresse de service de découverte de DP domestique doit être préconfigurée dans le client. La signification exacte de cette configuration est hors du domaine d'application de la présente norme.

Un client configuré pour la découverte à distance envoie des demandes de sonde directement à son DP domestique. Le message de sonde est envoyé par le client au DP sous la forme d'une opération de demande de services Web en utilisant la liaison http (voir C.12).

Si le DP domestique reçoit un message Probe provenant d'un client, il répond avec un message Probe Match correspondant conformément au schéma d'échange WS-Discovery normal (voir le schéma de séquence à la Figure 13).

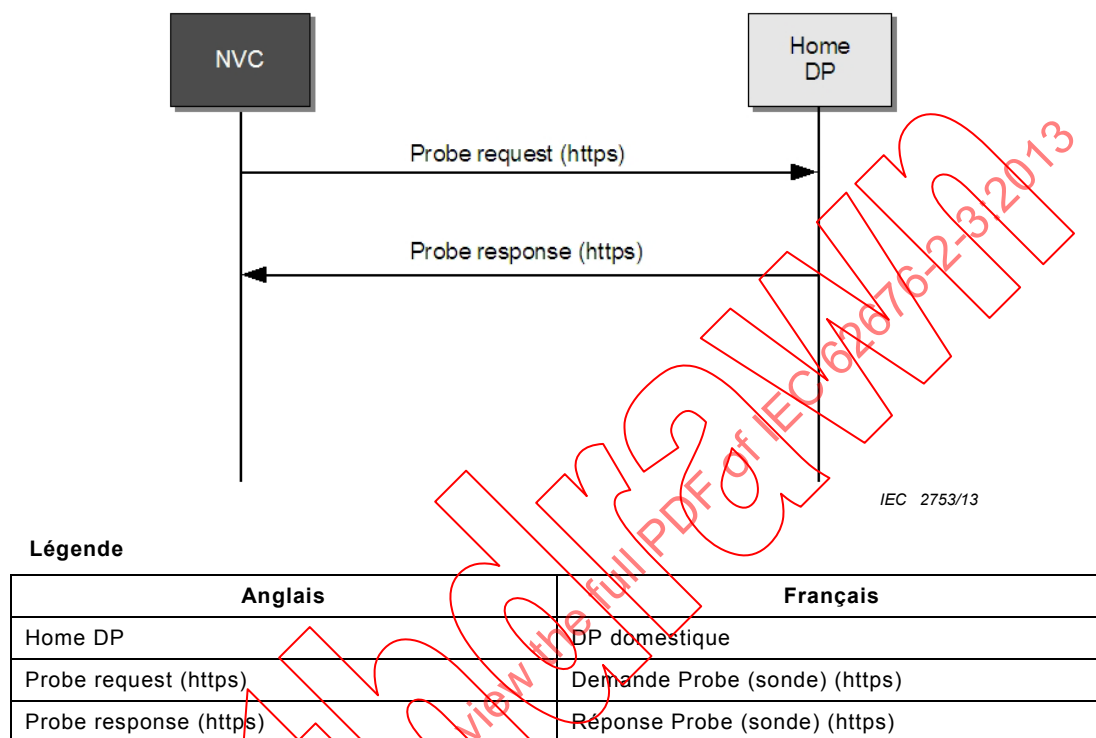


Figure 13 – Séquence de messages pour des clients (NVC) préconfigurés avec l'adresse de DP domestique

7.4.5 Sécurité

7.4.5.1 Découverte locale

La sécurité et la découverte peuvent être considérées comme des objectifs contradictoires. Bien que l'idée sous-jacente d'un protocole de découverte soit d'annoncer la présence d'un service, il est difficile d'exclure d'autres points terminaux de l'accès aux annonces de service. WS-Discovery ne donne pas d'accès supplémentaire aux services (si les autres mécanismes de sécurité définis dans la présente spécification sont utilisés), même sur le même réseau local. Il annonce simplement leur existence. De plus, la découverte locale fonctionne uniquement dans la portée de multidiffusion. Par conséquent, le principal impact de sécurité de WS-Discovery est le risque d'attaques de refus de service sur des dispositifs ou des problèmes de confidentialité, s'il est important de masquer la présence de dispositifs dans le réseau. Le risque des deux problèmes ci-dessus dépend en grande partie de l'environnement de déploiement du dispositif. Afin de réduire ces menaces, la présente norme a introduit les deux modes de découverte différents (voir 7.2). Cela donne toujours la possibilité au client de désactiver la fonction de découverte de dispositif dans le dispositif. En mode non découvrable, un dispositif n'annonce jamais sa présence avec des messages Hello et ne répond jamais à des demandes Probe ou Resolve.

7.4.5.2 Découverte à distance

Dans le scénario de réseau distant, le DP réside sur Internet et est vulnérable. Par conséquent, des mesures de sécurité supplémentaires doivent être prises pour protéger le DP contre les attaques. Les messages Hello, Probe et Probe Match à distance, tels que définis

en 7.4.3, doivent être envoyés sur HTTPS. Ce transport n'empêche *pas* les attaques de refus de service, mais il peut protéger contre les enregistrements de dispositif illégaux si l'authentification du client est utilisée. Si la protection contre les refus de service est un souci majeur, il est nécessaire de prendre d'autres mesures, qui sont hors du domaine d'application de la présente norme.

Avant d'enregistrer un dispositif dans la base de données de dispositif, il convient que le DP l'authentifie afin de s'assurer qu'il s'agit d'un dispositif "légal" annonçant sa présence (en utilisant des certificats de client, par exemple). La mise en œuvre de certificats de client est hors du domaine d'application de la présente norme.

Les messages Probe et Probe Match du client au DP à distance doivent être envoyés sur HTTPS. Le DP doit authentifier le NVC avant de répondre à une demande Probe. Cela peut être effectué en utilisant des certificats de client TLS ou un autre mécanisme d'authentification de client adapté.

8 Gestion de dispositif

Le service de dispositif est divisé en cinq catégories différentes: commandes de fonctionnalités, de réseau, de système, d'E/S et de sécurité. Cet ensemble de commandes peut être utilisé pour obtenir des informations sur les fonctionnalités et les configurations du dispositif ou pour définir des configurations de dispositif. Un dispositif doit prendre en charge le service de gestion du dispositif tel que spécifié en C.4. Un ensemble d'opérations de base est requis pour le service de gestion de dispositif, la prise en charge d'autres opérations étant recommandée ou facultative. Les exigences détaillées sont présentées sous les descriptions de commande.

8.1 Fonctionnalités

8.1.1 Get WSDL URL

Un point terminal peut demander une URL qui peut être utilisée pour extraire le schéma et les définitions WSDL complets d'un dispositif. La commande génère en retour un point d'entrée d'URL où toutes les définitions WSDL et de schéma spécifiques au produit nécessaires peuvent être extraites. Le dispositif doit fournir une URL pour le téléchargement WSDL et de schéma par l'intermédiaire de la commande GetWsdUrl (voir Tableau 9).

Tableau 9 – Commande Get WSDL URL

GetWsdUrl		Demande-Réponse
Nom du message	Description	
GetWsdUrlRequest	<i>Ceci est un message vide.</i>	
GetWsdUrlResponse	<i>L'URL demandée.</i> xs:anyURI WsdUrl [1][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.1.2 Echange de fonctionnalité

Un point terminal peut demander les fonctionnalités d'un dispositif à l'aide de l'opération de réponse de demande d'échange de fonctionnalité. Le dispositif doit indiquer toutes ses

fonctionnalités satisfaisant à l'ONVIF à l'aide de la commande GetCapabilities (voir Tableau 10).

La liste de fonctionnalités comprend des références aux adresses (XAddr) du service qui met en œuvre les opérations d'interface dans la catégorie.

Le Tableau 11 décrit la manière d'interpréter la fonctionnalité indiquée. Outre les adresses, les fonctionnalités reflètent uniquement les fonctions facultatives de la présente norme.

Tableau 10 – Commande GetCapabilities

GetCapabilities		Demande Réponse
Nom du message	Description	
GetCapabilitiesRequest	<i>Ce message contient une demande de fonctionnalités de dispositif. Le client peut demander toutes les fonctionnalités ou uniquement les fonctionnalités d'une catégorie de service particulière. Si aucune catégorie n'est spécifiée, le dispositif doit renvoyer toutes les fonctionnalités.</i> tt:CapabilityCategory Category [0][non limité]	
GetCapabilitiesResponse	<i>Le message de réponse de fonctionnalité contient les fonctionnalités de dispositif demandées en utilisant une structure de capacité XML hiérarchique.</i> tt:Capabilities Capabilities [1][1]	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:NoSuchService	<i>La catégorie de service WSDL demandée n'est pas prise en charge par le dispositif.</i>	

Tableau 11 – Fonctionnalités dans la commande GetCapabilities

Catégorie	Fonctionnalité	Description
Analyses	XAddr	Adresse du service d'analyse. Si ce champ est vide, le dispositif prend en charge l'analyse, mais pas les règles ou les interfaces de module.
	RuleSupport	Indique si le dispositif prend en charge l'interface de règles et la syntaxe de règles comme spécifié en 17.2.
	AnalyticsModuleSupport	Indique si le dispositif prend en charge l'interface de module d'analyse de scène comme spécifié en 0.
Dispositif	XAddr	Adresse du service de dispositif.
Dispositif – Réseau	IPFilter	Indique si le dispositif prend en charge la commande de filtrage IP en utilisant les commandes de 0, 0, 0 et 0.
	ZeroConfiguration	Indique si le dispositif prend en charge la configuration zéro conformément aux commandes de 0 et de 0.
	IPVersion6	Indique si le dispositif prend en charge IP version 6.
	DynDNS	Indique si le dispositif prend en charge la configuration DNS dynamique selon 0 et selon 0.
	Dot11Configuration	Indique si le dispositif prend en charge la configuration IEEE 802.11 comme spécifié en 0.
Dispositif – Système	DiscoveryResolve	Indique si le dispositif répond aux demandes de résolution comme décrit en 7.3.4.
	DiscoveryBye	Indique si le dispositif envoie des messages Bye comme décrit en 7.3.5.
	RemoteDiscovery	Indique si le dispositif prend en charge la découverte à distance comme spécifié en 7.4.
	SupportedVersions	Liste des versions de spécification ONFIV prises en charge par le dispositif.
	SystemBackup	Indique si le dispositif prend en charge la sauvegarde et la restauration du système comme spécifié en 0 et en 0.
	FirmwareUpgrade	Indique si le dispositif prend en charge la mise à niveau de micrologiciel comme spécifié en 0.
	SystemLogging	Indique si le dispositif prend en charge l'extraction de journal système comme spécifié en 0.
	HttpSystemBackup	Indique si le dispositif prend en charge la sauvegarde et la restauration du système à l'aide de HTTP GET et de POST.
	HttpFirmwareUpgrade	Indique si le dispositif prend en charge la mise à niveau de micrologiciel à l'aide de HTTP POST.
	HTTPSystemLogging	Indique si le dispositif prend en charge l'extraction de journal système à l'aide de HTTP GET (voir 8.3.2).
	HTTPSupportInformation	Indique si le dispositif prend en charge l'extraction d'informations de support à l'aide de HTTP GET (voir 8.3.2).
Dispositif – ES	InputConnectors	Nombre de connecteurs d'entrée.
	RelayOutputs	Nombre de sorties de relais.
	Auxiliary	Indique la prise en charge de service auxiliaire avec une liste des commandes auxiliaires prises en charge
Dispositif – Sécurité	TLS1.0	Prise en charge de TLS 1.0.
	TLS1.1	Prise en charge de TLS 1.1.
	TLS1.2	Prise en charge de TLS 1.2.
	OnboardKeyGeneration	Indique si le dispositif prend en charge la génération de clé intégrée et la création de certificats auto-signés comme spécifié en 8.4.8.
	AccessPolicyConfig	Indique si le dispositif prend en charge l'extraction et le chargement de règles de contrôle d'accès de dispositif conformément en 8.4.1 et en 0.

Catégorie	Fonctionnalité	Description
	X.509Token	Indique si le dispositif prend en charge le jeton WS-Security X.509 [WS-X.509Token].
	SAMLTToken	Indique si le dispositif prend en charge le jeton WS-Security SAML [WS-SAMLTToken].
	KerberosToken	Indique si le dispositif prend en charge le jeton WS-Security Kerberos [WS-KerberosToken].
	RELTToken	Indique si le dispositif prend en charge le jeton WS-Security REL [WS-RELTToken].
	Dot1X	Indique si le dispositif prend en charge l'authentification de réseau basée sur le port IEEE 802.1X
	SupportedEAPMethod	Liste des types de méthode EAP pris en charge. Les numéros correspondent à l'IANA [EAP-Registry].
	RemoteUserHandling	Indique si le dispositif prend en charge le traitement utilisateur distant et les méthodes correspondantes définies en 0 et en 8.4.22.
Evénement	XAddr	Adresse du service d'événement
	WSSubscriptionPolicySupport	Indique si le dispositif prend en charge la politique WS Subscription (abonnement) selon 15.1.2
	WSPullPointSupport	Indique si le dispositif prend en charge la politique WS PullPoint selon 15.1.2
	WSPausableSubscription-ManagerInterfaceSupport	Indique si le dispositif prend en charge l'interface WS Pausable Subscription Manager selon 15.1.2
Imagerie	XAddr	Adresse du service d'imagerie
Multimédia	XAddr	Adresse du service multimédia.
Multimédia – Transmission continue	RTPMulticast	Indique la prise en charge de la multidiffusion UDP décrite en 12.1.1.1.
	RTP_TCP	Indique si le dispositif prend en charge RTP sur TCP (voir 12.1.1.2).
	RTP_RTSP_TCP	Indique si le dispositif prend en charge le transport RTP/RTSP/TCP (voir 12.1.1.3).
Multimédia - Profil	MaximumNumberOfProfiles	Nombre maximal de MediaProfiles pris en charge par le dispositif.
PTZ	XAddr	Adresse du service PTZ.
Récepteur	XAddr	Adresse du service de récepteur.
	RTP_Multicast	Indique si le dispositif prend en charge la réception de la multidiffusion RTP.
	RTP_TCP	Indique si le dispositif prend en charge la réception de RTP sur TCP.
	RTP_RTSP_TCP	Indique si le dispositif prend en charge la réception de RTP sur RTSP sur TCP.
	SupportedReceivers	Nombre maximal de récepteurs que le dispositif prend en charge.
	MaximumRTSPURILength	Longueur maximale admise des URI RTSP.
Enregistrement	XAddr	Adresse du service de contrôle d'enregistrement.
	DynamicRecordings	Indique si le dispositif prend en charge la création et la suppression dynamiques des enregistrements (voir 19.4 et 19.5).
	DynamicTracks	Indique si le dispositif prend en charge la création et la suppression dynamiques des pistes (voir 19.9 et 19.10).
	DeleteData	Indique si le dispositif prend en charge la suppression explicite des données (voir 19.5).
Recherche	XAddr	Adresse du service de recherche d'enregistrement.

Catégorie	Fonctionnalité	Description
	MetadataSearch	Indique si le dispositif prend en charge la recherche générique des métadonnées enregistrées (voir 20.13 et 20.14).
Lecture	XAddr	Adresse du service de lecture.
Dispositif d'analyse	XAddr	Adresse du service de dispositif d'analyse du dispositif.
Affichage	XAddr	Adresse du service d'affichage.
Affichage - Présentation	FixedLayout	Indique si le dispositif comporte un certain nombre de présentations prédéfinies.
Dispositif ES	XAddr	Adresse du service de dispositif ES.
	VideoSources	Nombre d'entrées vidéo.
	VideoOutputs	Nombre de sorties vidéo.
	AudioSources	Nombre d'entrées audio.
	AudioOutputs	Nombre de sorties audio.
	RelayOutputs	Nombre de sorties de relais.

8.2 Réseau

8.2.1 Obtention de nom d'hôte

Cette opération est utilisée par un point terminal pour obtenir le nom d'hôte d'un dispositif. Le dispositif doit retourner ses configurations de nom d'hôte par l'intermédiaire de la commande GetHostname (voir Tableau 12).

Tableau 12 – Commande GetHostname

GetHostname	Demande-Réponse
Nom du message	Description
GetHostnameRequest	<i>Ceci est un message vide.</i>
GetHostnameResponse	<i>Ce message contient:</i> <i>"FromDHCP": True (Vrai) si le nom d'hôte est obtenu via DHCP</i> <i>"Name": nom de l'hôte. Dans le cas de DHCP, le nom d'hôte a été obtenu à partir du serveur DHCP.</i> xs:boolean FromDHCP [1][1] xs:token Name [0][1]
Codes de défaut	Description
	<i>Pas de défauts spécifiques à la commande!</i>

8.2.2 Définition de nom d'hôte

Cette opération définit le nom d'hôte sur un dispositif. Les configurations de nom d'hôte du dispositif doivent pouvoir être définies grâce à la commande SetHostname (voir Tableau 13). Attention: un appel adressé à SetDNS peut écraser un nom d'hôte déjà défini.

Tableau 13 – Commande SetHostname

SetHostname		Demande-Réponse
Nom du message	Description	
SetHostnameRequest	<i>Ce message contient:</i> <i>"Name": nom de l'hôte.</i> xs:token Name [1][1]	
SetHostnameResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:InvalidHostname	<i>Le dispositif ne peut pas accepter le nom d'hôte demandé.</i>	

8.2.3 Obtention de paramètres DNS

Cette opération permet d'obtenir les paramètres DNS d'un dispositif. Le dispositif doit retourner ses configurations DNS grâce à la commande GetDNS (voir Tableau 14).

Tableau 14 – Commande GetDNS

GetDNS		Demande-Réponse													
Nom du message	Description														
GetDNSRequest	Ceci est un message vide.														
GetDNSResponse	Ce message contient: "FromDHCP": True (vrai) si les serveurs DNS sont obtenus via DHCP. "SearchDomain": Domaine(s) à rechercher si le nom d'hôte n'est pas totalement qualifié. "DNSFromDHCP": liste de serveurs DNS obtenue via DHCP dans le cas où FromDHCP est égal à True (Vrai). Cela signifie que les adresses résolues dans le champ DNSFromDHCP proviennent de DHCP et décrivent le statut de configuration. "DNSManual": Liste des serveurs DNS manuellement spécifiés xs:boolean FromDHCP [1][1] <table><tr><td>xs:token</td><td>SearchDomain</td><td>[0][non</td><td>limité]</td></tr><tr><td>tt:IPAddress</td><td>DNSFromDHCP</td><td>[0][non</td><td>limité]</td></tr><tr><td>tt:IPAddress</td><td>DNSManual</td><td>[0][non</td><td>limité]</td></tr></table>			xs:token	SearchDomain	[0][non	limité]	tt:IPAddress	DNSFromDHCP	[0][non	limité]	tt:IPAddress	DNSManual	[0][non	limité]
xs:token	SearchDomain	[0][non	limité]												
tt:IPAddress	DNSFromDHCP	[0][non	limité]												
tt:IPAddress	DNSManual	[0][non	limité]												
Codes de défaut	Description														
	Pas de défauts spécifiques à la commande!														

8.2.4 Définition des paramètres DNS

Cette opération permet de définir les paramètres DNS d'un dispositif. Les configurations DNS du dispositif doivent pouvoir être définies grâce à la commande SetDNS (voir Tableau 15).

Tableau 15 – Commande SetDNS

SetDNS		Demande-Réponse
Nom du message	Description	
SetDNSRequest	<i>Ce message contient:</i> <i>"FromDHCP": True (vrai) si les serveurs DNS sont obtenus via DHCP</i> <i>"SearchDomain": Domaine(s) à rechercher si le nom d'hôte n'est pas totalement qualifié.</i> <i>"DNSManual": Liste des serveurs DNS manuellement spécifiés</i> xs:boolean FromDHCP [1][1] xs:token SearchDomain [0][non limité] tt:IPAddress DNSManual [0][non limité]	
SetDNSResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>L'adresse IPv6 suggérée est non valide.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>L'adresse IPv4 suggérée est non valide.</i>	

8.2.5 Obtention des paramètres NTP

Cette opération permet d'obtenir les paramètres NTP d'un dispositif. Si le dispositif prend en charge NTP, les paramètres de serveur NTP doivent pouvoir être obtenus grâce à la commande GetNTP (voir Tableau 16).

Tableau 16 – Commande GetNTP

GetNTP		Demande-Réponse	
Nom du message	Description		
GetNTPRequest	Ceci est un message vide.		
GetNTPResponse	Ce message contient: "FromDHCP": True (vrai) si les serveurs NTP sont obtenus via DHCP. "NTPFromDHCP": liste de serveurs NTP obtenus via DHCP dans le cas où FromDHCP est vrai. Cela signifie que les adresses de serveur NTP dans le champ NTPFromDHCP proviennent de DHCP et décrivent le statut de configuration actuel. "NTPManual": Liste de serveurs NTP manuellement spécifiés xs:boolean FromDHCP [1][1] tt:NetworkHost NTPFromDHCP [0][non limité] tt:NetworkHost NTPManual [0][non limité]		
Codes de défaut	Description		
	Pas de défauts spécifiques à la commande!		

8.2.6 Définition des paramètres NTP

Cette opération permet de définir les paramètres NTP d'un dispositif. Si le dispositif prend en charge NTP, les paramètres de serveur NTP doivent pouvoir être définis grâce à la commande SetNTP (voir Tableau 17).

Tableau 17 – Commande SetNTP

SetNTP		Demande-Réponse
Nom du message	Description	
SetNTPRequest	<i>Ce message contient:</i> <i>"FromDHCP": True (vrai) si les serveurs NTP sont obtenus via DHCP.</i> <i>"NTPManual": Liste de serveurs NTP manuellement spécifiés lorsqu'ils ne sont pas obtenus via DHCP.</i> xs:boolean FromDHCP [1][1] tt:NetworkHost NTPManual [0][non limitée]	
SetNTPResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>L'adresse IPv4 suggérée est non valide.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>L'adresse IPv6 suggérée est non valide.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDnsName	<i>Le nom de serveur NTP suggéré est non valide.</i>	

8.2.7 Obtention des paramètres de DNS dynamiques

Cette opération permet d'obtenir les paramètres DNS dynamiques d'un dispositif. Si le dispositif prend en charge un DNS dynamique comme spécifié dans la norme [RFC 2136] et la norme [RFC 4702], les types, nom et TTL doivent pouvoir être obtenus grâce à la commande GetDynamicDNS (voir Tableau 18).

Tableau 18 – Commande GetDynamicDNS

GetDynamicDNS		Demande-Réponse
Nom du message	Description	
GetDynamicDNSRequest	<i>Ceci est un message vide.</i>	
GetDynamicDNSResponse	<i>Ce message contient:</i> <i>"Type": type de mise à jour. Il existe trois types possibles: le dispositif ne veut pas de mise à jour (NoUpdate), le dispositif souhaite que le serveur DHCP effectue une mise à jour (ServerUpdates) et le dispositif effectue lui-même la mise à jour (ClientUpdates).</i> <i>"Name": Nom de DNS au cas où le dispositif effectue la mise à jour.</i> <i>"TTL": Durée de vie.</i> <i>tt:DynamicDNSType Type [1][1]</i> <i>tt:DNSName Name [0][1]</i> <i>xs:duration TTL [0][1]</i>	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.2.8 Définition des paramètres de DNS dynamiques

Cette opération permet de définir les paramètres DNS dynamiques d'un dispositif. Si le dispositif prend en charge le DNS dynamique comme spécifié dans la norme [RFC 2136] et la norme [RFC 4702], les types, nom et TTL doivent pouvoir être obtenus grâce à la commande SetDynamicDNS (voir Tableau 19).

Tableau 19 – Commande SetDynamicDNS

SetDynamicDNS		Demande-Réponse	
Nom du message	Description		
SetDynamicDNSRequest	<i>Ce message contient:</i> <i>"Type": type de mise à jour. Il existe trois types possibles: le dispositif ne veut pas de mise à jour (NoUpdate), le dispositif souhaite que le serveur DHCP effectue une mise à jour (ServerUpdates) et le dispositif effectue lui-même la mise à jour (ClientUpdates).</i> <i>"Name": Nom de DNS au cas où le dispositif effectue la mise à jour.</i> <i>"TTL": Durée de vie.</i> <i>tt:DynamicDNSType Type [1][1]</i> <i>tt:DNSName Name [0][1]</i> <i>xs:duration TTL [0][1]</i>		
SetDynamicDNSResponse	Ceci est un message vide.		
Codes de défaut	Description		
	Pas de défauts spécifiques à la commande!		

8.2.9 Obtention de configuration d'interface réseau

Cette opération permet d'extraire la configuration d'interface réseau d'un dispositif. Le dispositif doit prendre en charge le retour des paramètres de configuration d'interface réseau défini par le type NetworkInterface grâce à la commande GetNetworkInterfaces (voir Tableau 20).

Tableau 20 – Commande GetNetworkInterfaces

GetNetworkInterfaces		Demande-Réponse
Nom du message	Description	
GetNetworkInterfacesRequest	Ceci est un message vide.	
GetNetworkInterfacesResponse	Ce message contient un ensemble d'interfaces réseau de dispositif. tt:NetworkInterface NetworkInterfaces [0][non limité]	
Codes de défaut	Description	
	Pas de défauts spécifiques à la commande!	

8.2.10 Définition de configuration d'interface réseau

Cette opération permet de définir la configuration d'interface réseau d'un dispositif. Le dispositif doit prendre en charge la configuration réseau d'interfaces réseau prises en charge grâce à la commande SetNetworkInterfaces (voir Tableau 21).

Pour assurer l'interopérabilité avec un client n'ayant pas connaissance de l'extension IEEE 802.11, un dispositif doit conserver sa configuration IEEE 802.11 si l'élément de configuration IEEE 802.11 est absent de la demande.

Tableau 21 – Commande SetNetworkInterfaces

SetNetworkInterfaces		Demande-Réponse
Nom du message	Description	
SetNetworkInterfacesRequest	<i>Ce message contient:</i> <i>"InterfaceToken": Jeton de l'interface réseau cible.</i> <i>"NetworkInterface": Interface réseau à configurer.</i> tt:ReferenceToken InterfaceToken [1][1] tt:NetworkInterfaceSetConfiguration NetworkInterface [1][1]	
SetNetworkInterfacesResponse	<i>Ce message contient:</i> <i>"RebootNeeded": Indique si un redémarrage est nécessaire en cas de modifications des paramètres réseau.</i> xs:boolean RebootNeeded [1][1]	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface		Le jeton d'interface réseau fourni n'existe pas.
env:Sender ter:InvalidArgVal ter:InvalidMtuValue		La valeur MTU est non valide.
env:Sender ter:InvalidArgVal ter:InvalidInterfaceSpeed		La vitesse suggérée n'est pas prise en charge.
env:Sender ter:InvalidArgVal ter:InvalidInterfaceType		Le type d'interface réseau suggéré n'est pas pris en charge.
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address		L'adresse IPv4 suggérée est non valide.

SetNetworkInterfaces		Demande-Réponse
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>L'adresse IPv6 suggérée est non valide.</i>	
env:Receiver ter:ActionNotSupported ter:InvalidDot11	<i>La configuration IEEE 802.11 n'est pas prise en charge.</i>	
env:Sender ter:InvalidArgVal ter:InvalidSecurityMode	<i>Le mode de sécurité sélectionné n'est pas pris en charge.</i>	
env:Sender ter:InvalidArgVal ter:InvalidStationMode	<i>Le mode de station sélectionné n'est pas pris en charge.</i>	
env:Sender ter:InvalidArgVal ter:MissingDot11	<i>La configuration de sécurité ne contient pas de valeur IEEE 802.11.</i>	
env:Sender ter:InvalidArgVal ter:MissingPSK	<i>La configuration de sécurité ne contient pas de valeur PSK.</i>	
env:Sender ter:InvalidArgVal ter:MissingDot1X	<i>La configuration de sécurité ne contient pas de valeur IEEE 802.1X.</i>	
env:Sender ter:InvalidArgVal ter:IncompatibleDot1X	<i>La valeur IEEE 802.1X de la configuration de sécurité n'est pas compatible avec l'interface réseau.</i>	

8.2.11 Obtention de protocoles réseau

Cette opération extrait des protocoles réseau définis d'un dispositif. Le dispositif doit prendre en charge la commande GetNetworkProtocols retournant les protocoles réseau configurés (voir Tableau 22).

Tableau 22 – Commande GetNetworkProtocols

GetNetworkProtocols		Demande-Réponse
Nom du message	Description	
GetNetworkProtocolsRequest	<i>Ceci est un message vide.</i>	
GetNetworkProtocols-Response	<i>Ce message retourne un ensemble de protocoles définis pris en charge par le dispositif. Il existe trois protocoles définis, HTTP, HTTPS et RTSP. Les paramètres suivants peuvent être extraits pour chaque protocole:</i> <i>Port</i> <i>Enable/disable</i> tt:NetworkProtocol NetworkProtocols [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.2.12 Définition des protocoles réseau

Cette opération permet de configurer des protocoles réseau définis d'un dispositif. Le dispositif doit prendre en charge la configuration de protocoles réseau définis grâce à la commande SetNetworkProtocols (voir Tableau 23).

Tableau 23 – Commande SetNetworkProtocols

SetNetworkProtocols		Demande-Réponse
Nom du message	Description	
SetNetworkProtocolsRequest	<i>Ce message configure un ou plusieurs protocoles réseau définis pris en charge par le dispositif. Il existe actuellement trois protocoles définis, HTTP, HTTPS et RTSP. Les paramètres suivants peuvent être définis pour chaque protocole:</i> <i>Port</i> <i>Enable/disable</i> tt:NetworkProtocol NetworkProtocols [1][non limité]	
SetNetworkProtocols-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender	<i>Le service réseau fourni n'est pas pris en charge.</i>	
ter:InvalidArgVal		
ter:ServiceNotSupported		

8.2.13 Obtention de passerelle par défaut

Cette opération permet d'obtenir les paramètres de passerelle par défaut d'un dispositif. Le dispositif doit prendre en charge la commande GetNetworkDefaultGateway retournant la/les adresse(s) de passerelle par défaut configurée(s) (voir Tableau 24).

Tableau 24 – Commande GetNetworkDefaultGateway

GetNetworkDefaultGateway		Demande-Réponse
Nom du message	Description	
GetNetworkDefaultGateway-Request	<i>Ceci est un message vide.</i>	
GetNetworkDefaultGateway-Response	<i>Ce message contient:</i> <i>"IPv4Address": Adresse(s) de passerelle IPv4 par défaut.</i> <i>"IPv6Address": Adresse(s) de passerelle IPv6 par défaut.</i> tt:IPv4Address IPv4Address [0][non limité] tt:IPv6Address IPv6Address [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.2.14 Définition de passerelle par défaut

Cette opération permet de définir les paramètres de passerelle par défaut d'un dispositif. Le dispositif doit prendre en charge la configuration de passerelle par défaut grâce à la commande SetNetworkDefaultGateway (voir Tableau 25).

Tableau 25 – Commande SetNetworkDefaultGateway

SetNetworkDefaultGateway		Demande-Réponse
Nom du message	Description	
SetNetworkDefaultGateway-Request	<i>Ce message contient:</i> "IPv4Address": Adresse(s) de passerelle IPv4 par défaut. "IPv6Address": Adresse(s) de passerelle IPv6 par défaut. tt:IPv4Address IPv4Address [0][non limité] tt:IPv6Address IPv6Address [0][non limité]	
SetNetworkDefaultGateway-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender	<i>L'adresse de passerelle fournie n'est pas valide.</i>	
ter:InvalidArgVal		
ter:InvalidGatewayAddress		

8.2.15 Obtention de configuration zéro

Cette opération permet d'obtenir la configuration zéro d'un dispositif. Si le dispositif prend en charge la configuration IP dynamique conformément à la norme [RFC 3927], il doit prendre en charge le retour de l'adresse et du statut de configuration zéro IPv4 grâce à la commande GetZeroConfiguration (voir Tableau 26).

Tableau 26 – Commande GetZeroConfiguration

GetZeroConfiguration		Demande-Réponse
Nom du message	Description	
GetZeroConfigurationRequest	<i>Ceci est un message vide.</i>	
GetZeroConfigurationResponse	<i>Ce message contient:</i> <i>"InterfaceToken": Jeton de l'interface réseau</i> <i>"Enabled": Indique si la configuration zéro est activée ou non.</i> <i>"Addresses": Adresse(s) de configuration zéro IPv4.</i> tt:ReferenceToken InterfaceToken [1][1] xs:boolean Enabled [1][1] tt:IPv4Addresses Address [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.2.16 Définition de configuration zéro

Cette opération permet de définir la configuration zéro du dispositif. Si le dispositif prend en charge la configuration IP dynamique conformément à la norme [RFC 3927], il doit prendre en charge la configuration d'adresse et de statut de configuration zéro IPv4 grâce à la commande SetZeroConfiguration (voir Tableau 27).

Tableau 27 – Commande SetZeroConfiguration

SetZeroConfiguration		Demande-Réponse
Nom du message	Description	
SetZeroConfigurationRequest	<i>Ce message contient:</i> <i>"InterfaceToken": Jeton de l'interface réseau cible.</i> <i>"Enabled": Indique si la configuration zéro est activée ou non.</i> tt:ReferenceToken InterfaceToken [1][1] xs:boolean Enabled [1][1]	
SetZeroConfigurationResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender	<i>Le jeton d'interface réseau fourni n'existe pas.</i>	
ter:InvalidArgVal		
ter:InvalidNetworkInterface		

8.2.17 Obtention de filtre d'adresse IP

Cette opération permet d'extraire les paramètres de filtre d'adresse IP d'un dispositif. Si le dispositif prend en charge le contrôle d'accès de dispositif basé sur des règles de filtrage IP (plages d'adresses IP rejetées ou acceptées), il doit prendre en charge la commande GetIPAddressFilter (voir Tableau 28).

Tableau 28 – Commande GetIPAddressFilter

GetIPAddressFilter		Demande-Réponse
Nom du message	Description	
GetIPAddressFilterRequest	<i>Ceci est un message vide.</i>	
GetIPAddressFilterResponse	<i>Ce message contient:</i> <i>"Type": Définit s'il convient que le filtre refuse ou autorise l'accès.</i> <i>"IPv4Address": Adresse(s) de filtre IPv4</i> <i>"IPv6Address": Adresse(s) de filtre IPv6</i> tt:IPAddressFilterType Type [1][1] tt:PrefixedIPv4Address IPv4Address [0][non limité] tt:PrefixedIPv6Address IPv6Address [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.2.18 Définition de filtre d'adresse IP

Cette opération permet de définir les paramètres de filtre d'adresse IP d'un dispositif. Si le dispositif prend en charge le contrôle d'accès de dispositif basé sur des règles de filtrage IP (plages d'adresses IP rejetées ou acceptées), il doit prendre en charge la configuration de règles de filtrage IP grâce à la commande SetIPAddressFilter (voir Tableau 29).

Tableau 29 – Commande SetIPAddressFilter

SetIPAddressFilter		Demande-Réponse
Nom du message	Description	
SetIPAddressFilterRequest	<i>Ce message contient:</i> <i>"Type": Définit s'il convient que le filtre refuse ou autorise l'accès.</i> <i>"IPv4Address": Adresse(s) de filtre IPv4</i> <i>"IPv6Address": Adresse(s) de filtre IPv6</i> tt:IPAddressFilterType Type [1][1] tt:PrefixedIPv4Address IPv4Address [0][non limité] tt:PrefixedIPv6Address IPv6Address [0][non limité]	
SetIPAddressFilterResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>L'adresse IPv6 suggérée est non valide.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>L'adresse IPv4 suggérée est non valide.</i>	

8.2.19 Ajout d'une adresse de filtre IP

Cette opération permet d'ajouter une adresse de filtre IP à un dispositif. Si le dispositif prend en charge le contrôle d'accès de dispositif basé sur des règles de filtrage IP (plages d'adresses IP rejetées ou acceptées), il doit prendre en charge l'ajout d'adresses de filtrage IP grâce à la commande AddIPAddressFilter (voir Tableau 30).

Tableau 30 – Commande AddIPAddressFilter

AddIPAddressFilter		Demande-Réponse
Nom du message	Description	
AddIPAddressFilterRequest	<i>Ce message contient:</i> <i>"IPv4Address": Adresse(s) de filtre IPv4</i> <i>"IPv6Address": Adresse(s) de filtre IPv6</i> <i>tt:PrefixedIPv4Address IPv4Address [0][non limité]</i> <i>tt:PrefixedIPv6Address IPv6Address [0][non limité]</i>	
AddIPAddressFilterResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:IPFilterListIsFull	<i>Aucun filtre IP supplémentaire ne peut être ajouté, car la liste de filtres IP est pleine.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>L'adresse IPv6 suggérée est non valide.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>L'adresse IPv4 suggérée est non valide.</i>	

8.2.20 Suppression d'une adresse de filtre IP

Cette opération permet de supprimer une adresse de filtre IP d'un dispositif. Si le dispositif prend en charge le contrôle d'accès de dispositif basé sur des règles de filtrage IP (pages d'adresses IP rejetées ou acceptées), il doit prendre en charge la suppression d'adresses de filtrage IP grâce à la commande RemoveIPAddressFilter (voir Tableau 31).

Tableau 31 – Commande RemoveIPAddressFilter

RemoveIPAddressFilter		Demande-Réponse
Nom du message	Description	
RemoveIPAddressFilter-Request	<i>Ce message contient:</i> "IPv4Address": Adresse(s) de filtre IPv4 "IPv6Address": Adresse(s) de filtre IPv6 tt:PrefixedIPv4Address IPv4Address [0][non limité] tt:PrefixedIPv6Address IPv6Address [0][non limité]	
RemoveIPAddressFilter-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:InvalidIPv6Address	<i>L'adresse IPv6 suggérée est non valide.</i>	
env:Sender ter:InvalidArgVal ter:InvalidIPv4Address	<i>L'adresse IPv4 suggérée est non valide.</i>	
env:Sender ter:InvalidArgVal ter:NoIPv6Address	<i>L'adresse IPv6 à supprimer n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoIPv4Address	<i>L'adresse IPv4 à supprimer n'existe pas.</i>	

8.2.21 Configuration IEEE 802.11

Les exigences du présent paragraphe et des sous-paragraphe concernent uniquement les dispositifs avec prise en charge IEEE 802.11. Le terme "le dispositif" est utilisé ici pour indiquer un dispositif avec prise en charge IEEE 802.11.

Le dispositif doit prendre en charge la configuration IEEE 802.11 et doit, en réponse à la méthode GetNetworkInterfaces, renvoyer ieee80211 (71) comme IANA-IfTypes pour la/les interface(s) 802.11.

Un dispositif ne doit pas renvoyer d'élément de liaison dans la réponse GetNetworkInterfaces et doit ignorer les éléments Link de la demande SetNetworkInterfaces.

Il convient que le dispositif prenne en charge le fait que chaque interface réseau IEEE 802.11 puisse être associée à plusieurs autres configurations IEEE 802.11.

La configuration IEEE 802.11 est prise en charge grâce à un élément de configuration IEEE 802.11 facultatif de l'élément de configuration d'obtention et de définition de réseau. Les informations suivantes sont traitées:

- SSID;

- mode de station;
- configuration de réseau sans fil multiple;
- configuration de sécurité.

Les opérations suivantes permettent de faciliter la gestion de la configuration sans fil:

- obtention des fonctionnalités IEEE 802.11;
- obtention du statut IEEE 802.11;
- balayage de réseaux IEEE 802.11 disponibles.

8.2.21.1 SSID

Le dispositif doit prendre en charge la configuration du SSID.

8.2.21.2 Mode de station

Le dispositif doit prendre en charge le mode de station de l'infrastructure.

Le dispositif PEUT prendre en charge le mode de station de réseau ad-hoc. La configuration réelle nécessaire du mode de station de réseau ad-hoc, y compris la configuration manuelle du nombre de voies, n'entre pas dans le domaine d'application de la présente norme. Mais pour tenir compte des dispositifs qui prennent en charge les modes de station de réseau ad-hoc, la spécification permet de sélectionner (et de signaler) ce mode.

8.2.21.3 Configuration de réseau sans fil multiple

Chaque configuration IEEE 802.11 doit être identifiée par un alias (identifiant). L'alias doit être unique dans une configuration d'interface réseau. Le client doit fournir l'alias dans la demande SetNetworkInterfaces. Si le client souhaite mettre à jour une configuration sans fil existante, le même alias doit être utilisé. Une configuration sans fil, incluant l'alias, doit uniquement exister s'il s'agit d'une partie d'une configuration d'interface réseau.

Pour que le dispositif puisse hiérarchiser les priorités entre plusieurs configurations IEEE 802.11 alternatives, une valeur de priorité facultative peut être utilisée, une valeur plus élevée indiquant une priorité plus importante. Si la valeur de priorité est absente de la configuration, la priorité la plus basse doit être supposée. Si plusieurs configurations sans fil présentent la même valeur de priorité, l'ordre de ces configurations n'est pas défini.

L'algorithme utilisé par le dispositif pour activer un réseau IEEE 802.11 selon une liste hiérarchique de configurations IEEE 802.11 n'entre pas dans le domaine d'application de la présente norme.

8.2.21.4 Configuration de sécurité

La configuration de sécurité contient le mode de sécurité choisi et la configuration nécessaire pour ce mode. Les modes de sécurité suivants sont pris en charge:

- Néant
- PSK (Pre Shared Key - Clé prépartagée) (WPA- et WPA2-Personnel)
- IEEE 802.1X-2004 (WPA- et WPA2-Enterprise)

La configuration du mode de sécurité WEP n'entre pas dans le domaine d'application de la présente norme, mais pour tenir compte des dispositifs qui prennent en charge le mode de sécurité WEP, la présente norme permet de sélectionner (et de signaler) ce mode.

Pour assurer la confidentialité et l'intégrité des données, le dispositif doit, conformément à la spécification [IEEE 802.11-2007], prendre en charge l'algorithme CCMP et PEUT prendre en charge l'algorithme TKIP.

L'algorithme peut être sélectionné manuellement (CCMP, TKIP) ou automatiquement (Tous). En mode de sélection manuelle, le même algorithme doit être utilisé pour le chiffrement par paire et par groupe. Pour pouvoir prendre en charge d'autres algorithmes, une valeur "étendue" est disponible.

Le dispositif doit prendre en charge les modes de sélection manuelle et automatique.

8.2.21.4.1 Mode Néant

Le dispositif doit prendre en charge le mode de sécurité "Néant".

8.2.21.4.2 Mode PSK

Le dispositif doit prendre en charge le mode de sécurité "PSK".

Pour limiter les risques de compromettre la PSK, il convient que le dispositif ne transmette PAS de PSK à un client. De plus, il NE DOIT PAS renvoyer la PSK dans une réponse à un appel d'opération GetNetworkInterfaces.

Pour ajouter une configuration sans fil avec le mode de sécurité PSK, les règles suivantes s'appliquent:

- un client doit inclure une valeur PSK dans la demande SetNetworkInterfaces;
- le dispositif doit vérifier qu'une valeur PSK a été fournie. Si ce n'est pas le cas, il doit renvoyer une erreur.

Pour mettre à jour une configuration sans fil avec le mode de sécurité PSK, les règles suivantes s'appliquent:

- si le client souhaite conserver la valeur PSK, il convient de ne PAS l'inclure dans la demande SetNetworkInterfaces;
- le dispositif qui reçoit une demande SetNetworkInterfaces sans valeur PSK doit conserver sa valeur PSK.

La norme [IEEE 802.11-2007] stipule qu'il convient de distribuer la PSK au STA selon une méthode hors bande. Dans l'ONVIF, la règle de sécurité doit permettre de s'assurer que la PSK est suffisamment protégée.

8.2.21.4.3 Mode IEEE 802.1X-2004

Il convient que le dispositif prenne en charge le mode de sécurité IEEE 802.1X. Pour obtenir des exigences plus détaillées sur le mode de sécurité IEEE 802.1X-2004, voir [Configuration IEEE 802.1X].

8.2.21.5 Obtention des fonctionnalités Dot11

Cette opération renvoie les fonctionnalités IEEE 802.11 (voir Tableau 32 et Tableau 33). Le dispositif doit prendre en charge cette opération.

Tableau 32 – Fonctionnalités GetDot11

Fonctionnalités GetIEEE 802.11		Demande-Réponse
Nom du message	Description	
GetDot11Capabilities-Request	<i>Ceci est un message vide.</i>	
GetDot11Capabilites-Response	<i>tt:Dot11Capabilities Capabilities [1][1]</i>	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:InvalidDot11	<i>La configuration IEEE 802.11 n'est pas prise en charge.</i>	

Tableau 33 – Fonctionnalités IEEE 802.11

Fonctionnalité	Description
TKIP	Indique si le dispositif prend en charge l'algorithme TKIP.
ScanAvailableNetworks	Indique si le dispositif prend en charge le balayage de réseaux IEEE 802.11 disponibles.
MultipleConfiguration	Indique si le dispositif prend en charge plusieurs configurations IEEE 802.11 alternatives.
AdHocStationMode	Indique si le dispositif prend en charge le mode de station ad-hoc.
WEP	Indique si le dispositif prend en charge le mode de sécurité WEP.

8.2.21.6 Obtention du statut IEEE 802.11

Cette opération renvoie le statut d'une interface réseau sans fil. Le dispositif doit prendre en charge cette commande (voir Tableau 34). Les statuts suivants peuvent être renvoyés:

- SSID (impératif);
- BSSID (recommandé);
- chiffrement par paire (recommandé);
- chiffrement par groupe (recommandé);
- force du signal (recommandé);
- alias de la configuration sans fil active (impératif).

Tableau 34 – GetDot11Status

GetDot11Status		Demande-Réponse	
Nom du message		Description	
GetDot11StatusRequest		<i>tt:ReferenceToken InterfaceToken [1][1]</i>	
GetDot11StatusResponse		<i>tt:Dot11Status Status [1][1]</i>	
Codes de défaut		Description	
env:Receiver ter:ActionNotSupported ter:InvalidDot11		<i>La configuration IEEE 802.11 n'est pas prise en charge.</i>	
env:Sender ter:InvalidArgVal ter:NotDot11		<i>L'interface n'est pas une interface IEEE 802.11.</i>	
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface		<i>Le jeton d'interface réseau fourni n'existe pas.</i>	
env:Receiver ter:Action ter:NotConnectedDot11		<i>Le réseau IEEE 802.11 n'est pas connecté.</i>	

8.2.21.7 Balayage de réseaux IEEE 802.11 disponibles

Cette opération renvoie une liste des réseaux sans fil dans la gamme du dispositif. Il convient qu'un dispositif prenne en charge cette opération (voir Tableau 35). Les statuts suivants peuvent être renvoyés pour chaque réseau:

- SSID (impératif);
- BSSID (recommandé);
- authentification et suite(s) de gestion de clé (recommandé);
- chiffrement(s) par paire (recommandé);
- chiffrement(s) par groupe (recommandé);
- force du signal (recommandé).

Tableau 35 – ScanAvailable802.11Networks

ScanAvailable802.11Networks		Demande-Réponse
Nom du message	Description	
ScanAvailableDot11-NetworksRequest	<i>tt:ReferenceToken InterfaceToken [1][1]</i>	
ScanAvailableDot11-NetworksResponse	<i>tt:Dot11AvailableNetworks Networks [0][non limité]</i>	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:InvalidDot11	<i>La configuration IEEE 802.11 n'est pas prise en charge.</i>	
env:Sender ter:InvalidArgVal ter:NotDot11	<i>L'interface n'est pas une interface IEEE 802.11.</i>	
env:Sender ter:InvalidArgVal ter:InvalidNetworkInterface	<i>Le jeton d'interface réseau fourni n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:NotScanAvailable	<i>ScanAvailableDot11Networks n'est pas pris en charge.</i>	

8.3 Système

8.3.1 Informations de dispositif

Cette opération permet d'extraire des informations de dispositif (le fabricant, le modèle et la version de micrologiciel d'un dispositif, par exemple). Le dispositif doit prendre en charge le retour d'informations de dispositif grâce à la commande GetDeviceInformation (voir Tableau 36).

Tableau 36 – Commande GetDeviceInformation

GetDeviceInformation		Demande-Réponse
Nom du message	Description	
GetDeviceInformationRequest	<i>Ceci est un message vide.</i>	
GetDeviceInformationResponse	<i>Le message de réponse d'obtention d'informations de dispositif retourne les informations de dispositif suivantes:</i> xs:string Manufacturer [1][1] xs:string Model [1][1] xs:string FirmwareVersion [1][1] xs:string SerialNumber [1][1] xs:string HardwareId [1][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.2 Obtention des URI du système

Cette opération permet d'extraire les URI à partir desquels les informations système peuvent être téléchargées à l'aide de HTTP. Les URI peuvent être renvoyés pour les informations système suivantes:

- Journaux système. Plusieurs journaux système de types différents peuvent être renvoyés. Le format exact des journaux système est hors du domaine d'application de la présente norme;
- Informations de prise en charge. Il s'agit d'informations de diagnostic de dispositif aléatoires provenant d'un dispositif. Le format exact des informations de diagnostic est hors du domaine d'application de la présente norme;
- Sauvegarde du système. Le fichier reçu est un fichier de sauvegarde qui peut être utilisé pour restaurer ultérieurement la configuration en cours du dispositif. Le format exact du fichier de configuration de sauvegarde est hors du domaine d'application de la présente norme.

Si le dispositif permet d'extraire les fichiers journaux, les informations de prise en charge ou les données de sauvegarde du système, il convient de les mettre à disposition via HTTP GET. Si c'est le cas, il doit prendre en charge la commande GetSystemUris (voir Tableau 37).

Tableau 37 – Commande GetSystemUri

GetSystemUri		Demande-Réponse
Nom du message	Description	
GetSystemUriRequest	<i>Ceci est un message vide.</i>	
GetSystemUriResponse	<i>Ce message contient les URI à partir desquels les différents composants d'information du système peuvent être téléchargés.</i> tt:SystemLogUriList SystemLogUri [0][1] xs:anyURI SupportInfoUri [0][1] xs:anyURI SystemBackupUri [0][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.3 Sauvegarde

Cette opération permet d'extraire le(s) fichier(s) de configuration de sauvegarde système d'un dispositif. Il convient que le dispositif prenne en charge le retour de fichier(s) de configuration de sauvegarde grâce à la commande GetSystemBackup (voir Tableau 38). La sauvegarde est retournée avec une référence à un nom et un type MIME conjointement avec des données binaires. Le format exact des fichiers de configuration de sauvegarde est *hors du domaine d'application* de la présente Norme.

Le(s) fichier(s) de configuration de sauvegarde est (sont) transmis via MTOM [MTOM].

Tableau 38 – Commande GetSystemBackup

GetSystemBackup		Demande-Réponse
Nom du message	Description	
GetSystemBackupRequest	<i>Ceci est un message vide.</i>	
GetSystemBackupResponse	<i>Le message de réponse d'obtention de sauvegarde système contient le(s) fichier(s) de configuration de sauvegarde système.</i> tt:BackupFile BackupFiles [1][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.4 Restauration

Cette opération permet de restaurer le(s) fichier(s) de configuration de sauvegarde système qui ont été extraits d'un dispositif. Il convient que le dispositif prenne en charge la restauration du/des fichier(s) de configuration de sauvegarde grâce à la commande

RestoreSystem (voir Tableau 39). Le format exact du/des fichier(s) de configuration de sauvegarde est *hors du domaine d'application* de la présente Norme. Si la commande est prise en charge, elle doit accepter les fichiers de sauvegarde retournés par la commande GetSystemBackup.

Le(s) fichier(s) de configuration de sauvegarde est (sont) transmis via MTOM [MTOM].

Tableau 39 – Commande RestoreSystem

RestoreSystem		Demande-Réponse
Nom du message	Description	
RestoreSystemRequest	Ce message contient le(s) fichier(s) de sauvegarde système. tt:BackupFile BackupFiles [1][non limité]	
RestoreSystemResponse	Ceci est un message vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:InvalidBackupFile	Le(s) fichier(s) de sauvegarde n'est (ne sont) pas valide(s).	

8.3.5 Démarrage de la restauration du système

Cette opération permet de lancer une restauration du système à partir des données de configuration sauvegardées à l'aide du mécanisme HTTP POST. La réponse à la commande comprend une URL HTTP vers laquelle le fichier de sauvegarde peut être chargé. La restauration réelle a lieu à l'issue de l'opération HTTP POST. Il convient que les dispositifs prennent en charge la restauration du système grâce à la commande StartSystemRestore (voir Tableau 40). Le format exact des données de configuration de sauvegarde est hors du domaine d'application de la présente norme.

La restauration du système sur HTTP peut être assurée selon la procédure suivante:

- 1) le client appelle StartSystemRestore;
- 2) le serveur répond avec l'URI chargé;
- 3) le client transmet les données de configuration à l'URI de chargement à l'aide de HTTP POST;
- 4) le serveur applique la configuration chargée, puis procède au redémarrage, le cas échéant.

Si la restauration du système n'aboutit pas, car le fichier chargé n'est pas valide, la réponse HTTP POST doit être "415 Unsupported Media Type" (Type de média non pris en charge). Si la restauration du système n'aboutit pas suite à une erreur dans le dispositif, la réponse HTTP POST doit être "500 Internal Server Error" (Erreur du serveur interne).

La valeur de l'en-tête Content-Type de la demande HTTP POST doit être "application/octet-stream".

Tableau 40 – Commande StartSystemRestore

StartSystemRestore		Demande-Réponse
Nom du message	Description	
StartSystemRestoreRequest	<i>Ceci est un message vide.</i>	
StartSystemRestoreResponse	<i>Ce message contient:</i> <i>une URL vers laquelle le fichier de configuration du système peut être chargé.</i> <i>une durée facultative indiquant combien de temps il est prévu que le dispositif soit indisponible à l'issue du chargement.</i> xs:anyURI UploadUri [1][1] xs:duration ExpectedDowntime [0][1]	
Codes de défaut	Description	
	<i>Pas de défaut spécifique à la commande.</i>	

8.3.6 Obtention des date et heure système

Cette opération extrait la date et l'heure système du dispositif. Le dispositif doit prendre en charge le retour du réglage d'heure d'été et des date et heure système manuelles (le cas échéant) ou une indication de l'heure NTP (le cas échéant) grâce à la commande GetSystemDateAndTime (voir Tableau 41).

Un dispositif doit fournir les informations UTCDateTime, même si l'élément est marqué comme étant facultatif afin d'assurer la compatibilité en amont.

Tableau 41 – Commande GetSystemDateAndTime

GetSystemDateAndTime		Demande-Réponse
Nom du message	Description	
GetSystemDateAndTime-Request	<i>Ceci est un message vide.</i>	
GetSystemDateAndTime-Response	<i>Ce message contient les informations de date et d'heure du dispositif.</i> <i>"DateTimeType": Indique si la date et l'heure système sont définies manuellement ou par NTP</i> <i>"DaylightSavings": Heure d'été activée ou désactivée</i> <i>"TimeZone": Fuseau horaire tel qu'il est défini dans POSIX 1003.1, 8.3</i> <i>"UTCDateTime": date et heure en TUC</i> <i>"LocalDateTime": Date et heure locales du dispositif</i> tt:SetDateTimeType DateTimeType [1][1] xs:boolean DayLightSavings [1][1] tt:TimeZone TimeZone [0][1] tt:DateTime UTCDateTime [0][1] tt:DateTime LocalDateTime [0][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.7 Définition des date et heure système

Cette opération permet de définir les date et heure système du dispositif. Le dispositif doit prendre en charge la configuration du réglage d'heure d'été et des date et heure système manuelles (le cas échéant) ou une indication de l'heure NTP (le cas échéant) grâce à la commande SetSystemDateAndTime (voir Tableau 42).

Si les date et heure système sont définies manuellement, le client doit inclure UTCDateTime ou LocalDateTime dans la demande.

Tableau 42 – Commande SetSystemDateAndTime

SetSystemDateAndTime		Demande-Réponse
Nom du message	Description	
SetSystemDateAndTime-Request	<i>Ce message contient les informations de date et d'heure du dispositif.</i> <i>"DateTimeType": Indique si la date et l'heure système sont définies manuellement ou par NTP</i> <i>"DaylightSavings": Heure d'été activée ou désactivée</i> <i>"TimeZone": Le fuseau horaire est défini dans POSIX 1003.1, 8.3</i> <i>"UTCDateTime": date et heure en TUC. Si DateTimeType est NTP, UTCDateTime n'a aucune signification.</i> tt:SetDateTimeType DateTimeType [1][1] xs:boolean DayLightSavings [1][1] tt:TimeZone TimeZone [0][1] tt:DateTime UTCDateTime [0][1]	
SetSystemDateAndTime-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:InvalidTimeZone	<i>Un fuseau horaire non valide a été spécifié.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDateTime	<i>Une date ou une heure non valide a été spécifiée.</i>	

8.3.8 Réglages par défaut d'usine

Cette opération permet de réinitialiser les paramètres d'un dispositif à leurs valeurs par défaut d'usine. Le dispositif doit prendre en charge le réglage par défaut d'usine total et partiel grâce à commande SetSystemFactoryDefault (voir Tableau 43). La définition d'un *réglage par défaut d'usine partiel* est spécifique au produit et spécifique au fournisseur de dispositif. L'effet d'une opération de *réglage par défaut d'usine partiel* n'est pas totalement défini. Cependant, il doit être garanti qu'après une réinitialisation logicielle, le dispositif soit accessible à la même adresse IP que celle utilisée avant la réinitialisation. Cela signifie que les paramètres de réseau de base (les paramètres d'adresse IP, de sous-réseau et de passerelle ou DHCP, par exemple) restent inchangés après la réinitialisation partielle.

Tableau 43 – Commande SetSystemFactoryDefault

SetSystemFactoryDefault		Demande-Réponse
Nom du message	Description	
SetSystemFactoryDefault-Request	<i>Ce message contient les types de réglages par défaut d'usine à effectuer.</i> <i>"Hard": Tous les paramètres sont définis à leurs valeurs par défaut d'usine</i> <i>"Soft": Tous paramètres, à l'exception des paramètres spécifiques au fournisseur de dispositif, sont définis à leurs valeurs par défaut d'usine</i> tt:FactoryDefaultType FactoryDefault [1][1]	
SetSystemFactoryDefault-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.9 Mise à niveau de micrologiciel

Cette opération permet de mettre à niveau la version de micrologiciel d'un dispositif. Après une mise à niveau réussie, le message de réponse est envoyé avant le redémarrage du dispositif. Il convient que le dispositif prenne en charge la mise à niveau de micrologiciel grâce à la commande UpgradeSystemFirmware (voir Tableau 44). Le format exact des données de micrologiciel est *hors du domaine d'application* de la présente Norme.

Le micrologiciel est transmis via MTOM [MTOM].

Tableau 44 – Commande UpgradeSystemFirmware

UpgradeSystemFirmware		Demande-Réponse
Nom du message	Description	
UpgradeSystemFirmware-Request	Ce message contient le micrologiciel utilisé pour la mise à niveau. La mise à niveau de micrologiciel est "partielle", ce qui signifie que tous les paramètres conservent leur valeur actuelle. tt:AttachmentData Firmware [1][1]	
UpgradeSystemFirmware-Response	Ce message contient une chaîne "Message" permettant au dispositif de renvoyer un message au client ("Mise à niveau réussie, redémarrage dans x secondes.", par exemple) xs:string Message [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:InvalidFirmware	<i>Le micrologiciel est non valide, c'est-à-dire qu'il n'est pas pris en charge par ce dispositif.</i>	
env:Receiver ter:Action ter:FirmwareUpgrade-Refusé	<i>La mise à niveau de micrologiciel a échoué.</i>	

8.3.10 Démarrage de la mise à niveau du micrologiciel

Cette opération permet de lancer une mise à niveau de micrologiciel à l'aide du mécanisme HTTP POST. La réponse à la commande comprend une URL HTTP vers laquelle le fichier de mise à niveau peut être chargé. La mise à niveau réelle a lieu à l'issue de l'opération HTTP POST. Il convient que le dispositif prenne en charge la mise à niveau de micrologiciel grâce à la commande StartFirmwareUpgrade (voir Tableau 45). Le format exact des données de micrologiciel est hors du domaine d'application de la présente norme.

La mise à niveau de micrologiciel sur HTTP peut être assurée selon la procédure suivante:

- 1) le client appelle StartFirmwareUpgrade;
- 2) le serveur répond avec l'URI de chargement et une valeur de délai facultative;
- 3) le client attend pendant le délai indiqué, s'il est spécifié par le serveur;
- 4) le client transmet l'image de micrologiciel à l'URI de chargement à l'aide de HTTP POST;
- 5) le serveur se reprogramme à l'aide de l'image chargée, puis redémarre.

Si la mise à niveau de micrologiciel n'aboutit pas, car le fichier de mise à niveau n'est pas valide, la réponse HTTP POST doit être "415 Unsupported Media Type" (Type de média non pris en charge). Si la mise à niveau de micrologiciel n'aboutit pas suite à une erreur dans le dispositif, la réponse HTTP POST doit être "500 Internal Server Error" (Erreur du serveur interne).

La valeur de l'en-tête Content-Type de la demande HTTP POST doit être "application/octet-stream".

Tableau 45 – Commande StartFirmwareUpgrade

StartFirmwareUpgrade		Demande-Réponse
Nom du message	Description	
StartFirmwareUpgrade-Request	<i>Ceci est un message vide.</i>	
StartFirmwareUpgrade-Response	<i>Ce message contient:</i> <i>une URL vers laquelle le fichier de micrologiciel du système peut être chargé.</i> <i>Un délai facultatif; le client doit attendre pendant la durée indiquée avant de procéder au chargement du micrologiciel.</i> <i>Une durée indiquant combien de temps il est prévu que le dispositif soit indisponible à l'issue du chargement du micrologiciel.</i> xs:anyURI UploadUri [1][1] xs:duration UploadDelay [0][1] xs:duration ExpectedDowntime [0][1]	
Codes de défaut	Description	
	<i>Ras de défaut spécifique à la commande.</i>	

8.3.11 Obtention des journaux système

Cette opération permet d'extraire un journal système d'un dispositif. Il convient que le dispositif prenne en charge l'extraction d'informations de journal système grâce à la commande GetSystemLog (voir Tableau 46). Le format exact des journaux système est *hors du domaine d'application* de la présente Norme.

Les informations de journal système sont transmises via MTOM [MTOM] ou sous la forme d'une chaîne.

Tableau 46 – Commande GetSystemLog

GetSystemLog		Demande-Réponse
Nom du message	Description	
GetSystemLogRequest	<i>Ce message contient le type de journal système à extraire. Les informations de journal prises en charge sont définies dans deux types différents:</i> <i>"System": Journal système</i> <i>"Access": Journal d'accès client</i> tt:SystemLogType LogType [1][1]	
GetSystemLogResponse	<i>Ce message contient les informations de journal système demandées. Le dispositif peut choisir de retourner les informations de journal système sous la forme de données binaires dans une pièce jointe ou sous la forme d'une chaîne commune.</i> tt:AttachmentData Binary [0][1] xs:string String [0][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:AccesslogUnavailable	<i>Aucune information de journal d'accès disponible.</i>	
env:Sender ter:InvalidArgs ter:SystemlogUnavailable	<i>Aucune information de journal système disponible.</i>	

8.3.12 Obtention d'informations d'assistance

L'opération permet d'extraire des informations de diagnostic de dispositif aléatoires provenant d'un dispositif. Le dispositif PEUT prendre en charge l'extraction d'informations de diagnostic grâce à la commande GetSystemSupportInformation (voir Tableau 47). Le format exact des informations de diagnostic est *hors du domaine d'application* de la présente Norme.

Les informations de diagnostic sont transmises sous la forme d'une pièce jointe via MTOM [MTOM] ou sous forme de chaîne.

Tableau 47 – Commande GetSystemSupportInformation

GetSystemSupportInformation		Demande-Réponse
Nom du message	Description	
GetSystemSupport-InformationRequest	<i>Ceci est un message vide.</i>	
GetSystemSupport-Information Response	<i>Le message contient les informations de support. Le dispositif peut choisir de retourner les informations de support sous la forme de données binaires ou d'une chaîne commune.</i> tt:AttachmentData BinaryFormat [0][1] xs:string StringFormat [0][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:SupportInformation-Unavailable	<i>Aucune information de prise en charge n'est disponible.</i>	

8.3.13 Redémarrage

Cette opération permet de redémarrer un dispositif. Le message de réponse doit être envoyé avant le redémarrage du dispositif. Le dispositif doit prendre en charge le redémarrage grâce à la commande SystemReboot (voir Tableau 48).

Tableau 48 – Commande SystemReboot

SystemReboot		Demande-Réponse
Nom du message	Description	
SystemReboot	<i>Ceci est un message vide.</i>	
SystemRebootResponse	Ce message contient une chaîne "Message" permettant au dispositif de renvoyer un message au client ("Redémarrage dans x secondes", par exemple). xs:string Message [1][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.14 Obtention de paramètres de domaine d'application

Cette opération permet de *demande* les paramètres de domaine d'application d'un dispositif. Les paramètres de domaine d'application sont utilisés dans la découverte de dispositif pour mettre en correspondance un message de sonde (voir Article 7). Les paramètres de domaine d'application sont de deux types différents:

- fixed;
- configurable.

Les paramètres de domaine d'application fixes *ne peuvent pas* être modifiés par l'intermédiaire de l'interface de gestion de dispositif, mais ce sont des caractéristiques de dispositif permanentes faisant partie des configurations de micrologiciel. Le type de domaine d'application est indiqué dans la liste de domaines d'application retournée dans la réponse de demande de paramètres de domaine d'application. Les paramètres de domaine d'application configurables peuvent être définis par l'intermédiaire des opérations de définition et d'ajout de paramètres de domaine d'application (voir 0 et 0). Le dispositif doit prendre en charge l'extraction de paramètres de domaine d'application de découverte grâce à la commande GetScopes (voir Tableau 49). Étant donné que certains paramètres de domaine d'application sont obligatoires, le client attend toujours une liste de domaines d'application dans la réponse.

Tableau 49 – Commande GetScopes

GetScopes		Demande-Réponse
Nom du message	Description	
GetScopesRequest	Ceci est un message vide.	
GetScopesResponse	Le message de réponse de domaine d'application contient une liste d'URI définissant les domaines d'application du dispositif. Voir également l'Article 7 pour les définitions de domaine d'application de l'ONVIF. tt:Scope: Scopes [1][non limité]	
Codes de défaut	Description	
env:Receiver	La liste de domaines d'application est vide.	
ter:Action		
ter:EmptyScope		

8.3.15 Définition des paramètres de domaine d'application

Cette opération permet de *définir* les paramètres de domaine d'application d'un dispositif. Les paramètres de domaine d'application sont utilisés dans la découverte de dispositif pour mettre en correspondance un message de sonde (voir Article 7).

Cette opération permet de *remplacer* tous les paramètres de domaine d'application configurables existants (paramètres non fixes). Si cela doit être évité, il convient d'utiliser plutôt la commande d'ajout de domaine d'application. Le dispositif doit prendre en charge la configuration de paramètres de domaine d'application de découverte grâce à la commande SetScopes (voir Tableau 50).

Tableau 50 – Commande SetScopes

SetScopes		Demande-Réponse
Nom du message	Description	
SetScopesRequest	<i>Le domaine d'application de définition contient une liste d'URI définissant le domaine d'application du dispositif. Voir également l'Article 7.</i> xs:anyURI: Scopes [1][non limité]	
SetScopesResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:OperationProhibited ter:ScopeOverwrite	<i>Le paramètre de domaine d'application supprime les paramètres de domaine d'application de dispositif fixe. La commande est rejetée.</i>	
env:Receiver ter:Action ter:TooManyScopes	<i>La liste de domaines d'application demandée dépasse le nombre de domaines d'application pris en charge.</i>	

8.3.16 Ajout de paramètres de domaine d'application

Cette opération permet d'ajouter de nouveaux paramètres de domaine d'application configurables à un dispositif. Les paramètres de domaine d'application sont utilisés dans la découverte de dispositif pour mettre en correspondance un message de sonde (voir Article 7). Le dispositif doit prendre en charge l'ajout de paramètres de domaine d'application de découverte grâce à la commande AddScopes (voir Tableau 51).

Tableau 51 – Commande AddScopes

AddScopes		Demande-Réponse
Nom du message	Description	
AddScopesRequest	<i>Le message d'ajout de domaine d'application contient une liste d'URI à ajouter à la liste de domaines d'application configurables existants. Voir également l'Article 7.</i> xs:anyURI:ScopeItem [1][non limité]	
AddScopesResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Receiver ter:Action ter:TooManyScopes	<i>La liste de domaines d'application demandée dépasse le nombre de domaines d'application pris en charge.</i>	

8.3.17 Suppression de paramètres de domaine d'application

Cette opération permet de *supprimer* des paramètres de domaine d'application configurables d'un dispositif. Les paramètres de domaine d'application sont utilisés dans la découverte de dispositif pour mettre en correspondance un message de sonde (voir Article 7). Le dispositif doit prendre en charge la suppression de paramètres de domaine d'application de découverte grâce à la commande RemoveScopes (voir Tableau 52).

Tableau 52 – Commande RemoveScopes

RemoveScopes		Demande-Réponse
Nom du message	Description	
RemoveScopesRequest	Le message de suppression de domaine d'application contient une liste d'URI qu'il convient de supprimer du domaine d'application du dispositif. xs:anyURI: ScopeItem [1][non limité]	
RemoveScopesResponse	Le message de réponse de domaine d'application contient une liste d'URI qui ont été supprimées du domaine d'application du dispositif. xs:anyURI: ScopeItem [0][non limité]	
Codes de défaut		Description
env:Sender	ter:OperationProhibited	Tentative de suppression de paramètre de domaine d'application fixe; commande rejetée.
	ter:FixedScope	
env:Sender	ter:InvalidArgVal	Tentative de suppression d'un domaine d'application qui n'existe pas.
	ter:NoScope	

8.3.18 Obtention de mode de découverte

Cette opération permet d'extraire le mode de découverte d'un dispositif. Voir 7.2 pour la définition des différents modes de découverte de dispositif. Le dispositif doit prendre en charge l'extraction des paramètres de mode de découverte grâce à la commande GetDiscoveryMode (voir Tableau 53).

Tableau 53 – Commande GetDiscoveryMode

GetDiscoveryMode		Demande-Réponse
Nom du message	Description	
GetDiscoveryModeRequest	<i>Ceci est un message vide.</i>	
GetDiscoveryModeResponse	<i>Ce message contient les paramètres de mode de découverte actuel, c'est-à-dire découvrable ou non découvrable.</i> tt:DiscoveryMode: DiscoveryMode [1][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.19 Définition du mode de découverte

Cette opération permet de définir le mode de découverte d'un dispositif. Voir 7.2 pour la définition des différents modes de découverte de dispositif. Le dispositif doit prendre en charge la configuration des paramètres de mode de découverte grâce à la commande SetDiscoveryMode (voir Tableau 54).

Tableau 54 – Commande SetDiscoveryMode

SetDiscoveryMode		Demande-Réponse
Nom du message	Description	
SetDiscoveryModeRequest	<i>Ce message contient le réglage de mode de découverte demandé, c'est-à-dire, découvrable ou non découvrable.</i> tt:DiscoveryMode: DiscoveryMode [1][1]	
SetDiscoveryModeResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.20 Obtention du mode de découverte à distance

Cette opération permet d'extraire le mode de découverte à distance d'un dispositif. Voir 7.4 pour la définition des extensions de découverte à distance. Un dispositif qui prend en charge la découverte à distance doit prendre en charge l'extraction des paramètres de mode de découverte à distance grâce à la commande GetRemoteDiscoveryMode (voir Tableau 55).

Tableau 55 – Commande GetRemoteDiscoveryMode

GetRemoteDiscoveryMode		Demande-Réponse
Nom du message	Description	
GetRemoteDiscoveryMode-Request	<i>Ceci est un message vide.</i>	
GetRemoteDiscoveryMode-Response	<i>Ce message contient les paramètres réels de mode de découverte à distance, c'est-à-dire découvrable ou non découvrable.</i> tt:DiscoveryMode: RemoteDiscoveryMode [1][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.21 Définition de mode de découverte à distance

Cette opération permet de définir le mode de fonctionnement de découverte à distance d'un dispositif. Voir 7.4 pour la définition des extensions distantes de découverte à distance. Un dispositif qui prend en charge la découverte à distance doit prendre en charge la configuration des paramètres de mode de découverte grâce à la commande SetRemoteDiscoveryMode (voir Tableau 56).

Tableau 56 – Commande SetRemoteDiscoveryMode

SetRemoteDiscoveryMode		Demande-Réponse
Nom du message	Description	
SetRemoteDiscoveryMode-Request	<i>Ce message contient le réglage de mode de découverte à distance demandé, c'est-à-dire découvrable ou non découvrable.</i> tt:DiscoveryMode: RemoteDiscoveryMode [1][1]	
SetRemoteDiscoveryMode-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.22 Obtention d'adresses DP distantes

Cette opération permet d'extraire l'adresse ou les adresses DP distantes d'un dispositif. Si le dispositif prend en charge la découverte à distance, comme spécifié en 7.4, il doit prendre en charge l'extraction de la/des adresse(s) DP distante(s) grâce à la commande GetDPAddresses (voir Tableau 57).

Tableau 57 – Commande GetDPAddresses

GetDPAddresses		Demande-Réponse
Nom du message	Description	
GetDPAddressesRequest	<i>Ceci est un message vide.</i>	
GetDPAddressesResponse	<i>Ce message contient l'adresse ou les adresses DP distantes configurées du dispositif. Si aucune adresse DP distante n'est configurée, une liste vide est retournée.</i> tt:NetworkHost: DPAAddress [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.3.23 Définition d'adresses DP distantes

Cette opération permet de définir l'adresse ou les adresses DP distantes d'un dispositif. Si le dispositif prend en charge la découverte à distance, comme spécifié en 7.4, il doit prendre en charge la configuration de la/des adresse(s) DP distante(s) grâce à la commande SetDPAddresses (voir Tableau 58).

Tableau 58 – Commande SetDPAddresses

SetDPAddresses		Demande-Réponse
Nom du message	Description	
SetDPAddressesRequest	<i>Ce message contient l'adresse ou les adresses DP distantes configurées du dispositif.</i> tt:NetworkHost: DPAAddress [0][non limité]	
SetDPAddressesResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.4 Sécurité

Le présent paragraphe contient un ensemble d'opérations de gestion de sécurité. Ces opérations sont sensibles aux attaques sur le réseau et doivent être protégées par un niveau d'autorisation approprié afin de ne pas mettre le dispositif en danger.

8.4.1 Obtention de politique d'accès

Il convient que l'accès aux différents services et sous-ensembles de services soit soumis à un contrôle d'accès. Le cadre WS-Security définit les conditions préalables pour l'authentification de point terminal. Les décisions d'autorisation peuvent alors être prises en utilisant une *politique de sécurité d'accès*. La présente Norme ne spécifie aucun format de description de politique particulier ni politique de sécurité particulière. C'est au fabricant de dispositif ou au

fournisseur de système de choisir une politique et un format de description de politique. Cependant, une politique d'accès (dans un format aléatoire) peut être demandée à l'aide de cette commande (voir Tableau 59). Si le dispositif prend en charge les paramètres de politique d'accès sur la base de l'authentification WS-Security, le dispositif doit prendre en charge cette commande.

Tableau 59 – Commande GetAccessPolicy

GetAccessPolicy		Demande-Réponse
Nom du message	Description	
GetAccessPolicyRequest	<i>Ceci est un message vide.</i>	
GetAccessPolicyResponse	Ce message contient le fichier de politique demandé. tt:BinaryData PolicyFile [1][1]	
Codes de défaut	Description	
env:Receiver ter:Action ter:EmptyPolicy	<i>Le fichier de politique du dispositif n'existe pas ou est vide.</i>	

8.4.2 Définition de politique d'accès

Cette commande permet de définir la politique de sécurité d'accès du dispositif (pour plus de détails sur la politique de sécurité d'accès, voir la commande Get, 8.4.1). Si le dispositif prend en charge les paramètres de politique d'accès sur la base de l'authentification WS-Security, le dispositif doit prendre en charge cette commande (voir Tableau 60).

Tableau 60 – Commande SetAccessPolicy

SetAccessPolicy		Demande-Réponse
Nom du message	Description	
SetAccessPolicyRequest	Ce message contient le fichier de politique à définir. tt:BinaryData PolicyFile [1][1]	
SetAccessPolicyResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:PolicyFormat	<i>La politique demandée ne peut pas être définie en raison d'un format de politique inconnu.</i>	

8.4.3 Obtention d'utilisateurs

Cette opération permet d'extraire les utilisateurs enregistrés et les identifiants correspondants sur un dispositif. Le dispositif doit prendre en charge l'extraction des utilisateurs de dispositif enregistrés et leurs identifiants pour le jeton d'utilisateur grâce à la commande GetUsers (voir Tableau 61).

Tableau 61 – Commande GetUsers

GetUsers		Demande-Réponse
Nom du message	Description	
GetUsersRequest	<i>Ceci est un message vide.</i>	
GetUsersResponse	<i>Ce message contient la liste des utilisateurs et les identifiants correspondants. Chaque entrée comprend:</i> <i>le nom d'utilisateur</i> <i>le niveau d'utilisateur,</i> <i>par conséquent, le mot de passe de nom d'utilisateur n'est pas inclus dans la réponse.</i> tt:User: User [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.4.4 Création d'utilisateurs

Cette opération permet de créer des utilisateurs de dispositif et les identifiants correspondants sur un dispositif pour le profil de jeton d'utilisateur (voir 5.12 pour les définitions de jeton d'utilisateur). Le dispositif doit prendre en charge la création d'utilisateurs de dispositif et leurs identifiants pour le jeton d'utilisateur grâce à la commande CreateUsers (voir Tableau 62). Soit la création de tous les utilisateurs aboutit, soit un message de défaut doit être retourné sans créer d'utilisateur.

Il est recommandé que les dispositifs satisfaisant à l'ONVIF prennent en charge une longueur de mot de passe d'au moins 28 octets, étant donné qu'un client peut suivre le mécanisme de déduction de mot de passe donnant lieu à un " mot de passe équivalent " d'une longueur de 28 octets (voir 3.1.2 de [ONVIF Security]).

Tableau 62 – Commande CreateUsers

CreateUsers		Demande-Réponse
Nom du message	Description	
CreateUsersRequest	<i>Ce message contient un élément de paramètres d'utilisateur pour un nouvel utilisateur. Chaque entrée d'utilisateur comprend:</i> <i>le nom d'utilisateur</i> <i>le mot de passe</i> <i>le niveau d'utilisateur</i> tt:User: User [1][non limité]	
CreateUsersResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:OperationProhibited ter:UsernameClash	<i>Le nom d'utilisateur existe déjà.</i>	
env:Sender ter:OperationProhibited ter:PasswordTooLong	<i>Le mot de passe est trop long</i>	
env:Sender ter:OperationProhibited ter:UsernameTooLong	<i>Le nom d'utilisateur est trop long</i>	
env:Sender ter:OperationProhibited ter:Password	<i>Mot de passe trop faible</i>	
env:Receiver ter:Action ter:TooManyUsers	<i>Nombre maximal d'utilisateurs pris en charge dépassé.</i>	
env:Sender ter:OperationProhibited ter:AnonymousNotAllowed	<i>Le niveau d'utilisateur anonyme n'est pas autorisé.</i>	
env:Sender ter:OperationProhibited ter:UsernameTooShort	<i>Le nom d'utilisateur est trop court</i>	

8.4.5 Suppression d'utilisateurs

Cette opération supprime des utilisateurs sur un dispositif pour le profil de jeton d'utilisateur (voir 5.12 pour les définitions de jeton d'utilisateur). Le dispositif doit prendre en charge la suppression d'utilisateurs de dispositif et leurs identifiants pour le jeton d'utilisateur grâce à la

commande DeleteUsers (voir Tableau 63). Un dispositif peut avoir un ou plusieurs utilisateurs fixes qui ne peuvent pas être supprimés pour assurer l'accès à l'unité. Soit la suppression de tous les utilisateurs aboutit, soit un message de défaut doit être retourné et aucun utilisateur n'est supprimé.

Tableau 63 – Commande DeleteUsers

DeleteUsers		Demande-Réponse
Nom du message	Description	
DeleteUsersRequest	<i>Ce message contient le nom de l'utilisateur ou des utilisateurs à supprimer.</i> xs:string: Username [1][non limité]	
DeleteUsersResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:UsernameMissing	<i>Nom d'utilisateur NON reconnu</i>	
env:Sender ter:InvalidArgVal ter:FixedUser	<i>Le nom d'utilisateur ne peut pas être supprimé</i>	

8.4.6 Définition de paramètres d'utilisateur

Cette opération permet de mettre à jour les paramètres d'un ou de plusieurs utilisateurs sur un dispositif pour le profil de jeton d'utilisateur. Le dispositif doit prendre en charge la mise à jour d'utilisateurs de dispositif et leurs identifiants pour le jeton d'utilisateur grâce à la commande SetUser (voir Tableau 64). Soit le traitement de toutes les demandes de modification aboutit, soit un message de défaut doit être retourné et aucune demande de modification n'est traitée.

Si la valeur de mot de passe facultative est ignorée, le mot de passe de l'utilisateur est effacé.

Tableau 64 – Commande SetUser

SetUser		Demande-Réponse
Nom du message	Description	
SetUserRequest	<i>Ce message contient une liste des utilisateurs et des paramètres correspondants à mettre à jour.</i> <i>le nom d'utilisateur</i> <i>le mot de passe</i> <i>le niveau d'utilisateur</i> tt>User: User [1][non limité]	
SetUserResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:UsernameMissing	<i>Nom d'utilisateur NON reconnu</i>	
env:Sender ter:OperationProhibited ter>PasswordTooLong	<i>Le mot de passe est trop long</i>	
env:Sender ter:OperationProhibited ter>PasswordTooWeak	<i>Mot de passe trop faible</i>	
env:Sender ter:OperationProhibited ter:AnonymousNotAllowed	<i>Le niveau d'utilisateur anonyme n'est pas autorisé.</i>	

8.4.7 Configuration IEEE 802.1X

La présente norme définit les paramètres ci-dessous sous la forme d'un ensemble de paramètres de configuration IEEE 802.1X.

- Jeton de configuration

Ce paramètre indique un jeton de référence des paramètres de configuration IEEE 802.1X. Il est défini sous la forme 'Dot1XConfigurationToken' dans [ONVIF Schema]. Cette convention de dénomination de "Dot1X", qui représente en réalité "IEEE 802.1X", est utilisée pour assurer une meilleure lisibilité de l'élément de schéma dans le code source généré.

- Identité EAP

Ce paramètre indique le nom d'utilisateur du demandeur qui se connecte au réseau géré IEEE 802.1X. Il est défini sous la forme "Identity" dans [ONVIF Schema].

- Méthode EAP

Ce paramètre indique la méthode d'authentification utilisée. Il est défini sous la forme "EAPMethod" dans [ONVIF Schema].

- ID de certificat CA

Ce paramètre indique l'ID du certificat CA utilisé pour la vérification du serveur d'authentification. Il est défini sous la forme "CACertificateID" dans [ONVIF Schema].

- Paramètres de configuration respectifs pour la méthode EAP sélectionnée

Selon la méthode EAP sélectionnée, certains paramètres spécifiques sont nécessaires:

- **[EAP-MD5], [EAP-PEAP/MSCHAP-V2], [types EAP-TTLS]:** Mot de passe d'identité permettant au serveur d'authentification de vérifier l'utilisateur (le dispositif) à l'aide du mot de passe spécifié. La méthode [EAP-MD5] ne s'applique pas à l'utilisation de 802.11 (WPA-Enterprise).
- **[EAP-TLS]:** ID de certificat client permettant au serveur RADIUS de vérifier l'utilisateur (le dispositif) à l'aide du certificat spécifié.

Ces paramètres IEEE 802.1X sont appelés par la configuration de sécurité dans le cadre d'une certaine configuration d'interface réseau. Pour plus de détails, voir 0.

La présente norme part du principe que la configuration IEEE 802.1X du dispositif est réalisée à l'extérieur du réseau géré IEEE 802.1X. En cas de reconfiguration des paramètres IEEE 802.1X, il est également supposé que cette opération est réalisée à l'extérieur du réseau géré 802.1X.

À noter que dans l'ONVIF 2.0, la prise en charge d'IEEE 802.1X est limitée aux interfaces IEEE 802.11.

8.4.7.1 Création d'une configuration IEEE 802.1X

Cette opération permet de créer l'ensemble de paramètres de configuration IEEE 802.1X du dispositif. Le dispositif doit prendre en charge cette commande (voir Tableau 65) s'il prend en charge IEEE 802.1X. Si le dispositif reçoit cette demande avec une spécification de jeton de configuration qui existe déjà (Dot1XConfigurationToken), il convient qu'il réponde par une erreur "*ter:ReferenceToken*" afin de signaler un conflit de configuration.

Tableau 65 – Commande CreateDot1XConfiguration

CreateDot1XConfiguration		Demande-Réponse	
Nom du message		Description	
CreateDot1XConfigurationRequest		<i>Ce message contient:</i> tt:Dot1XConfiguration Dot1XConfiguration[1][1]	
CreateDot1XConfigurationResponse		<i>Ceci est un message vide.</i>	
Codes de défaut		Description	
env:Receiver ter:ActionNotSupported ter:EAPMethodNotSupported		<i>La méthode EAP suggérée n'est pas prise en charge.</i>	
env:Receiver ter:Action ter:MaxDot1X		<i>Nombre maximal de configurations IEEE 802.1X atteint.</i>	
env:Sender ter:OperationProhibited ter:CertificateID		<i>Erreur d'ID de certificat non valide.</i>	
env:Sender ter:InvalidArgVal ter:ReferenceToken		<i>Dot1XConfigurationToken existe déjà.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDot1X		<i>Configuration IEEE 802.1X non valide.</i>	

8.4.7.2 Définition d'une configuration IEEE 802.1X

Si la commande CreateDot1XConfiguration tente de créer un ensemble de paramètres de configuration, cette opération modifie l'ensemble de paramètres de configuration IEEE 802.1X du dispositif. Un dispositif qui prend en charge IEEE 802.1X doit prendre en charge cette commande (voir Tableau 66).

Tableau 66 – Commande SetDot1XConfigurationRequest

SetDot1XConfiguration		Demande-Réponse
Nom du message	Description	
SetDot1XConfigurationRequest	Ce message contient: tt:Dot1XConfiguration Dot1XConfiguration[1][1]	
SetDot1XConfigurationResponse	Ceci est un message vide.	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:EAPMethodNotSupported	La méthode EAP suggérée n'est pas prise en charge.	
env:Sender ter:OperationProhibited ter:CertificateID	Erreur d'ID de certificat non valide.	
env:Sender ter:OperationProhibited ter:ReferenceToken	Erreur Dot1XConfigurationToken non valide	
env:Sender ter:InvalidArgVal ter:InvalidDot1X	Configuration IEEE 802.1X non valide.	

8.4.7.3 Obtention d'une configuration IEEE 802.1X

Cette opération permet d'extraire un ensemble de paramètres de configuration IEEE 802.1X du dispositif en spécifiant le jeton de configuration (Dot1XConfigurationToken) (voir Tableau 67).

Un dispositif qui prend en charge IEEE 802.1X doit prendre en charge cette commande.

Que la méthode 802.1X de la configuration extraite contienne ou pas un mot de passe, le dispositif ne doit pas inclure l'élément Password dans la réponse.

Tableau 67 – Commande GetDot1XConfiguration

GetDot1XConfiguration		Demande-Réponse
Nom du message	Description	
GetDot1XConfigurationRequest	<i>Ce message contient:</i> tt:ReferenceToken Dot1XConfigurationToken[1][1]	
GetDot1XConfigurationResponse	<i>Ce message contient:</i> tt:Dot1XConfiguration Dot1XConfiguration[1][1]	
Codes de défaut	Description	
env:Sender	<i>Erreur Dot1XConfigurationToken non valide</i>	
ter:OperationProhibited		
ter:ReferenceToken		

8.4.7.4 Obtention des configurations IEEE 802.1X

Cette opération permet d'extraire tous les ensembles de paramètres de configuration IEEE 802.1X existants du dispositif. Le dispositif doit répondre avec toutes les configurations IEEE 802.1X que le client peut obtenir pour savoir dans quelle mesure les configurations IEEE 802.1X existent et comment elles sont configurées (voir Tableau 68).

Un dispositif qui prend en charge IEEE 802.1X doit prendre en charge cette commande.

Que la méthode 802.1X de la configuration extraite contienne ou pas un mot de passe, le dispositif ne doit pas inclure l'élément Password dans la réponse.

Tableau 68 – Commande GetDot1XConfigurations

GetDot1XConfigurations		Demande-Réponse
Nom du message	Description	
GetDot1XConfigurationsRequest	Ceci est un message vide.	
GetDot1XConfigurationsResponse	Ce message contient: tt: Dot1XConfiguration Dot1XConfiguration[0][non limité]	
Codes de défaut	Description	
Pas de défauts spécifiques à la commande!		

8.4.7.5 Suppression d'une configuration IEEE 802.1X

Cette opération permet de supprimer un ensemble de paramètres de configuration IEEE 802.1X du dispositif. La configuration qu'il convient de supprimer est indiquée par "Dot1XConfigurationToken" dans la demande. Un dispositif qui prend en charge IEEE 802.1X doit prendre en charge cette commande (voir Tableau 69).

Tableau 69 – Commande DeleteDot1XConfigurations

DeleteDot1XConfigurations		Demande-Réponse
Nom du message	Description	
DeleteDot1XConfigurationRequest	<i>Ce message contient:</i> tt:ReferenceToken Dot1XConfigurationToken[1][1]	
DeleteDot1XConfigurationResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:OperationProhibited ter:ReferenceToken	<i>Erreur Dot1XConfigurationToken non valide</i>	
env:Receiver ter:OperationProhibited ter:ReferenceToken	<i>La configuration IEEE 802.1X spécifiée ne peut pas être supprimée.</i>	

8.4.8 Création d'un certificat auto-signé

Cette opération génère une paire de clés privée/publique et peut créer un certificat de dispositif auto-signé suite à la génération de la paire de clés. Le certificat est créé à l'aide d'un mécanisme de génération de paire de clés *intégrées* adapté.

Si le dispositif prend en charge la génération de paire de clés *intégrées*, celui qui prend en charge TLS doit prendre en charge cette commande de création de certificat. De même, si un dispositif prend en charge la génération de paire de clés *intégrées*, celui qui prend en charge IEEE 802.1X doit prendre en charge cette commande pour générer la paire de clés. Les certificats et les paires de clés sont identifiés par des ID de certificat. Ces ID sont choisis par le demandeur de génération de certificat ou par le dispositif (si aucune valeur d'ID n'est donnée) (voir Tableau 70).

Tableau 70 – Commande CreateCertificate

CreateCertificate		Demande-Réponse
Nom du message	Description	
CreateCertificateRequest	<i>Ce message contient (le cas échéant) l'ID de certificat demandé et les paramètres supplémentaires demandés: subject, valid not before et valid not after.</i> xs:token CertificateID [0][1] xs:string Subject [0][1] xs:dateTime ValidNotBefore [0][1] xs:dateTime ValidNotAfter [0][1]	
CreateCertificateResponse	<i>Ce message contient le certificat auto-signé généré.</i> tt:Certificate NvtCertificate [1][1]	
Codes de défaut	Description	
env:Receiver ter:Action ter:KeyGeneration	<i>La génération de clé privée/publique n'a pas abouti.</i>	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>L'ID de certificat existe déjà.</i>	
env:Sender ter:InvalidArgVal ter:InvalidDateTime	<i>Le paramètre ValidNotBefore ou ValidNotAfter spécifié n'est pas valide.</i>	

8.4.9 Obtention de certificats

Cette opération permet d'extraire tous les certificats de serveur du dispositif (y compris les certificats auto-signés) pour l'authentification TLS, ainsi que tous les certificats client du dispositif pour l'authentification IEEE 802.1X (voir Tableau 71). Cette commande répertorie uniquement les certificats de serveur TLS et les certificats client IEEE 802.1X du dispositif (pas les certificats CA, ni les certificats racines dignes de confiance). Les certificats sont retournés sous la forme de données binaires. Un dispositif qui prend en charge TLS doit prendre en charge cette commande et les certificats doivent être codés en utilisant les règles de codage ASN.1 [X.681], [X.682], [X.683] DER [X.690]. Voir Tableau 71.

Tableau 71 – Commande GetCertificates

GetCertificates		Demande-Réponse
Nom du message	Description	
GetCertificatesRequest	<i>Ceci est un message vide.</i>	
GetCertificatesResponse	Ce message contient une liste de certificats de dispositif. tt:Certificate NvtCertificate [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.4.10 Obtention de certificats CA

Les certificats CA sont chargés dans un dispositif et utilisés dans les deux cas suivants. L'un est destiné à l'authentification de client TLS dans la fonction de serveur TLS. L'autre est destiné à l'authentification du serveur d'authentification dans la fonction IEEE 802.1X. Cette opération permet d'extraire tous les certificats CA chargés dans un dispositif (voir Tableau 72). Un dispositif qui prend en charge l'authentification client TLS ou IEEE 802.1X doit prendre en charge cette commande, et les certificats retournés doivent être codés en utilisant les règles de codage ASN.1 [X.681], [X.682], [X.683] DER [X.690]. See Table 72.

Tableau 72 – Commande GetCACertificates

GetCACertificates		Demande-Réponse
Nom du message	Description	
GetCACertificatesRequest	<i>Ceci est un message vide.</i>	
GetCACertificatesResponse	Ce message contient une liste de certificats CA. tt:Certificate CACertificate [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.4.11 Obtention de statut de certificat

Cette opération est spécifique à la fonctionnalité TLS. Cette opération extrait le statut (activé/désactivé) des certificats de serveur TLS du dispositif. Un dispositif qui prend en charge TLS doit prendre en charge cette commande (voir Tableau 73).

Tableau 73 – Commande GetCertificatesStatus

GetCertificatesStatus		Demande-Réponse
Nom du message	Description	
GetCertificatesStatusRequest	<i>Ceci est un message vide.</i>	
GetCertificatesStatus-Response	Ce message contient une liste de certificats de serveur de dispositif référencés par ID et leur statut. Le statut est défini comme étant une valeur booléenne (vrai = activé, faux = désactivé). tt:CertificateStatus CertificateStatus [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.4.12 Définition de statut de certificat

Cette opération est spécifique à la fonctionnalité TLS. Cette opération définit le statut (activé/désactivé) des certificats de serveur TLS du dispositif. Un dispositif qui prend en charge TLS doit prendre en charge cette commande (voir Tableau 74). En règle générale, *un seul* certificat de serveur de dispositif peut être activé à la fois.

Tableau 74 – Commande SetCertificatesStatus

SetCertificatesStatus		Demande-Réponse
Nom du message	Description	
SetCertificatesStatusRequest	Ce message contient une liste de certificats de serveur de dispositif référencés par ID et le statut de certificat demandé, c'est-à-dire, activé ou désactivé. tt:CertificateStatus CertificateStatus [0][non limité]	
SetCertificatesStatus-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender	<i>Référence de certificat inconnue.</i>	
ter:InvalidArgVal		
ter:CertificateID		

8.4.13 Obtention de demande de certificat

Cette opération permet de demander une signature de certificat PKCS #10 au dispositif. Le champ d'informations renvoyé doit être mis en forme comme indiqué dans [PKCS#10] ou au format [PKCS#10] codé PEM (voir Tableau 75). Pour que cette commande fonctionne, le dispositif doit déjà disposer d'une paire de clés privée/publique. Il convient que cette paire de

clés soit référencée par *CertificateID* comme indiqué dans la description de paramètre d'entrée. *CertificateID* fait référence à la paire de clés générée à l'aide de la commande *CreateCertificate* définie en 8.4.8.

Un dispositif qui prend en charge la génération de paire de clés *intégrées* prenant en charge TLS ou IEEE 802.1X à l'aide du certificat client, doit prendre en charge cette commande.

Tableau 75 – Commande GetPkcs10Request

GetPkcs10Request		Demande-Réponse
Nom du message	Description	
GetPkcs10RequestRequest	<i>Ce message contient une référence au certificat (paire de clés) et des paramètres de certificat facultatifs pour la demande de certificat. Il est nécessaire que ces attributs soient codés en tant qu'objets DER ASN.1.</i> xs:token CertificateID [1][1] xs:string Subject [0][1] xs:BinaryData Attributes [0][1]	
GetPkcs10RequestResponse	<i>Ce message contient la structure de données de demande PKCS#10.</i> tt:BinaryData Pkcs10Request [1][1]	
Codes de défaut		Description
env:Sender	<i>CertificateID non valide</i>	
ter:InvalidArgVal		
ter:CertificateID		
env:Receiver	<i>La création d'une signature PKCS#10 n'a pas abouti.</i>	
ter:Action		
ter:Signature		

8.4.14 Obtention de statut de certificat de client

Cette opération est spécifique à la fonctionnalité TLS. Cette opération extrait le statut (activé/désactivé) d'authentification de client TLS du dispositif. Un dispositif qui prend en charge TLS doit prendre en charge cette commande (voir Tableau 76).

Tableau 76 – Commande GetClientCertificateMode

GetClientCertificateMode		Demande-Réponse
Nom du message	Description	
GetClientCertificateMode-Request	<i>Ceci est un message vide.</i>	
GetClientCertificateMode-Response	Ce message contient le statut d'authentification de client du dispositif, c'est-à-dire, activé ou désactivé. xs:boolean Enabled [1][1]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.4.15 Définition de statut de certificat de client

Cette opération est spécifique à la fonctionnalité TLS. Cette opération définit le statut (activé/désactivé) d'authentification de client TLS du dispositif. Un dispositif qui prend en charge TLS doit prendre en charge cette commande (voir Tableau 77).

Tableau 77 – Commande SetClientCertificateMode

SetClientCertificateMode		Demande-Réponse
Nom du message	Description	
SetClientCertificateMode-Request	Ce message contient le statut d'authentification de client du dispositif demandé, c'est-à-dire, activé ou désactivé. xs:boolean Enabled [1][1]	
SetClientCertificateMode-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Receiver ter:InvalidArgVal ter:ClientAuth	<i>Tentative d'activation d'authentification de client, mais l'authentification de client n'est pas prise en charge ou non configurée.</i>	

8.4.16 Chargement de certificat de dispositif

Le(s) certificat(s) de serveur TLS ou le(s) certificat(s) de client IEEE 802.1X créés à l'aide de la commande de demande de certificat PKCS#10 peuvent être chargés dans le dispositif grâce à cette commande (voir 8.4.13). L'ID de certificat de la demande doit être présent. Le dispositif peut trier le(s) certificat(s) reçu(s) en fonction de la clé publique et des informations de sujet dans le(s) certificat(s).

L'ID de certificat de la demande correspond à la valeur d'ID souhaitée par le client. Le dispositif est supposé analyser les paires de clés générées présentes dans le dispositif afin

d'identifier celle correspondant au certificat chargé, puis d'associer le certificat à la paire de clés.

Un dispositif qui prend en charge la génération de paire de clés *intégrées* prenant en charge TLS ou IEEE 802.1X doit prendre en charge cette commande (voir Tableau 78).

Les certificats doivent être codés à l'aide des règles de codage ASN.1 [X.681], [X.682], [X.683] DER [X.690].

Cette commande s'applique à tous les types de dispositif, même si le nom de paramètre est appelé NVTCertificate pour des raisons historiques.

Tableau 78 – Commande LoadCertificates

LoadCertificates		Demande-Réponse
Nom du message	Description	
LoadCertificatesRequest	Ce message contient une liste de certificats de dispositif à charger. tt:Certificate NVTCertificate [1][non limité]	
LoadCertificatesResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:CertificateFormat	<i>Format de certificat incorrect ou bien le format n'est pas pris en charge par le dispositif.</i>	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>L'ID de certificat existe déjà.</i>	
env:Sender ter:InvalidArgVal ter:InvalidCertificate	<i>Certificat non valide.</i>	

8.4.17 Chargement de certificat de dispositif avec sa clé privée

Dans certains cas, une autorité de certification ou un organisme équivalent crée un certificat sans disposer d'une demande de signature de certificat PKCS#10. Dans ces cas-là, le certificat est associé à sa clé privée (voir Tableau 79). Cette commande est utilisée pour ce type de situation. La valeur d'ID souhaitée par le client est éventuellement attribuée à l'ID de certificat dans la demande. Si l'ID de certificat n'est pas précisé dans la demande, le dispositif peut choisir l'ID en conséquence.

Cette opération permet d'importer une paire de clés privée/publique dans le dispositif.

Les certificats doivent être codés à l'aide des règles de codage ASN.1 [X.681], [X.682], [X.683] DER [X.690].

Un dispositif qui ne prend pas en charge la génération de paire de clés intégrées et prend en charge TLS ou IEEE 802.1X à l'aide du certificat client, doit prendre en charge cette commande. Un dispositif qui prend en charge la génération de paire de clés intégrées PEUT prendre en charge cette commande. Il convient que la politique de sécurité d'un dispositif qui prend en charge cette opération s'assure que la clé privée est suffisamment protégée.

Tableau 79 – Commande LoadCertificateWithPrivateKey

LoadCertificateWithPrivateKey		Demande-Réponse
Nom du message	Description	
LoadCertificateWithPrivateKeyRequest	Ce message contient une paire de clés privée/publique à importer. tt:CertificateWithPrivateKey CertificateWithPrivateKey[1][non limité]	
LoadCertificateWithPrivateKeyResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env: Sender ter:InvalidArgVal ter:CertificateFormat	Format de certificat incorrect ou bien le format n'est pas pris en charge par le dispositif.	
env:Sender ter:InvalidArgVal ter:CertificateID	L'ID de certificat existe déjà.	
env: Sender ter:InvalidArgVal ter: KeysNotMatching	La clé publique et la clé privée ne correspondent pas.	

8.4.18 Obtention de demande d'informations de certificat

Cette opération permet de demander les informations spécifiées par l'ID de certificat. Il convient que le dispositif réponde par les données "Issuer DN", "Subject DN", "Key usage", "Extended key usage", "Key Length", "Version", "Serial Number", "Signature Algorithm" et "Validity" comme informations du certificat, tant que le dispositif peut extraire ces informations du certificat spécifié. IssuerDN et SubjectDN doivent être codés à l'aide des règles de la norme [RFC 4514].

Il convient qu'un dispositif qui prend en charge TLS ou IEEE 802.1X prenne en charge cette commande (voir Tableau 80).

Tableau 80 – Commande GetCertificateInformation

GetCertificateInformation	Demande-Réponse
Nom du message	Description
GetCertificateInformationRequest	<i>Ce message contient:</i> <i>CertificateID: jeton du certificat.</i> <i>xs: token CertificateID [1][1]</i>
GetCertificateInformationResponse	<i>Ce message contient:</i> <i>tt:CertificateInformation CertificateInformation[1][1]</i>
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:CertificateID	<i>ID de certificat non valide</i>

8.4.19 Chargement de certificats CA

Cette commande est utilisée lorsque des certificats CA ou racines dignes de confiance sont à charger afin de vérifier leur homologue (vérification du certificat de client dans la fonction TLS ou du certificat de serveur dans la fonction IEEE 802.1X) (voir Tableau 81).

Un dispositif qui prend en charge TLS ou IEEE 802.1X doit prendre en charge cette commande. Le dispositif doit prendre en charge le format DER. Les autres formats peuvent être pris en charge par le dispositif. Le dispositif peut trier les certificats reçus en fonction des informations de clé publique et de sujet qu'ils contiennent. Soit le chargement de tous les certificats CA aboutit, soit un message de défaut doit être retourné sans charger de certificat CA.

Tableau 81 – Commande LoadCACertificates

LoadCACertificates		Demande-Réponse
Nom du message	Description	
LoadCACertificatesRequest	Ce message contient une liste de certificats CA de dispositif à charger. tt:Certificate CACertificate [1][non limité]	
LoadCACertificatesResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:CertificateFormat	<i>Format de certificat incorrect ou bien le format n'est pas pris en charge par le dispositif.</i>	
env:Sender ter:InvalidArgVal ter:CACertificateID	<i>L'ID de certificat CA existe déjà.</i>	
env:Receiver ter:OperationProhibited ter:MaxCertificates	<i>Nombre maximal de certificats déjà chargés.</i>	

8.4.20 Suppression de certificat

Cette opération permet de supprimer un ou plusieurs certificats. Le dispositif PEUT également supprimer la paire de clés privée/publique associée au certificat à supprimer. Le dispositif qui prend en charge TLS ou IEEE 802.1X doit prendre en charge la suppression d'un ou de plusieurs certificats grâce à cette commande (voir Tableau 82). Soit la suppression de tous les certificats aboutit, soit un message de défaut doit être retourné sans supprimer de certificat.

Tableau 82 – Commande DeleteCertificates

DeleteCertificates		Demande-Réponse
Nom du message	Description	
DeleteCertificatesRequest	Ce message supprime des certificats identifiés avec le paramètre CertificateID. xs:token CertificateID[1][non limité]	
DeleteCertificatesResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:CertificateID	<i>Référence de certificat inconnue.</i>	
env:Receiver ter:OperationProhibited ter:CertificateID	<i>Les certificats spécifiés ne peuvent pas être supprimés.</i>	

8.4.21 Obtention de l'utilisateur distant

Cette opération renvoie l'utilisateur distant configuré (le cas échéant). Un dispositif prenant en charge le traitement des utilisateurs distants doit prendre en charge cette opération (voir Tableau 83). L'utilisateur est uniquement valide pour le profil WS-UserToken ou en tant qu'utilisateur HTTP/RTSP.

L'algorithme d'utilisation de mot de passe est décrit en 5.12.2.2. *Un dispositif ne doit jamais retourner le champ Password de RemoteUser.*

Tableau 83 – Commande GetRemoteUser

GetRemoteUser		Demande-Réponse
Nom du message	Description	
GetRemoteUserRequest	<i>Ceci est un message vide.</i>	
GetRemoteUserResponse	<i>Ce message contient l'utilisateur distant configuré (le cas échéant). Les valeurs retournées sont:</i> • <i>xs:string Username [1][1]</i> • <i>xs:boolean UseDerivedPassword [1][1]</i> <i>tt:RemoteUser: RemoteUser [0][1]</i>	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:NotRemoteUser	<i>Le traitement des utilisateurs distants n'est pas pris en charge</i>	

8.4.22 Définition de l'utilisateur distant

Cette opération permet de définir l'utilisateur distant. Un dispositif prenant en charge le traitement des utilisateurs distants doit prendre en charge cette opération (voir Tableau 84). L'utilisateur est uniquement valide pour le profil WS-UserToken ou en tant qu'utilisateur HTTP/RTSP.

Le mot de passe défini doit toujours être le mot de passe d'origine (pas déduit).

Si **UseDerivedPassword** est défini, la déduction de mot de passe doit être réalisée par le dispositif lors de la connexion à un dispositif distant. L'algorithme d'utilisation de la déduction de mot de passe est décrit en 5.12.2.2.

Pour supprimer l'utilisateur distant, il convient d'appeler **SetRemoteUser** sans le paramètre **RemoteUser**.

Tableau 84 – Commande SetRemoteUser

SetRemoteUser		Demande-Réponse
Nom du message	Description	
SetRemoteUserRequest	<i>Ce message contient l'utilisateur distant. Les valeurs possibles sont:</i> <i>xs:string Username [1][1]</i> <i>xs:string Password [0][1]</i> <i>xs:boolean UseDerivedPassword [1][1]</i> <i>tt:RemoteUser: RemoteUser [0][1]</i>	
SetRemoteUserResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:NotRemoteUser	<i>Traitement des utilisateurs distants non pris en charge</i>	

8.4.23 Obtention de la référence de point terminal

Un client peut demander la propriété d'adresse de référence de point terminal de service du dispositif qui peut permettre de déduire le mot de passe équivalent pour l'opération d'utilisateur distant. Le dispositif doit prendre en charge la commande GetEndpointReference retournant la propriété d'adresse de la référence de point terminal de service du dispositif (voir Tableau 85).

Tableau 85 – Commande GetEndpointReference

GetEndpointReference		Demande-Réponse
Nom du message	Description	
GetEndpointReferenceRequest	<i>Ceci est un message vide.</i>	
GetEndpointReferenceResponse	<i>L'URL demandée.</i> <i>xs:string GUID [1][1]</i>	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.5 Entrée/sortie (E/S)

Les commandes du présent paragraphe sont obsolètes. Voir l'Article 0.

Les commandes d'entrée/sortie (E/S) sont utilisées pour contrôler l'état ou observer le statut des ports E/S. Si le dispositif comporte des ports E/S, il doit prendre en charge les

commandes E/S.

8.5.1 Obtention de sorties de relais

Cette opération permet d'extraire une liste de l'ensemble des sorties de relais disponibles et leurs paramètres (voir Tableau 86).

Tableau 86 – Commande GetRelayOutputs

GetRelayOutputs		Demande-Réponse
Nom du message	Description	
GetRelayOutputsRequest	<i>Ceci est un message vide.</i>	
GetRelayOutputsResponse	<i>Ce message contient un ensemble de sorties de relais.</i> tt:RelayOutput RelayOutputs [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

8.5.2 Définition de réglages de sortie de relais

Cette opération définit les paramètres d'une sortie de relais (voir Tableau 87).

Le relais peut fonctionner dans deux modes de relais:

- Bistable – après la définition de l'état, le relais reste dans cet état;
- Monostable – après la définition de l'état, le relais retourne à son état de veille après le temps spécifié.

L'état de veille physique d'une sortie de relais peut être configuré en attribuant la valeur "open" (ouvert) ou "closed" (fermé) à IdleState (inversion du comportement de relais).

L'état de veille "open" signifie que le relais est ouvert lorsque l'état de relais est "inactive" (inactif) par l'intermédiaire de la commande de déclenchement (voir 0) et fermé lorsque l'état de relais est "active" (actif) à l'aide de la même commande.

L'état de veille "closed" signifie que le relais est fermé lorsque l'état du relais est "inactive" par l'intermédiaire de la commande de déclenchement (voir 0) et ouvert lorsque l'état est "active" à l'aide de la même commande.

Tableau 87 – Commande SetRelayOutputSettings

SetRelayOutputSettings		Demande-Réponse
Nom du message	Description	
SetRelayOutputSettings Request	<i>Ce message contient:</i> <i>"RelayToken": Référence de jeton vers la sortie de relais demandée.</i> <i>"RelayOutputSettings": Paramètres du relais</i> . tt:ReferenceToken RelayOutputToken [1][1] tt:RelayOutputSettings RelayOutputSettings [1][1]	
SetRelayOutputSettings Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Référence de jeton de relais inconnue.</i>	
env:Sender ter:InvalidArgVal ter:ModeError	<i>Délai monostable non valide.</i>	

8.5.3 Déclenchement de sortie de relais

Cette opération déclenche une sortie de relais¹ (voir Tableau 88).

¹ Il n'existe pas de commande GetRelayState. L'état logique réel de la sortie de relais est transmis par notification et leurs propriétés.

Tableau 88 – Commande SetRelayOutputState

SetRelayOutputState		Demande-Réponse
Nom du message	Description	
SetRelayOutputStateRequest	<i>Ce message contient:</i> <i>"RelayToken": Référence de jeton vers la sortie de relais demandée.</i> <i>"LogicalState": Demande de déclenchement, c'est-à-dire, actif ou inactif.</i> tt:ReferenceToken RelayOutputToken [1][1] tt:RelayLogicalState LogicalState [1][1]	
SetRelayOutputStateResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Référence de jeton de relais inconnue.</i>	

8.5.4 Opérations auxiliaires

Le présent paragraphe décrit les opérations de gestion des commandes auxiliaires prises en charge par un dispositif (contrôle d'une lampe à infrarouge, d'un chauffage, d'un dispositif d'essuyage ou d'un thermomètre, par exemple) connecté au dispositif.

Les commandes prises en charge peuvent être extraites par le paramètre AuxiliaryData déduit de la réponse de la commande GetCapabilities. Il convient que la commande transmise à l'aide de cette commande corresponde à l'une des commandes prises en charge figurant dans la réponse AuxiliaryData. Si la réponse de commande de fonctionnalité répertorie uniquement la commande *irlampon*, l'argument SendAuxiliaryCommand est *irlampon*, qui peut indiquer l'allumage de la lampe à infrarouge connectée.

Un dispositif qui prend en charge une fonctionnalité de service auxiliaire doit prendre en charge cette commande (voir Tableau 89).

Tableau 89 – Commande Send auxiliary

SendAuxiliaryCommand		Demande-Réponse
Nom du message	Description	
SendAuxiliaryCommandRequest	<i>Ce message contient la commande auxiliaire.</i> tt:AuxiliaryData AuxiliaryCommand[1][1]	
SendAuxiliaryCommandResponse	<i>La réponse contient la réponse auxiliaire.</i> tt:AuxiliaryData AuxiliaryCommandResponse[0][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:AuxiliaryDataNotSupported	<i>Le AuxiliaryCommand demandé n'est pas pris en charge.</i>	

8.6 Codes de défaut spécifiques au service

Le Tableau 90 répertorie les codes de défaut spécifiques au service de dispositif. De plus, chaque commande peut également générer un défaut générique (voir Tableau 6).

Les défauts spécifiques sont définis sous la forme d'un sous-code de défaut générique (voir 5.11.2.1). Le sous-code générique parent est le sous-code en haut de chaque ligne ci-dessous, le sous-code de défaut spécifique se trouvant en bas de la cellule.

Tableau 90 – Codes de défaut spécifiques au service de dispositif

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Receiver	ter:Action	La politique est vide	Le fichier de politique du dispositif n'existe pas ou est vide.
	ter:EmptyPolicy		
env:Receiver	ter:Action	La liste de domaines d'application est vide	La liste de domaines d'application est vide.
	ter:EmptyScope		
env:Receiver	ter:Action	La mise à niveau a échoué	La mise à niveau de micrologiciel a échoué.
	ter:FirmwareUpgradeFailed		
env:Receiver	ter:Action	La génération de clé a échoué	La génération de clé privée/publique n'a pas abouti.
	ter:KeyGeneration		
env:Receiver	ter:Action	La création d'une signature a échoué	La création d'une signature PKCS#10 n'a pas abouti.
	ter:Signature		
env:Receiver	ter:InvalidArgVal	Authentification de client non prise en charge	Tentative d'activer l'authentification de client, mais l'authentification de client n'est pas prise en charge ou non configurée.
	ter:ClientAuth		
env:Receiver	ter:Action	Trop grand nombre d'utilisateurs	Nombre maximal d'utilisateurs pris en charge dépassé.
	ter:TooManyUsers		
env:Receiver	ter:Action	Liste de domaines d'application trop grande	La liste de domaines d'application dépasse le nombre de domaines d'application pris en charge.
	ter:TooManyScopes		
env:Receiver	ter:ActionNotSupported	Le service n'est pas pris en charge.	La catégorie de service WSDL demandée n'est pas prise en charge par le dispositif.
	ter:NoSuchService		
env:Sender	ter:InvalidArgs	Aucun journal d'accès disponible	Aucune information de journal d'accès n'est disponible.
	ter:AccesslogUnavailable		
env:Sender	ter:InvalidArgVal	Format non valide	Format de certificat incorrect ou bien le format n'est pas pris en charge par le dispositif.
	ter:CertificateFormat		
env:Sender	ter:InvalidArgVal	ID de certificat non valide	Référence de certificat inconnue ou bien l'ID de certificat existe déjà.
	ter:CertificateID		
env:Sender	ter:InvalidArgVal	ID de certificat CA non valide	Référence de certificat CA inconnue ou bien l'ID de certificat CA existe déjà.
	ter:CACertificateID		

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Sender	ter:InvalidArgVal	Fichier non valide	Le(s) fichier(s) de sauvegarde n'est (ne sont) pas valide(s).
	ter:InvalidBackupFile		
env:Sender	ter:InvalidArgVal	Date et heure non valides.	Une date ou une heure non valide a été spécifiée.
	ter:InvalidDateTime		
env:Sender	ter:InvalidArgVal	Nom non valide	Le nom de serveur NTP suggéré est non valide.
	ter:InvalidDnsName		
env:Sender	ter:InvalidArgs	Micrologiciel non valide	Le micrologiciel est non valide, c'est-à-dire qu'il n'est pas pris en charge par ce dispositif.
	ter:InvalidFirmware		
env:Sender	ter:InvalidArgVal	Adresse non valide	L'adresse de passerelle fournie n'est pas valide.
	ter:InvalidGatewayAddress		
env:Sender	ter:InvalidArgVal	Nom non valide	Le dispositif ne peut pas accepter le nom d'hôte demandé.
	ter:InvalidHostname		
env:Sender	ter:InvalidArgVal	Vitesse non valide	La vitesse suggérée n'est pas prise en charge.
	ter:InvalidInterfaceSpeed		
env:Sender	ter:InvalidArgVal	Type non valide	Le type d'interface réseau suggéré n'est pas pris en charge.
	ter:InvalidInterfaceType		
env:Sender	ter:InvalidArgVal	Adresse non valide	L'adresse IPv4 suggérée est non valide.
	ter:InvalidIPv4Address		
env:Sender	ter:InvalidArgVal	L'adresse n'existe pas	L'adresse IPv4 à supprimer n'existe pas.
	ter:NoIPv4Address		
env:Sender	ter:InvalidArgVal	Adresse non valide	L'adresse IPv6 suggérée est non valide.
	ter:InvalidIPv6Address		
env:Sender	ter:InvalidArgVal	L'adresse n'existe pas	L'adresse IPv6 à supprimer n'existe pas.
	ter:NoIPv6Address		
env:Sender	ter:InvalidArgVal	Données non valides	La valeur MTU est non valide.
	ter:InvalidMtuValue		
env:Sender	ter:InvalidArgVal	Jeton non valide	Le jeton d'interface réseau fourni n'existe pas.
	ter:InvalidNetworkInterface		
env:Sender	ter:InvalidArgVal	Données non valides	Un fuseau horaire non valide a été spécifié.
	ter:InvalidTimeZone		
env:Sender	ter:InvalidArgVal	La liste est pleine	Aucun filtre IP supplémentaire ne peut être ajouté, car la liste de filtres IP est pleine.
	ter:IPFilterListIsFull		
env:Sender	ter:InvalidArgVal	Données non valides	Délai monostable non valide.
	ter:ModeError		

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Sender	ter:InvalidArgs	Format non valide	La politique demandée ne peut pas être définie en raison d'un format de politique inconnu.
	ter:PolicyFormat		
env:Sender	ter:InvalidArgVal	Jeton de relais inconnu.	La référence de jeton est inconnue.
	ter:RelayToken		
env:Sender	ter:InvalidArgVal	Le service n'est pas pris en charge.	Le service réseau fourni n'est pas pris en charge.
	ter:ServiceNotSupported		
env:Sender	ter:InvalidArgVal	Aucune information de prise en charge disponible	Aucune information de prise en charge n'est disponible.
	ter:SupportInformationUnavailable		
env:Sender	ter:InvalidArgs	Aucun journal système disponible	Aucune information de journal système n'est disponible.
	ter:SystemlogUnavailable		
env:Sender	ter:InvalidArgVal	Nom d'utilisateur non reconnu	Nom d'utilisateur NON reconnu
	ter:UsernameMissing		
env:Sender	ter:OperationProhibited	Tentative de suppression de domaine d'application fixe	Tentative de suppression de domaine d'application fixe, commande rejetée.
	ter:FixedScope		
env:Sender	ter:InvalidArgVal	Le domaine d'application n'existe pas	Tentative de suppression d'un domaine d'application qui n'existe pas.
	ter:NoScope		
env:Sender	ter:OperationProhibited	Mot de passe trop faible	Mot de passe trop faible
	ter:Password		
env:Sender	ter:OperationProhibited	Mot de passe trop long	Le mot de passe est trop long.
	ter>PasswordTooLong		
env:Sender	ter:OperationProhibited	Mot de passe trop long	Le mot de passe est trop court.
	ter:UsernameTooShort		
env:Sender	ter:OperationProhibited	Tentative d'écrasement de réglage de domaine d'application de dispositif permanent	Un paramètre de domaine d'application remplace un réglage de domaine d'application de dispositif permanent, commande rejetée.
	ter:ScopeOverwrite		
env:Sender	ter:OperationProhibited	Le nom d'utilisateur existe déjà.	Le nom d'utilisateur existe déjà.
	ter:UsernameClash		
env:Sender	ter:OperationProhibited	Nom d'utilisateur trop long	Le nom d'utilisateur est trop long.
	ter:UsernameTooLong		
env:Receiver	ter:ActionNotSupported	Non supporté	La configuration IEEE 802.11 n'est pas prise en charge.
	ter:InvalidDot11		
env:Sender	ter:InvalidArgVal	Non pris en charge	Le mode de sécurité

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
	ter:InvalidSecurityMode		sélectionné n'est pas pris en charge.
env:Sender	ter:InvalidArgVal	Non pris en charge	Le mode de station sélectionné n'est pas pris en charge.
	ter:InvalidStationMode		
env:Sender	ter:InvalidArgVal	Valeur IEEE 802.11 absente	La configuration de sécurité ne contient pas de valeur IEEE 802.11.
	ter:MissingDot11		
env:Sender	ter:InvalidArgVal	Valeur PSK absente	La configuration de sécurité ne contient pas de valeur PSK.
	ter:MissingPSK		
env:Sender	ter:InvalidArgVal	Valeur IEEE 802.1X absente	La configuration de sécurité ne contient pas de valeur IEEE 802.1X.
	ter:MissingDot1X		
env:Sender	ter:InvalidArgVal	La valeur IEEE 802.1X est incompatible	La valeur IEEE 802.1X de la configuration de sécurité n'est pas compatible avec l'interface réseau.
	ter:IncompatibleDot1X		
env:Sender	ter:InvalidArgVal	Pas IEEE 802.11	L'interface n'est pas une interface IEEE 802.11.
	ter:NotDot11		
env:Sender	ter:InvalidArgVal	Configuration IEEE 802.1X non valide.	La configuration IEEE 802.1X n'est pas valide.
	ter:InvalidDot1X		
env:Receiver	ter:Action	IEEE 802.11 non connecté	Le réseau IEEE 802.11 n'est pas connecté.
	ter:NotConnectedDot11		
env:Receiver	ter:ActionNotSupported	ScanAvailableIEEE802.11Networks n'est pas pris en charge.	ScanAvailableIEEE802.11Networks n'est pas pris en charge.
	ter:NotScanAvailable		
env:Receiver	ter:ActionNotSupported	Le traitement des utilisateurs distants n'est pas pris en charge.	Le traitement des utilisateurs distants n'est pas pris en charge.
	ter:NotRemoteUser		
env:Receiver	ter:ActionNotSupported	La méthode EAP suggérée n'est pas prise en charge.	La méthode EAP suggérée n'est pas prise en charge.
	ter:EAPMethodNotSupported		
env:Receiver	ter:Action	Nombre maximal de configurations IEEE 802.1X atteint.	Le dispositif a atteint le nombre maximal de configurations IEEE 802.1X.
	ter:MaxDot1X		
env:Receiver	ter:OperationProhibited	La configuration IEEE 802.1X spécifiée ne peut pas être supprimée.	La configuration IEEE 802.1X spécifiée ne peut pas être supprimée.
	ter:ReferenceToken		

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Receiver	ter:OperationProhibited	Le/les certificat(s) spécifié(s) ne peuvent pas être supprimés.	Le/les certificat(s) spécifié(s) ne peuvent pas être supprimés.
	ter:CertificateID		
env:Sender	ter:OperationProhibited	Erreur Dot1XConfiguration Token non valide.	Le jeton de configuration IEEE 802.1X spécifié n'est pas valide.
	ter:ReferenceToken		
env:Sender	ter:OperationProhibited	Erreur d'ID de certificat non valide.	L'ID de certificat spécifié n'est pas valide.
	ter:CertificateID		
env:Sender	ter:InvalidArgVal	Dot1XConfiguration Token existe déjà.	Le Dot1XConfigurationToken spécifié existe déjà dans le dispositif.
	ter:ReferenceToken		
env:Sender	ter:InvalidArgVal	Certificat non valide.	Le certificat spécifié n'est pas valide.
	ter:InvalidCertificate		
env:Receiver	ter:OperationProhibited	Nombre maximal de certificats déjà chargés.	Le dispositif a atteint le nombre maximal de certificats chargés.
	ter:MaxCertificates		
env:Sender	ter:OperationProhibited	Mot de passe trop faible	Mot de passe trop faible
	ter:PasswordTooWeak		
env:Sender	ter:InvalidArgVal	Le AuxiliaryCommand demandé n'est pas pris en charge.	Le AuxiliaryCommand demandé n'est pas pris en charge.
	ter:AuxiliaryDataNotSupported		
env:Sender	ter:InvalidArgVal	Valeur de délai d'expiration spécifiée non valide.	La valeur TimeOut spécifiée n'est pas valide.
	ter:InvalidTimeoutValue		
env:Sender	ter:OperationProhibited	Nombre d'octets disponibles dépassé.	Nombre d'octets disponibles dépassé.
	ter:DataLengthOver		

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Sender	ter:OperationProhibited	La séquence de caractères (délimiteur) n'est pas prise en charge.	La séquence de caractères (délimiteur) n'est pas prise en charge.
	ter:DelimiterNotSupport		
env:Receiver	ter:OperationProhibited	Le dispositif n'est pas prêt à fonctionner en mode de commande.	Le dispositif n'est pas prêt à fonctionner en mode de commande.
	ter:InvalidMode		
env:Sender	ter:InvalidArgVal	Suppression d'utilisateur fixe	Le client tente de supprimer un utilisateur fixe.
	ter:FixedUser		
env:Sender	ter:OperationProhibited	Le niveau d'utilisateur anonyme n'est pas autorisé.	Le niveau d'utilisateur anonyme n'est pas autorisé.
	ter:AnonymousNotAllowed		
Env:Sender	ter:InvalidArgVal	Les clés ne correspondent pas	La clé publique et la clé privée ne correspondent pas.
	ter:KeysNotMatching		

9 Service ES de dispositif

Ce service propose des commandes permettant d'extraire et de configurer les entrées et sorties physiques d'un dispositif.

Les commandes permettant de demander les entrées et sorties vidéo et audio disponibles sont définies avec les commandes permettant de demander les relais disponibles. Ce service offre également des fonctions de demande et de modification de la configuration de ces entités.

Un dispositif disposant de sources et de sorties physiques doit prendre en charge ce service comme indiqué en C.3.

Certaines fonctionnalités de ce service coïncident avec les fonctionnalités définies dans le Media Service. S'il faut qu'un dispositif (un NVT, par exemple) mette en œuvre ces services, il convient qu'il utilise les commandes définies dans ce service pour configurer ses entrées et sorties audio ou ses sources vidéo.

9.1 VideoOutputs

9.1.1 General

Le type VideoOutput représente les sorties vidéo physiques d'un dispositif qui peuvent être connectées à un moniteur pour afficher le signal vidéo. La structure contient les paramètres de présentation qui peuvent être configurés à l'aide du service d'affichage (voir Article 14).

9.1.2 GetVideoOutputs

Cette commande permet de répertorier toutes les sorties vidéo disponibles d'un dispositif. Un dispositif comportant une ou plusieurs sorties vidéo physiques doit prendre en charge la liste des sorties vidéo disponibles grâce à la commande GetVideoOutputs (voir Tableau 91).

Tableau 91 – Commande GetVideoOutputs

GetVideoOutputs		Demande-Réponse
Nom du message	Description	
GetVideoOutputsRequest	<i>Ceci est un message vide.</i>	
GetVideoOutputsResponse	<i>Contient une liste des structures décrivant toutes les sorties vidéo disponibles du dispositif. Si un dispositif ne contient aucun VideoOutputs, une liste vide est retournée.</i> tt:VideoOutput VideoOutputs [0][non limité]	
Codes de défaut	Description	
<i>Pas de codes de défaut spécifiques.</i>		

9.2 VideoOutputConfiguration

9.2.1 GetVideoOutputConfiguration

Cette opération demande la configuration d'une sortie vidéo. Un dispositif comportant une ou plusieurs sorties vidéo doit prendre en charge l'extraction de VideoOutputConfiguration grâce à cette commande (voir Tableau 92).

Tableau 92 – Commande GetVideoOutputConfiguration

GetVideoOutputConfiguration		Demande-Réponse
Nom du message	Description	
GetVideoOutputConfigurationRequest	<i>Ce message contient le jeton de VideoOutput.</i> tt:ReferenceToken VideoOutputToken[1][1]	
GetVideoOutputConfigurationResponse	<i>Ce message contient le VideoOutputConfiguration demandé avec le jeton correspondant.</i> tt:VideoOutputConfiguration VideoOutputConfiguration [1][1]	
Codes de défaut	Description	
env:Sender	<i>Le VideoOutput demandé indiqué avec VideoOutputToken n'existe pas.</i>	
ter:InvalidArgVal		
ter:NoVideoOutput		

9.2.2 SetVideoOutputConfiguration

Cette opération modifie une configuration de sortie vidéo (voir Tableau 93). Un dispositif comportant une ou plusieurs sorties vidéo doit prendre en charge les paramètres de sa configuration de sortie vidéo grâce à cette commande.

Tableau 93 – Commande SetVideoOutputConfiguration

SetVideoOutputConfiguration		Demande-Réponse
Nom du message	Description	
SetVideoOutputConfiguration-Request	<i>L'élément Configuration contient la configuration VideoOutput modifiée.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:VideoOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoOutputConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	La sortie vidéo demandée n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	

9.2.3 GetVideoOutputConfigurationOptions

Cette opération demande les VideoOutputConfigurationOptions d'un VideoOutput. Un dispositif comportant une ou plusieurs sorties vidéo doit prendre en charge l'extraction de VideoOutputConfigurationOptions grâce à cette commande (voir Tableau 94).

Tableau 94 – Commande GetVideoOutputConfigurationOptions

GetVideoOutputConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetVideoOutputConfiguration-OptionsRequest	<i>L'élément VideoOutputToken spécifie le VideoOutput dont les options sont demandées. Le VideoOutput doit exister dans le dispositif.</i> tt:ReferenceToken VideoOutputToken[1][1]	
GetVideoOutputConfiguration-OptionsResponse	<i>La réponse contient les VideoOutputOptions du dispositif.</i> tt:VideoOutputConfigurationOptions VideoOutputOptions[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie vidéo demandée n'existe pas</i>	

9.3 VideoSources

9.3.1 Généralités

Une VideoSource représente une entrée vidéo physique. La structure contient la résolution en pixels de la vidéo, le taux de trame et les paramètres d'imagerie. Les paramètres d'imagerie peuvent être manipulés par l'intermédiaire de la commande ImagingService, si elle est prise en charge et contient des paramètres de mise au point, d'exposition et de luminosité, par exemple.

9.3.2 GetVideoSources

Cette opération permet d'extraire toutes les sources vidéo disponibles pour le dispositif (voir Tableau 95). Un dispositif comportant une ou plusieurs entrées vidéo doit prendre en charge la liste des sources vidéo disponibles grâce à la commande GetVideoSources.

Tableau 95 – Commande GetVideoSources

GetVideoSources		Demande-Réponse
Nom du message	Description	
GetVideoSourcesRequest	<i>Ceci est un message vide.</i>	
GetVideoSourcesResponse	<i>Contient une liste des structures décrivant toutes les sources vidéo disponibles du dispositif. Si un dispositif ne contient aucune source vidéo, une liste vide est retournée.</i> tt:VideoSource VideoSource [0][non limité]	
Codes de défaut	Description	
<i>Pas de codes de défaut spécifiques.</i>		

9.4 VideoSourceConfiguration

Une VideoSourceConfiguration (configuration de source vidéo) contient une référence à une VideoSource (source vidéo) et une structure Bounds (bornes) contenant la zone de pixels totale de la VideoSource (source vidéo) ou une sous-portion de celle-ci. Les Bounds et VideoSource définissent l'image qui est transmise en continu à un client.

9.4.1 GetVideoSourceConfiguration

Cette opération permet de répertorier les configurations de source vidéo d'un élément VideoSource. Un dispositif comportant une ou plusieurs sources vidéo doit prendre en charge la commande GetVideoSourceConfigurations (voir Tableau 96).

Tableau 96 – Commande GetVideoSourceConfiguration

GetVideoSourceConfiguration		Demande-Réponse
Nom du message	Description	
GetVideoSourceConfigurationRequest	<i>Ce message contient le jeton de l'entrée vidéo.</i> tt:ReferenceToken VideoSourceToken[1][1]	
GetVideoSourceConfigurationResponse	<i>Ce message contient la VideoSourceConfiguration demandée avec le jeton correspondant.</i> tt:VideoSourceConfiguration VideoSourceConfiguration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoVideoSource	<i>Le VideoSource demandé indiqué avec VideoSourceToken n'existe pas.</i>	

9.4.2 SetVideoSourceConfiguration

Cette opération modifie une configuration d'entrée vidéo. Un dispositif comportant une ou plusieurs sources vidéo doit prendre en charge les paramètres de VideoSourceConfiguration grâce à cette commande (voir Tableau 97).

Tableau 97 – Commande SetVideoSourceConfiguration

SetVideoSourceConfiguration		Demande-Réponse
Nom du message	Description	
SetVideoSourceConfiguration-Request	<i>L'élément Configuration contient la configuration VideoSource modifiée. L'élément Configuration contient un élément qui spécifie le VideoSource dont la configuration est à modifier. Le VideoSource doit exister dans le dispositif.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:VideoSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoSourceConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoVideoSource	Le VideoSource demandé n'existe pas	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	

9.4.3 GetVideoSourceConfigurationOptions

Cette opération demande les VideoSourceConfigurationOptions d'un VideoSource. Un dispositif comportant une ou plusieurs sources vidéo doit prendre en charge cette commande (voir Tableau 98).

Tableau 98 – Commande GetVideoSourceConfiguartionOptions

GetVideoSourceConfiguartionOptions		Demande-Réponse
Nom du message	Description	
GetVideoSourceConfiguration-OptionsRequest	<i>L'élément VideoSourceToken spécifie l'entrée vidéo dont les options sont demandées. L'entrée vidéo doit exister dans le dispositif</i> tt:ReferenceToken VideoSourceToken[1][1]	
GetVideoSourceConfiguration-OptionsResponse	<i>L'élément VideoSourceOptions renvoie les limites valides, ainsi qu'un élément qui délivre le VideoSourceToken disponible. La Source dont les options sont demandées doit être indiquée dans ce champ.</i> tt:VideoSourceConfigurationOptions VideoSourceOptions[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoVideoSource	<i>L'entrée vidéo demandée n'existe pas</i>	

9.5 AudioOutputs

9.5.1 Généralités

La sortie audio représente les sorties audio physiques qui peuvent être connectées à un haut-parleur.

9.5.2 GetAudioOutputs

Cette commande répertorie toutes les sorties audio disponibles d'un dispositif. Un dispositif comportant une ou plusieurs sorties audio physiques doit prendre en charge la liste des sorties audio disponibles grâce à la commande GetAudioOutputs (voir Tableau 99).

Tableau 99 – Commande GetAudioOutputs

GetAudioOutputs		Demande-Réponse
Nom du message	Description	
GetAudioOutputsRequest	<i>Ceci est un message vide.</i>	
GetAudioOutputsResponse	<i>Contient une liste des structures décrivant toutes les sorties audio disponibles du dispositif. Si un dispositif ne contient aucun AudioOutputs, une liste vide est retournée.</i> tt:AudioOutput AudioOutputs [0][non limité]	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio ou AudioOutputs n'est pas pris en charge par le dispositif</i>	

9.6 AudioOutputConfiguration

Un élément AudioOutputConfiguration contient une référence à un AudioOutput existant. La configuration d'AudioOutput contient un paramètre de contrôle du niveau de sortie.

9.6.1 GetAudioOutputConfiguration

Cette opération demande l'AudioOutputConfiguration d'un AudioOutput. Un dispositif comportant un ou plusieurs AudioOutputs doit prendre en charge l'extraction d'AudioOutputConfiguration grâce à cette commande (voir Tableau 100).

Tableau 100 – Commande GetAudioOutputConfiguration

GetAudioOutputConfiguration		Demande-Réponse
Nom du message	Description	
GetAudioOutputConfigurationRequest	<i>Ce message contient le jeton d'AudioOutput.</i> tt:ReferenceToken AudioOutputToken[1][1]	
GetAudioOutputConfigurationResponse	<i>Ce message contient l'AudioOutputConfiguration demandé avec le jeton correspondant.</i> tt:AudioOutputConfiguration AudioOutputConfiguration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAudioOutput	<i>L'élément AudioOutput demandé indiqué avec AudioOutputToken n'existe pas.</i>	

9.6.2 SetAudioOutputConfiguration

Cette opération modifie une configuration de sortie audio. Un dispositif comportant une ou plusieurs sorties audio doit prendre en charge les paramètres d'AudioOutputConfiguration grâce à cette commande (voir Tableau 101).

Tableau 101 – Commande SetAudioOutputConfiguration

SetAudioOutputConfiguration		Demande-Réponse
Nom du message	Description	
SetAudioOutputConfiguration-Request	<i>L'élément Configuration contient la configuration AudioOutput modifiée. L'élément Configuration contient un élément qui spécifie la sortie audio dont la configuration est à modifier. La sortie audio doit exister dans le dispositif.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AudioOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioOutputConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAudioOutput	La sortie audio demandée n'existe pas	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis	

9.6.3 GetAudioOutputConfigurationOptions

Cette opération demande les AudioOutputConfigurationOptions d'un AudioOutput. Un dispositif comportant un ou plusieurs AudioOutputs doit prendre en charge cette commande (voir Tableau 102).

Tableau 102 – Commande GetAudioOutputConfigurationOptions

GetAudioOutputConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetAudioOutputConfiguration-OptionsRequest	<i>L'élément AudioOutputToken spécifie la sortie vidéo dont les options sont demandées. La sortie audio doit exister dans le dispositif</i> tt:ReferenceToken AudioOutputToken[1][1]	
GetAudioOutputConfiguration-OptionsResponse	<i>L'élément AudioOutputsOptions retourne les plages de valeurs valides pour SendPrimacy et OutputLevel, ainsi que l'AudioOutputToken disponible. La sortie dont les options sont demandées doit être indiquée dans ce champ.</i> tt:AudioOutputConfigurationOptions AudioOutputOptions[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAudioOutput	<i>La sortie audio demandée n'existe pas.</i>	

9.7 AudioSources

9.7.1 Généralités

Une AudioSource représente une entrée audio non codée et déclare le nombre de canaux d'entrée.

9.7.2 GetAudioSources

Cette opération permet de répertorier toutes les sources audio disponibles pour le dispositif. Un dispositif comportant une ou plusieurs sources audio doit prendre en charge la liste des entrées audio disponibles grâce à la commande GetAudioSources (voir Tableau 103).

Tableau 103 – Commande GetAudioSources

GetAudioSources		Demande-Réponse
Nom du message	Description	
GetAudioSourcesRequest	<i>Ceci est un message vide.</i>	
GetAudioSourcesResponse	<i>Contient une liste de structures décrivant toutes les sources audio disponibles du dispositif. Si un dispositif ne contient aucune entrée audio, une liste vide est retournée</i> tt:AudioSource AudioSource [0][non limité]	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Le NVT ne prend pas en charge l'audio.</i>	

9.8 AudioSourceConfiguration

Un élément AudioSourceConfiguration contient une référence à une source audio.

9.8.1 GetAudioSourceConfiguration

Cette opération permet de répertorier la configuration d'une entrée audio. Un dispositif comportant une ou plusieurs entrées audio doit prendre en charge la commande GetAudioSourceConfiguration (voir Tableau 104).

Tableau 104 – Commande GetAudioSourceConfiguration

GetAudioSourceConfiguration		Demande-Réponse
Nom du message	Description	
GetAudioSourceConfigurationRequest	<i>Ce message contient le jeton d'AudioSource.</i> tt:ReferenceToken AudioSourceToken[1][1]	
GetAudioSourceConfigurationResponse	<i>Ce message contient l'AudioSourceConfiguration demandé avec le jeton correspondant.</i> tt:AudioSourceConfiguration AudioSourceConfiguration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAudioSource	<i>L'AudioSource demandée indiquée avec AudioSourceToken n'existe pas.</i>	

9.8.2 SetAudioSourceConfiguration

Cette opération modifie une configuration de source audio. Un dispositif comportant une ou plusieurs sources audio doit prendre en charge les paramètres d'AudioSourceConfiguration grâce à cette commande (voir Tableau 105).

Tableau 105 – Commande SetAudioSourceConfiguration

SetAudioSourceConfiguration		Demande-Réponse
Nom du message	Description	
SetAudioSourceConfiguration-Request	<i>L'élément Configuration contient la configuration AudioSource modifiée. L'élément Configuration contient un élément qui spécifie l'AudioSource dont la configuration est à modifier. L'entrée audio doit exister dans le dispositif.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AudioSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioSourceConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAudioSource	L'AudioSource demandée n'existe pas	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	

9.8.3 GetAudioSourceConfigurationOptions

Cette opération demande les AudioSourceConfigurationOptions d'une AudioSource. Un dispositif comportant une ou plusieurs sources audio doit prendre en charge cette commande (voir Tableau 106).

Tableau 106 – Commande GetAudioSourceConfigurationOptions

GetAudioSourceConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetAudioSourceConfigurationOptions-Request	<i>L'élément AudioSourceToken spécifie l'entrée audio dont les options sont demandées. L'AudioSource doit exister dans le dispositif.</i> tt:ReferenceToken AudioSourceToken[1][1]	
GetAudioSourceConfiguration-Response	<i>Les AudioSourcesOptions retournent l'AudioSourceToken disponible. La Source dont les options sont demandées doit être indiquée dans ce champ.</i> tt:AudioSourceConfigurationOptions AudioSourceOptions[1][1]	
Codes de défaut		Description
env:Sender		L'entrée audio demandée n'existe pas
ter:InvalidArgVal		
ter:NoAudioSource		

9.9 Sorties relais

Les commandes d'entrée/sortie (E/S) sont utilisées pour contrôler l'état ou observer le statut des ports E/S. Si le dispositif comporte des ports E/S, il doit prendre en charge les commandes E/S.

Les sorties relais sont également définies dans DeviceManagement (voir Entrée/Sortie (E/S)). Les sorties relais peuvent accéder aux services DeviceManagement et DeviceIO.

9.9.1 Obtention de sorties de relais

Cette opération permet d'extraire une liste de l'ensemble des sorties de relais disponibles et leurs paramètres (voir Tableau 107).

Tableau 107 – Commande GetRelayOutputs

GetRelayOutputs		Demande-Réponse
Nom du message	Description	
GetRelayOutputsRequest	<i>Ceci est un message vide.</i>	
GetRelayOutputsResponse	<i>Ce message contient un ensemble de sorties de relais.</i> tt:RelayOutput RelayOutputs [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

9.9.2 Définition de réglages de sortie de relais

Cette opération définit les paramètres d'une sortie de relais.

Le relais peut fonctionner dans deux modes de relais:

- Bistable – après la définition de l'état, le relais reste dans cet état;
- Monostable – après la définition de l'état, le relais retourne à son état de veille après le temps spécifié.

L'état de veille physique d'une sortie de relais peut être configuré en attribuant la valeur "open" (ouvert) ou "closed" (fermé) à IdleState (inversion du comportement de relais).

L'état de veille "open" signifie que le relais est ouvert lorsque l'état de relais est "inactive" (inactif) par l'intermédiaire de la commande de déclenchement (voir 0) et fermé lorsque l'état de relais est "active" (actif) à l'aide de la même commande.

L'état de veille "closed" signifie que le relais est fermé lorsque l'état du relais est "inactive" par l'intermédiaire de la commande de déclenchement (voir 0) et ouvert lorsque l'état est "active" à l'aide de la même commande (voir Tableau 108).

Tableau 108 – Commande SetRelayOutputSettings

SetRelayOutputSettings		Demande-Réponse
Nom du message	Description	
SetRelayOutputSettings Request	<i>Ce message contient:</i> <i>"RelayOutputToken": Référence de jeton vers la sortie de relais demandée.</i> <i>"RelayOutputSettings": Paramètres du relais</i> tt:ReferenceToken RelayOutputToken [1][1] tt:RelayOutputSettings RelayOutputSettings [1][1]	
SetRelayOutputSettings Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Référence de jeton de relais inconnue.</i>	
env:Sender ter:InvalidArgVal ter:ModeError	<i>Délai monostable non valide</i>	

9.9.3 Déclenchement de sortie de relais

Cette opération déclenche une sortie de relais² (voir Tableau 109).

² Il n'existe pas de commande GetRelayState. L'état logique réel de la sortie de relais est transmis par notification et leurs propriétés.

Tableau 109 – Commande SetRelayOutputState

SetRelayOutputState		Demande-Réponse
Nom du message	Description	
SetRelayOutputStateRequest	<i>Ce message contient:</i> <i>"RelayOutputToken": Référence de jeton vers la sortie de relais demandée.</i> <i>"LogicalState": Demande de déclenchement, c'est-à-dire, actif ou inactif.</i> tt:ReferenceToken RelayOutputToken [1][1] tt:RelayLogicalState LogicalState [1][1]	
SetRelayOutputStateResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:RelayToken	<i>Référence de jeton de relais inconnue.</i>	

9.10 Codes de défaut spécifiques au service

Le Tableau 110 présente les codes de défaut spécifiques au service DeviceIO. De plus, chaque commande peut également générer un défaut générique (voir Tableau 6).

Tableau 110 – Codes de défaut spécifiques au service DeviceIO

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Sender	ter:InvalidArgVal	Paramètres de configuration non valides	Les paramètres de configuration ne peuvent pas être définis.
	ter:ConfigModify		
env:Sender	ter:InvalidArgVal	Le jeton de sortie vidéo n'existe pas.	Le VideoOutput demandé indiqué avec VideoOutputToken n'existe pas.
	ter:NoVideoOutput		
env:Sender	ter:InvalidArgVal	Le jeton de source vidéo n'existe pas.	Le VideoSource demandé indiqué avec VideoSourceToken n'existe pas.
	ter:NoVideoSource		
env:Sender	ter:InvalidArgVal	Le jeton de sortie audio n'existe pas.	L'élément AudioOutput demandé indiqué avec AudioOutputToken n'existe pas.
	ter:NoAudioOutput		
env:Sender	ter:InvalidArgVal	Le jeton de source audio n'existe pas.	L'AudioSource demandée indiquée avec AudioSourceToken n'existe pas.
	ter:NoAudioSource		
env:Sender	ter:InvalidArgVal	Référence de jeton de relais inconnue	Le RelayOutput demandé indiqué avec RelayOutputToken n'existe pas.
	ter:RelayToken		
env:Sender	ter:InvalidArgVal	Délai monostable non valide.	
	ter:ModeError		

10 Configuration d'imagerie

Le service d'imagerie comporte des opérations utilisées pour contrôler et configurer les propriétés d'imagerie sur un dispositif. Il convient qu'un dispositif comportant une ou plusieurs sources vidéo prenne en charge le service d'imagerie défini en C.7. Les paramètres d'imagerie font partie de l'entité VideoSource. Cela signifie que les paramètres d'imagerie affectent directement une source vidéo spécifique.

10.1 Paramètres d'imagerie

Le service d'imagerie comprend des opérations d'extraction ou de définition des paramètres d'imagerie et des plages valides pour ces paramètres. Certains paramètres n'ont aucun effet si un mode spécifique n'est pas défini. Certains des paramètres requièrent une fonctionnalité d'imagerie spécifique qui peut être demandée par l'intermédiaire de la commande GetOptions. Les paramètres suivants sont disponibles par l'intermédiaire des opérations de service d'imagerie:

BacklightCompensation: Active/désactive le mode BLC (On/Off)

- On
 - Paramètre de niveau optionnel (unité non spécifiée).
- Off

Brightness: Ajuste la luminosité de l'image (unité non spécifiée).

ColorSaturation: Ajuste la saturation de couleur de l'image (unité non spécifiée).

Sharpness: Ajuste la netteté dans l'image (unité non spécifiée).

Contrast: Ajuste le contraste de l'image (unité non spécifiée).

Exposure:

- Auto – Active l'algorithme d'exposition sur le dispositif:
 - Priority – Définit le mode de priorité d'exposition (faible bruit/taux de trame).
 - Window – Masque d'exposition rectangulaire.
 - Min/MaxExposureTime – Plage de temps d'exposition pouvant être utilisée par l'algorithme.
 - Min/MaxGain – Plage de gain de capteur pouvant être utilisée par l'algorithme.
 - Min/MaxIris – Plage de diaphragme pouvant être utilisée par l'algorithme.
- Manual – Désactive l'algorithme d'exposition sur le dispositif:
 - ExposureTime – Temps d'exposition fixe utilisé par le capteur d'image (μ s).
 - Gain – Gain fixe utilisé par le capteur d'image (dB).
 - Iris – Atténuation fixe de la lumière d'entrée affectée par le diaphragme (dB). 0 dB correspond à un diaphragme totalement ouvert.

Focus:

- Auto (paramètres qui s'appliquent au mode automatique uniquement):
 - Near/FarLimit – Limites pour la lentille de mise au point (m).
- Manuel (paramètres qui s'appliquent au mode manuel uniquement):
 - Default speed – Vitesse par défaut pour l'opération de déplacement de mise au point (en l'absence du paramètre de vitesse). Une commande manuelle est effectuée par l'intermédiaire de la commande de mouvement (voir 0).

Ir cut filter: Bascule l'état de filtre de coupure Ir entre On (actif), Off (inactif) et Auto. L'état Auto laisse l'algorithme d'exposition décider s'il convient d'activer/désactiver le filtre de coupure Ir.

Whitebalance:

- Mode de balance des blancs automatique (auto/manuel).
- Manuel (paramètres qui s'appliquent au mode manuel uniquement):
 - Rgain (sans unité).
 - Bgain (sans unité).

WideDynamicRange: Gamme dynamique étendue (On (actif)/Off (inactif)).

- On
 - Paramètre de niveau optionnel (sans unité).
- Off

Les paramètres d'imagerie disponibles peuvent être extraits grâce à la commande GetVideoSources du service multimédia, comme spécifié en 11.3.2. Les paramètres d'imagerie font partie de la source vidéo.

10.1.1 Obtention de paramètres d'imagerie

Cette opération permet de demander les paramètres d'imagerie d'une source vidéo sur le dispositif. Si la Source vidéo prend en charge l'un des paramètres d'imagerie définis par le type ImagingSettings dans [ONVIF Schema], il convient que les paramètres d'imagerie du dispositif puissent être extraits grâce à la commande GetImagingSettings (voir Tableau 111).

Les paramètres d'imagerie sont décrits en 10.1.

Tableau 111 – Commande GetImagingSettings

GetImagingSettings		Demande-Réponse	
Nom du message		Description	
GetImagingSettingsRequest		<i>Ce message contient une référence à la VideoSource (source vidéo) pour laquelle il convient que les ImagingSettings (réglages d'imagerie) soient demandés.</i> tt:ReferenceToken VideoSourceToken[1][1]	
GetImagingSettingsResponse		<i>Ce message contient les ImagingSettings (réglages d'imagerie) pour la VideoSource qui est demandée.</i> tt:ImagingSettings20ImagingSettings[1][1]	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoSource		<i>La VideoSource demandée n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource		<i>La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.</i>	

10.1.2 Définition des paramètres d'imagerie

Cette opération définit les paramètres d'imagerie d'une source vidéo sur un dispositif. Si le dispositif prend en charge l'un des paramètres d'imagerie définis par le type ImagingSettings dans [ONVIF Schema], il convient que les paramètres d'imagerie du dispositif puissent être configurés grâce à la commande SetImagingSettings (voir Tableau 112).

Les paramètres d'imagerie configurables possibles sont décrits en 10.1. Des options de réglage sont obtenues grâce à la commande définie en 0.

Tableau 112 – Commande SetImagingSettings

SetImagingSettings		Demande-Réponse	
Nom du message		Description	
SetImagingSettingsRequest		<i>Ce message contient une référence à la VideoSource (source vidéo) et aux ImagingSettings (réglages d'imagerie) qu'il convient de définir.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:ReferenceToken VideoSourceToken[1][1] tt:ImagingSettings20ImagingSettings[1][1] xs:boolean ForcePersistence [0][1]	
SetImagingSettingsResponse		Ce message ne contient pas de réponse.	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoSource		La VideoSource demandée n'existe pas.	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource		La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.	
env:Sender ter:InvalidArgVal ter:SettingsInvalid		Les réglages demandés sont incorrects.	

10.1.3 Obtention d'options

Cette opération permet d'extraire les plages valides pour les paramètres d'imagerie comportant des plages spécifiques au dispositif. Si le dispositif prend en charge la commande SetImagingSettings pour définir le paramètre d'imagerie, il doit obtenir les options de configuration à partir du dispositif grâce à la commande GetOptions (voir Tableau 113).

Tableau 113 – Commande GetOptions

GetOptions		Demande-Réponse	
Nom du message		Description	
GetOptionsRequest		<i>Référence à la VideoSource (source vidéo) pour laquelle les options de paramètre d'imagerie sont demandées.</i> tt:ReferenceToken VideoSourceToken[1][1]	
GetOptionsResponse		<i>Ce message contient les plages valides pour les paramètres d'imagerie qui sont classés comme étant spécifiques au dispositif.</i> tt:ImagingOptions20 ImagingOptions[1][1]	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoSource		<i>La VideoSource demandée n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource		<i>La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.</i>	

10.1.4 Move

La commande Move permet de déplacer la lentille de mise au point de manière absolue, relative ou continue depuis sa position actuelle (voir Tableau 114). L'argument de vitesse est facultatif pour les commandes absolue et relative, mais requis pour le mode continu. Si aucun argument de vitesse n'est utilisé, la vitesse par défaut est utilisée. Les ajustements de mise au point par l'intermédiaire de cette opération désactivent la mise au point automatique. Il convient qu'un dispositif avec prise en charge de la commande de mise au point à distance prenne en charge la commande absolue, relative ou continue grâce à l'opération de déplacement.

Les fonctionnalités d'imagerie spécifient les opérations de mise au point spécifiques prises en charge par cette opération. Au moins une fonctionnalité de commande de mise au point est requise pour que cette opération soit fonctionnelle.

L'opération de déplacement contient les commandes suivantes:

Absolute – Requier le paramètre de position et accepte éventuellement un argument de vitesse. Un type sans unité est utilisé par défaut pour le positionnement et la vitesse de mise au point. Éventuellement, la position peut être demandée en unités m^{-1} , si elle est prise en charge.

Relative – Requier le paramètre de distance et accepte éventuellement un argument de vitesse. Une distance négative désigne une direction négative.

Continuous – Requier un argument de vitesse. Un argument de vitesse négative signifie une direction négative.

Tableau 114 – Commande Move (mise au point)

Move		Demande-Réponse	
Nom du message		Description	
MoveRequest		<i>Référence à la VideoSource (source vidéo) pour l'opération de déplacement (de mise au point) demandée.</i> tt:ReferenceToken VideoSourceToken[1][1] tt:FocusMove Focus[1][1]	
MoveResponse		<i>Ce message est vide.</i>	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoSource		<i>La VideoSource demandée n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource		<i>La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.</i>	

10.1.5 Obtention d'options de déplacement

La commande GetMoveOptions permet d'extraire les options de déplacement de lentille de mise au point à utiliser dans la commande de déplacement (voir 0). Un dispositif qui prend en charge l'opération de déplacement de lentille doit également prendre en charge la commande GetMoveOptions (voir Tableau 115).

Tableau 115 – Commande GetMoveOptions (mise au point)

GetMoveOptions		Demande-Réponse	
Nom du message		Description	
GetMoveOptionsRequest		<i>Référence à la VideoSource (source vidéo) pour les options de déplacement demandées.</i> tt:ReferenceToken VideoSourceToken [1][1]	
GetMoveOptionsResponse		<i>Ce message contient les plages valides pour les options de déplacement de lentille de mise au point.</i> tt:MoveOptions20 MoveOptions [1][1]	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoSource		<i>La VideoSource demandée n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource		<i>La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.</i>	

10.1.6 Stop

La commande Stop arrête tous les déplacements de mise au point en cours de la lentille (voir Tableau 116). Si le dispositif prend en charge la mise au point, il convient que la mise au point puisse être arrêtée grâce à l'opération d'arrêt. Cette opération n'affecte pas une mise au point automatique en cours.

Tableau 116 – Commande Stop (mise au point)

Stop		Demande-Réponse	
Nom du message		Description	
StopRequest		<i>Référence à la VideoSource (source vidéo) sur laquelle il convient d'arrêter le déplacement de mise au point.</i> tt:ReferenceToken VideoSourceToken[1][1]	
StopResponse		Ce message est vide.	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoSource		La VideoSource demandée n'existe pas.	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource		La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.	

10.1.7 Obtention du statut d'imagerie

La commande GetStatus demande le statut d'imagerie actuel du dispositif (voir Tableau 117). Si le dispositif prend en charge la commande de déplacement de mise au point, il convient que le statut d'imagerie disponible puisse être obtenu grâce à la commande GetStatus.

Le statut d'imagerie contient:

- Position de mise au point, statut de déplacement et informations d'erreur.
 - La position de mise au point est représentée dans un type sans unité.
 - Le statut de déplacement peut être dans un état MOVING (Mobile), IDLE (Veille) ou UNKNOWN (Inconnu).
 - Les informations d'erreur présentées sous la forme d'une chaîne (une erreur de positionnement indiquée par le matériel, par exemple).

Tableau 117 – Commande GetStatus (mise au point)

GetStatus		Demande-Réponse	
Nom du message		Description	
GetStatusRequest		Ce message contient une référence à la VideoSource (source vidéo) au niveau de laquelle il convient de demander le statut d'imagerie. tt:VideoSourceToken VideoSourceToken[1][1]	
GetStatusResponse		Ce message contient le statut d'imagerie demandé. tt:ImagingStatus20 ImagingStatus[1][1]	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoSource		La VideoSource demandée n'existe pas.	
env:Receiver ter:ActionNotSupported ter:NoImagingForSource		La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.	

10.2 Codes de défaut spécifiques au service

Le Tableau 118 présente les codes de défaut spécifiques au service d'imagerie. De plus, chaque commande peut également générer un défaut générique (voir Tableau 6).

Les défauts spécifiques sont définis en tant que sous-code d'un défaut générique (voir article 5.11.2.1). Le sous-code générique parent est le sous-code en haut de chaque ligne ci-dessous, le sous-code de défaut spécifique se trouvant en bas de la cellule.

Tableau 118 – Codes de défaut spécifiques d'imagerie

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Receiver	ter:ActionNotSupported	La VideoSource ne prend pas en charge les paramètres d'imagerie	La VideoSource demandée ne prend pas en charge les paramètres d'imagerie.
	ter:NoImagingForSource		
env:Sender	ter:InvalidArgVal	Configuration non valide	Les réglages demandés sont incorrects.
	ter:SettingsInvalid		
env:Sender	ter:InvalidArgVal	La source vidéo n'existe pas	La VideoSource demandée n'existe pas.
	ter:NoSource		

11 Configuration multimédia

Le service multimédia est utilisé pour configurer les propriétés de transmission multimédia du dispositif. Le NVT doit prendre en charge le service multimédia spécifié en C.8.

Le service multimédia permet à un client de procéder à des configurations de transmission continue de média et toute autre transmission continue en temps réel. Les configurations multimédia sont gérées par l'intermédiaire de profils multimédia. Une vue d'ensemble du modèle de configuration multimédia de l'ONVIF est présentée en 4.8.

Les commandes de service multimédia sont divisées en deux catégories principales:

- Configuration multimédia:
 - Commandes de profil multimédia
 - Commandes de source vidéo
 - Commandes de codeur vidéo
 - Commandes de source audio
 - Commandes de codeur audio
 - Commandes d'analyse vidéo
 - Commandes de métadonnées
 - Commandes de sortie audio
 - Commandes de décodeur audio
- Transmission multimédia:
 - Demande d'URI de flux
 - Obtention d'URI d'instantané
 - Commandes de contrôle de multidiffusion
 - Point de synchronisation multimédia

Un ensemble basique d'opérations est requis pour le service multimédia. Il est recommandé de prendre en charge d'autres opérations. Les exigences détaillées sont présentées sous les descriptions de commande.

11.1 Codecs audio et vidéo

Le NVT transmet en continu des données audio et vidéo en utilisant des algorithmes de codage adaptés. Il peut également être en mesure de décoder les éléments audio. Le NVT prend en charge des codecs audio et vidéo, des débits binaires et une résolution en fonction des choix du fabricant. Afin d'assurer l'interopérabilité entre le NVT et le client, la présente Norme spécifie les profils de codec suivants:

- Le NVT doit prendre en charge JPEG QVGA.
- Le NVT doit prendre en charge le format G.711 μ Law (Simplex-Camera Microphone Only, 1 voie) [UIT-T G.711] s'il prend en charge l'audio.

11.2 Profil multimédia

Un profil multimédia est composé d'un ensemble de configurations multimédia. Les profils multimédia sont utilisés par un client pour configurer les propriétés d'un flux multimédia provenant d'un NVT.

Un NVT doit fournir au moins un profil multimédia au démarrage. Il convient qu'un NVT fournisse des profils "prêts à l'emploi" pour les configurations multimédia les plus courantes produites par le dispositif.

Un profil est composé d'un ensemble d'*entités de configuration* interconnectées. Les configurations sont fournies par le NVT et peuvent être statiques ou créées de manière dynamique par le NVT. Par exemple, les configurations dynamiques peuvent être créées par le NVT en fonction des ressources de codage disponibles actuelles. Une entité de configuration est d'un des types suivants:

- Configuration de source vidéo
- Configuration de source audio
- Configuration de codeur vidéo
- Configuration de codeur audio
- Configuration PTZ
- Configuration d'analyse vidéo
- Configuration de métadonnées
- Configuration de sortie audio
- Configuration de décodeur audio

Un profil est composé de l'ensemble ou d'un sous-ensemble de ces entités de configuration. Suivant les fonctionnalités du NVT, une entité de configuration particulière peut faire partie ou non d'un profil. Par exemple, un profil avec une source audio et une configuration de codeur audio peut exister uniquement dans un dispositif avec support audio.

11.2.1 Création de profil multimédia

Cette opération permet de créer un profil multimédia vide (voir Tableau 119). Le profil multimédia doit être créé dans le NVT et doit être persistant (rester après un redémarrage). Le NVT doit prendre en charge la création de profils multimédia comme défini dans la présente Norme grâce à la commande *CreateProfile*. Le nombre maximal de profils qu'un dispositif prend en charge est retourné dans les fonctionnalités multimédia.

Un profil créé doit pouvoir être supprimé et la valeur "false" doit être attribuée à l'attribut "fixed" du NVT dans le profil retourné.

Tableau 119 – Commande CreateProfile

CreateProfile		Demande-Réponse
Nom du message	Description	
CreateProfileRequest	<i>Contient le paramètre Name du profil à créer et un paramètre facultatif Token, spécifiant l'identifiant unique du nouveau profil multimédia</i> tt:Name Name [1][1] tt:ReferenceToken Token [0][1]	
CreateProfileResponse	<i>Retourne une structure de profil vide sans entité de configuration.</i> tt:Profile Profile [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:ProfileExists	<i>Un profil avec le jeton ProfileToken existe déjà.</i>	
env:Receiver ter:Action ter:MaxNVTProfiles	<i>Le nombre maximal de profils pris en charge a été atteint.</i>	

11.2.2 Obtention de profils multimédia

Un point terminal peut demander les profils multimédia *existants* d'un NVT à l'aide de la commande GetProfiles (voir Tableau 120). Les profils préconfigurés ou configurés de manière dynamique peuvent être extraits en utilisant cette commande. Cette commande permet de répertorier *tous* les profils configurés dans un dispositif. Il n'est pas nécessaire que le client connaisse le profil multimédia pour utiliser la commande. Le NVT doit prendre en charge l'extraction de profils multimédia grâce à la commande GetProfiles.

Un NVT doit contenir l'attribut "fixed" dans tous les éléments Profile retournés.

Tableau 120 – Commande GetProfiles

GetProfiles		Demande-Réponse
Nom du message	Description	
GetProfilesRequest	<i>Ceci est un message vide.</i>	
GetProfilesResponse	La réponse contient une liste de profils. Chaque profil contient un ensemble d'entités de configuration définissant une configuration spécifique qui peut être utilisée pour la transmission multimédia, l'analyse, la transmission de métadonnées, etc. tt:Profile Profiles [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

11.2.3 Obtention de profil multimédia

Si le jeton de profil est déjà connu, un profil peut être extrait grâce à la commande GetProfile (voir Tableau 121). Le NVT doit prendre en charge l'extraction d'un profil multimédia spécifique par l'intermédiaire de la commande GetProfile.

Un NVT doit contenir l'attribut "fixed" dans l'élément Profile retourné.

Tableau 121 – Commande GetProfile

GetProfile		Demande-Réponse
Nom du message	Description	
GetProfileRequest	Ce message contient le jeton pour le profil demandé. tt:ReferenceToken ProfileToken [1][1]	
GetProfileResponse	La réponse contient le Profile indiqué par le paramètre Token. Un profil contient un ensemble d'entités de configuration définissant une configuration spécifique qui peut être utilisée pour la transmission multimédia, l'analyse, la transmission de métadonnées, etc. tt:Profile Profile [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	

11.2.4 Ajout de configuration de source vidéo à un profil

Cette opération ajoute un VideoSourceConfiguration à un profil multimédia existant (voir Tableau 122). Si une telle configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Le NVT doit prendre en charge l'ajout d'une configuration de source vidéo à un profil grâce à la commande AddVideoSourceConfiguration.

Tableau 122 – Commande AddVideoSourceConfiguration

AddVideoSourceConfiguration		Demande-Réponse
Nom du message	Description	
AddVideoSourceConfiguration Request	Contient une référence au VideoSourceConfiguration à ajouter et au profil auquel il doit être ajouté. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddVideoSourceConfiguration Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Le VideoSourceConfiguration indiqué par le ConfigurationToken n'existe pas.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.</i>	

11.2.5 Ajout de configuration de codeur vidéo à un profil

Cette opération ajoute un VideoEncoderConfiguration à un profil multimédia existant (voir Tableau 123). Si une configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Un NVT doit prendre en charge l'ajout d'une configuration de codeur vidéo à un profil grâce à la commande AddVideoEncoderConfiguration.

L'ajout d'une VideoEncoderConfiguration à un profil signifie qu'un flux utilisant ce profil contient des données vidéo. Il convient d'ajouter des configurations de codeur vidéo après l'ajout d'une configuration de source vidéo.

Tableau 123 – Commande AddVideoEncoderConfiguration

AddVideoEncoderConfiguration		Demande-Réponse
Nom du message	Description	
AddVideoEncoderConfiguration Request	Contient une référence au VideoEncoderConfiguration à ajouter et au profil auquel il doit être ajouté. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddVideoEncoderConfiguration Response	Ceci est un message vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Sender ter:InvalidArgs ter:NoConfig	Le VideoEncoderConfiguration indiqué par le ConfigurationToken n'existe pas.	
env:Receiver ter:Action ter:ConfigurationConflict	D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.	

11.2.6 Ajout de configuration de source audio à un profil

Cette opération permet d'ajouter un AudioSourceConfiguration à un profil multimédia existant (voir Tableau 124). Si une configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge l'ajout de la configuration de source audio à un profil grâce à la commande AddAudioSourceConfiguration.

Tableau 124 – Commande AddAudioSourceConfiguration

AddAudioSourceConfiguration		Demande-Réponse
Nom du message	Description	
AddAudioSourceConfiguration Request	Contient une référence à l'AudioSourceConfiguration à ajouter et au profil auquel il doit être ajouté. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioSourceConfiguration Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>L'AudioSourceConfiguration indiqué par le ConfigurationToken n'existe pas.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>L'audio n'est pas pris en charge.</i>	

11.2.7 Ajout de configuration de codeur audio à un profil

Cette opération permet d'ajouter un AudioEncoderConfiguration à un profil multimédia existant (voir Tableau 125). Si une configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge l'ajout de configurations de codeur audio à un profil grâce à la commande AddAudioEncoderConfiguration.

L'ajout d'un AudioEncoderConfiguration à un profil multimédia signifie que les flux utilisant ce profil multimédia contiennent des données audio. Il convient d'ajouter des configurations de codeur audio après avoir ajouté une configuration de source audio.

Tableau 125 – Commande AddAudioEncoderConfiguration

AddAudioEncoderConfiguration		Demande-Réponse
Nom du message	Description	
AddAudioEncoderConfiguration Request	Contient une référence à l'AudioEncoderConfiguration à ajouter et au profil auquel il doit être ajouté. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioEncoderConfiguration Response	Ceci est un message vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoConfig	L'AudioEncoderConfiguration indiqué par le ConfigurationToken n'existe pas.	
env:Receiver ter:Action ter:ConfigurationConflict	D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	L'audio n'est pas pris en charge.	

11.2.8 Ajout de configuration PTZ à un profil

Cette opération permet d'ajouter un PTZConfiguration à un profil multimédia existant (voir Tableau 126). Si une configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Un NVT qui prend en charge le contrôle de PTZ doit prendre en charge l'ajout de configurations PTZ à un profil grâce à la commande AddPTZConfiguration.

L'ajout d'un PTZConfiguration à un profil multimédia signifie que des flux utilisant ce profil multimédia peuvent contenir le statut PTZ (dans les métadonnées) et que le profil multimédia peut être utilisé pour contrôler le déplacement PTZ (voir Article 16).

Tableau 126 – Commande AddPTZConfiguration

AddPTZConfiguration		Demande-Réponse
Nom du message	Description	
AddPTZConfigurationRequest	Contient une référence au PTZConfiguration à ajouter et le profil auquel il doit être ajouté. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddPTZConfigurationResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Le PTZConfiguration indiqué par le ConfigurationToken n'existe pas.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

11.2.9 Ajout de configuration d'analyse vidéo à un profil

Cette opération permet d'ajouter une configuration VideoAnalytics à un profil multimédia existant (voir Tableau 127). Si une configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Un NVT qui prend en charge l'analyse vidéo doit prendre en charge l'ajout de configurations d'analyse vidéo à un profil grâce à la commande AddVideoAnalyticsConfiguration.

L'ajout d'un VideoAnalyticsConfiguration à un profil multimédia signifie que des flux utilisant ce profil multimédia peuvent contenir des données d'analyse vidéo (dans les métadonnées) telles que définies par la référence de configuration soumise. Les données d'analyse vidéo sont spécifiées en 17.1 et les configurations d'analyse sont gérées grâce aux commandes définies en 0.

Un profil contenant uniquement une configuration d'analyse vidéo mais pas de configuration de source vidéo est incomplet. Par conséquent, il convient qu'un client ajoute dans un premier temps une configuration de source vidéo à un profil avant d'ajouter une configuration d'analyse vidéo. Le NVT peut refuser l'ajout d'une configuration d'analyse vidéo avant une configuration de source vidéo. Dans ce cas, il convient qu'il réponde avec un défaut ConfigurationConflict.

Tableau 127 – Commande AddVideoAnalytics

AddVideoAnalytics		Demande-Réponse
Nom du message	Description	
AddVideoAnalyticsRequest	Contient une référence à l'élément VideoAnalytics à ajouter et au profil auquel il doit être ajouté tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddVideoAnalyticsResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Le VideoAnalytics indiqué par le ConfigurationToken n'existe pas.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:VideoAnalyticsNotSupported	<i>La VideoAnalytics (analyse vidéo) n'est pas prise en charge.</i>	

11.2.10 Ajout de configuration de métadonnées à un profil

Cette opération permet d'ajouter une configuration de métadonnées à un profil multimédia existant (voir Tableau 128). Si une configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Un NVT doit prendre en charge l'ajout d'une configuration de métadonnées à un profil grâce à la commande AddMetadataConfiguration.

L'ajout d'un MetadataConfiguration à un profil signifie que des flux utilisant ce profil contiennent des métadonnées. Les métadonnées peuvent être composées d'événements, du statut PTZ et/ou de données d'analyse vidéo. Les configurations de métadonnées sont gérées grâce aux commandes définies en 11.10 et 11.9.4.

Tableau 128 – Commande AddMetadataConfiguration

AddMetadataConfiguration		Demande-Réponse
Nom du message	Description	
AddMetadataConfiguration Request	Contient une référence au MetadataConfiguration à ajouter et le profil auquel il doit être ajouté. tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddMetadataConfiguration Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Le MetadataConfiguration indiqué par le ConfigurationToken n'existe pas.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.</i>	

11.2.11 Ajout de configuration de sortie audio

Cette opération permet d'ajouter un AudioOutputConfiguration à un profil multimédia existant (voir Tableau 129). Si une configuration existe dans le profil multimédia, elle est remplacée. Le changement doit être persistant. Un NVT qui comporte une sortie audio doit prendre en charge l'ajout d'une configuration de sortie audio à un profil grâce à la commande AddAudioOutputConfiguration.

Tableau 129 – AddAudioOutputConfiguration

AddAudioOutputConfiguration		Demande-Réponse
Nom du message	Description	
AddAudioOutputConfiguration Request	<i>Contient une référence à l'AudioOutputConfiguration à ajouter et le profil auquel il doit être ajouté.</i> tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioOutputConfiguration Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgs ter:NoConfig	<i>L'AudioOutputConfiguration indiqué par le ConfigurationToken n'existe pas.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>L'Audio ou la sortie audio n'est pas pris en charge</i>	

11.2.12 Ajout de configuration de décodeur audio

Cette opération permet d'ajouter un AudioDecoderConfiguration à un profil multimédia existant (voir Tableau 130). Si une configuration existe dans le profil multimédia, elle doit être remplacée. Le changement doit être persistant. Un NVT qui comporte des fonctionnalités de décodage audio doit prendre en charge l'ajout d'une configuration de décodeur audio à un profil grâce à la commande AddAudioDecoderConfiguration.

Tableau 130 – AddAudioDecoderConfiguration

AddAudioDecoderConfiguration		Demande-Réponse
Nom du message	Description	
AddAudioDecoderConfiguration Request	<i>Contient une référence à l'AudioConfiguration à ajouter et le profil auquel il doit être ajouté.</i> tt:ReferenceToken ProfileToken [1][1] tt:ReferenceToken ConfigurationToken [1][1]	
AddAudioDecoderConfiguration Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgs ter:NoConfig	<i>L'AudioDecoderConfiguration indiqué par le ConfigurationToken n'existe pas.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont en conflit avec celle à ajouter et son ajout génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	<i>L'Audio ou le décodage audio n'est pas pris en charge</i>	

11.2.13 Suppression de configuration de source vidéo d'un profil

Cette opération permet de supprimer une VideoSourceConfiguration d'un profil multimédia existant (voir Tableau 131). Si le profil multimédia ne contient pas de VideoSourceConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Le NVT doit prendre en charge la suppression d'une configuration de source vidéo d'un profil grâce à la commande RemoveVideoSourceConfiguration.

Il convient que les configurations de source vidéo soient supprimées uniquement après la suppression d'une VideoEncoderConfiguration du profil multimédia.

Tableau 131 – Commande RemoveVideoSourceConfiguration

RemoveVideoSourceConfiguration		Demande-Réponse
Nom du message	Description	
RemoveVideoSourceConfiguration-Request	Contient une référence au profil multimédia duquel la VideoSourceConfiguration doit être supprimée. tt:ReferenceToken ProfileToken [1][1]	
RemoveVideoSourceConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe aucune configuration de source vidéo dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de la VideoSourceConfiguration et sa suppression génère un conflit de profil multimédia.</i>	

11.2.14 Suppression de configuration de codeur vidéo d'un profil

Cette opération permet de supprimer un VideoEncoderConfiguration d'un profil multimédia existant (voir Tableau 132). Si le profil multimédia ne contient pas de VideoEncoderConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Le NVT doit prendre en charge la suppression d'une configuration de codeur vidéo d'un profil grâce à la commande RemoveVideoEncoderConfiguration.

Tableau 132 – Commande RemoveVideoEncoderConfiguration

RemoveVideoEncoderConfiguration		Demande-Réponse
Nom du message	Description	
RemoveVideoEncoderConfiguration-Request	Contient une référence au profil multimédia duquel la VideoEncoderConfiguration doit être supprimée. tt:ReferenceToken ProfileToken [1][1]	
RemoveVideoEncoderConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration de codeur vidéo dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de la VideoEncoderConfiguration et sa suppression génère un conflit de profil multimédia.</i>	

11.2.15 Suppression de configuration de source audio d'un profil

Cette opération permet de supprimer un AudioSourceConfiguration d'un profil multimédia existant (voir Tableau 133). Si le profil multimédia ne contient pas d'AudioSourceConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Un NVT qui prend en charge la transmission audio en continu d'un NVT vers un client doit prendre en charge la suppression d'une configuration de source audio d'un profil grâce à la commande RemoveAudioSourceConfiguration.

Il convient que les configurations de source audio soient supprimées uniquement après la suppression d'un AudioEncoderConfiguration du profil multimédia.

Tableau 133 – Commande RemoveAudioSourceConfiguration

RemoveAudioSourceConfiguration		Demande-Réponse
Nom du message	Description	
RemoveAudioSourceConfiguration-Request	Contient une référence au profil multimédia duquel l'AudioSourceConfiguration doit être supprimée. tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioSourceConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration de source audio dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de l'AudioSourceConfiguration et sa suppression génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>L'audio n'est pas pris en charge.</i>	

11.2.16 Suppression de configuration de codeur audio d'un profil

Cette opération permet de supprimer un AudioEncoderConfiguration d'un profil multimédia existant (voir Tableau 134). Si le profil multimédia ne contient pas d'AudioEncoderConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la suppression de configurations de codeur audio d'un profil grâce à la commande RemoveAudioEncoderConfiguration.

Tableau 134 – Commande RemoveAudioEncoderConfiguration

RemoveAudioEncoderConfiguration		Demande-Réponse
Nom du message	Description	
RemoveAudioEncoderConfiguration-Request	Contient une référence au profil multimédia duquel l'AudioEncoderConfiguration doit être supprimée. tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioEncoderConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration de codeur audio dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de l'AudioEncoderConfiguration et sa suppression génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>L'audio n'est pas pris en charge.</i>	

11.2.17 Suppression de configuration PTZ d'un profil

Cette opération permet de supprimer un PTZConfiguration d'un profil multimédia existant (voir Tableau 135). Si le profil multimédia ne contient pas de PTZConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Un NVT qui prend en charge le contrôle PTZ doit prendre en charge la suppression de configurations PTZ d'un profil grâce à la commande RemovePTZConfiguration.

Tableau 135 – Commande RemovePTZConfiguration

RemovePTZConfiguration		Demande-Réponse
Nom du message	Description	
RemovePTZConfiguration-Request	Contient une référence au profil multimédia duquel la PTZConfiguration doit être supprimée. tt:ReferenceToken ProfileToken [1][1]	
RemovePTZConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration PTZ dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de la PTZConfiguration et sa suppression génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

11.2.18 Suppression de configuration d'analyse vidéo d'un profil

Cette opération permet de supprimer un VideoAnalyticsConfiguration d'un profil multimédia existant (voir Tableau 136). Si le profil multimédia ne contient pas de VideoAnalyticsConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Un NVT qui prend en charge l'analyse vidéo doit prendre en charge la suppression d'une configuration d'analyse vidéo d'un profil grâce à la commande RemoveVideoAnalyticsConfiguration.

Tableau 136 – Commande RemoveVideoAnalyticsConfiguration

RemoveVideoAnalyticsConfiguration		Demande-Réponse
Nom du message	Description	
RemoveVideoAnalyticsConfiguration-Request	Contient une référence au profil multimédia duquel la VideoAnalyticsConfiguration doit être supprimée. tt:ReferenceToken ProfileToken [1][1]	
RemoveVideoAnalyticsConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration d'analyse vidéo dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de la VideoAnalyticsConfiguration et sa suppression génère un conflit de profil multimédia.</i>	
env:Receiver ter:ActionNotSupported ter:VideoAnalyticsNotSupported	<i>La VideoAnalytics (analyse vidéo) n'est pas prise en charge.</i>	

11.2.19 Suppression de configuration de métadonnées d'un profil

Cette opération permet de supprimer un MetadataConfiguration d'un profil multimédia existant (voir Tableau 137). Si le profil multimédia ne contient pas de MetadataConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Un NVT doit prendre en charge la suppression d'une configuration de métadonnées d'un profil grâce à la commande RemoveMetadataConfiguration.

Tableau 137 – Commande RemoveMetadataConfiguration

RemoveMetadataConfiguration		Demande-Réponse
Nom du message	Description	
RemoveMetadataConfiguration-Request	Contient une référence au profil multimédia duquel la MetadataConfiguration doit être supprimée. tt:ReferenceToken ProfileToken [1][1]	
RemoveMetadataConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration de métadonnées dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de la MetadataConfiguration et sa suppression génère un conflit de profil multimédia.</i>	

11.2.20 Suppression de configuration de sortie audio

Cette opération permet de supprimer un AudioOutputConfiguration d'un profil multimédia existant (voir Tableau 138). Si le profil multimédia ne contient pas d'AudioOutputConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Un NVT qui comporte au moins une sortie audio doit prendre en charge la suppression d'une configuration de sortie audio d'un profil grâce à la commande RemoveAudioOutputConfiguration.

Tableau 138 – RemoveAudioOutputConfiguration

RemoveAudioOutputConfiguration		Demande-Réponse
Nom du message	Description	
RemoveAudioOutputConfiguration-Request	<i>Contient une référence au profil multimédia duquel l'AudioOutputConfiguration doit être supprimé.</i> tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioOutputConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration de sortie audio dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes de l'AudioOutputConfiguration et sa suppression génère un conflit de profil multimédia.</i>	
env: Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>L'Audio ou la sortie audio n'est pas pris en charge</i>	

11.2.21 Suppression de configuration de décodeur audio

Cette opération permet de supprimer un AudioDecoderConfiguration d'un profil multimédia existant (voir Tableau 139). Si le profil multimédia ne contient pas d'AudioDecoderConfiguration, l'opération n'a aucun effet. La suppression doit être persistante. Un NVT qui prend en charge le décodage audio doit prendre en charge la suppression d'une configuration de décodeur audio d'un profil grâce à la commande RemoveAudioDecoderConfiguration.

Tableau 139 – RemoveAudioDecoderConfiguration

RemoveAudioDecoderConfiguration		Demande-Réponse
Nom du message	Description	
RemoveAudioDecoderConfiguration-Request	<i>Contient une référence au profil multimédia duquel l'AudioDecoderConfiguration doit être supprimé.</i> tt:ReferenceToken ProfileToken [1][1]	
RemoveAudioDecoderConfiguration-Response	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>Il n'existe pas de configuration de décodeur audio dans le profil multimédia.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>D'autres configurations du profil multimédia sont dépendantes d'AudioDecoderConfiguration, et sa suppression génère un conflit de profil multimédia.</i>	
env: Receiver ter:ActionNotSupported ter::AudioDecodingNotSupported	<i>Audio ou AudioDecoding n'est pas pris en charge</i>	

11.2.22 Suppression de profil multimédia

Cette opération permet de supprimer un profil (voir Tableau 140). Ce changement doit toujours être persistant. Le NVT doit prendre en charge la suppression d'un profil multimédia grâce à la commande DeleteProfile.

Tableau 140 – Commande DeleteProfile

DeleteProfile		Demande-Réponse
Nom du message	Description	
DeleteProfileRequest	<i>Contient un ProfileToken qui indique le profil multimédia à supprimer.</i> tt:ReferenceToken ProfileToken [1][1]	
DeleteProfileResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:Action ter:DeletionOfFixedProfile	<i>Le profil fixe ne peut pas être supprimé.</i>	

11.3 Source vidéo

11.3.1 Généralités

Un VideoSource représente une entrée vidéo non codée. La structure contient la résolution en pixels de la vidéo, le taux de frame et les paramètres d'imagerie. Les paramètres d'imagerie peuvent être manipulés par l'intermédiaire de la commande ImagingService, si elle est prise en charge et contient des paramètres de mise au point, d'exposition et de luminosité, par exemple. Voir Article 0 pour plus d'informations.

11.3.2 GetVideoSources

Cette opération permet d'extraire toutes les sources vidéo disponibles pour le dispositif (voir Tableau 141). Le NVT doit prendre en charge la liste des sources vidéo disponibles grâce à la commande GetVideoSources.

Tableau 141 – Commande GetVideoSources

GetVideoSources		Demande-Réponse
Nom du message	Description	
GetVideoSourcesRequest	<i>Ceci est un message vide.</i>	
GetVideoSourcesResponse	<i>Contient une liste des structures décrivant toutes les sources vidéo disponibles du dispositif.</i> tt:VideoSource VideoSources [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

11.4 Configuration de source vidéo

Une VideoSourceConfiguration (configuration de source vidéo) contient une référence à une VideoSource (source vidéo) et une structure Bounds (bornes) contenant la zone de pixels totale de la VideoSource (source vidéo) ou une sous-portion de celle-ci. Les Bounds et VideoSource définissent l'image qui est transmise en continu à un client. Si un VideoSourceConfiguration est utilisé à l'intérieur d'un profil la valeur de son paramètre UseCount est augmentée pour indiquer qu'une modification de cette configuration peut affecter d'autres utilisateurs.

11.4.1 Obtention de configurations de source vidéo

Cette opération permet de répertorier toutes les configurations de source vidéo *existantes* d'un NVT. Cette commande permet de répertorier *toutes* les configurations de source vidéo d'un dispositif. Il n'est pas nécessaire que le client dispose d'informations sur les configurations de source vidéo pour utiliser la commande. Le NVT doit prendre en charge la liste des configurations de source vidéo disponibles grâce à la commande GetVideoSourceConfigurations (voir Tableau 142).

Tableau 142 – Commande GetVideoSourceConfigurations

GetVideoSourceConfigurations		Demande-Réponse
Nom du message	Description	
GetVideoSourceConfigurations-Request	<i>Ceci est un message vide.</i>	
GetVideoSourceConfigurations-Response	<i>Ce message contient une liste de toutes les configurations de source vidéo existantes dans le NVT. Une configuration de source vidéo pointe toujours vers une source vidéo réelle avec l'élément SourceToken.</i> tt:VideoSourceConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

11.4.2 Obtention de configuration de source vidéo

Si le jeton de configuration de source vidéo est déjà connu, la configuration de source vidéo peut être extraite grâce à la commande `GetVideoSourceConfiguration`. Le NVT doit prendre en charge l'extraction de configurations de source vidéo spécifiques grâce à la commande `GetVideoSourceConfiguration` (voir Tableau 143).

Tableau 143 – Commande `GetVideoSourceConfiguration`

GetVideoSourceConfiguration		Demande-Réponse
Nom du message	Description	
GetVideoSourceConfiguration-Request	<i>Ce message contient le jeton de la configuration de source vidéo demandée.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoSourceConfiguration-Response	<i>Ce message contient la VideoSourceConfiguration demandée avec le jeton correspondant. Une configuration de source vidéo pointe toujours vers une source vidéo réelle avec l'élément SourceToken.</i> tt:VideoSourceConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	

11.4.3 Obtention de configurations de source vidéo compatibles

Cette opération permet de demander toutes les configurations de source vidéo du NVT compatibles avec un certain profil multimédia. Chacune des configurations retournées doit être un paramètre d'entrée valide pour la commande `AddVideoSourceConfiguration` sur le profil multimédia. Le résultat varie suivant les fonctionnalités, configurations et paramètres du dispositif. Le NVT doit prendre en charge la liste des configurations de source vidéo compatibles (avec un profil spécifique) grâce à la commande `GetCompatibleVideoSourceConfigurations` (voir Tableau 144).

Tableau 144 – Commande GetCompatibleVideoSourceConfigurations

GetCompatibleVideoSourceConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleVideoSource-ConfigurationsRequest	<i>Contient le jeton d'un profil multimédia existant.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleVideoSource-ConfigurationsResponse	<i>Contient une liste de configurations de source vidéo qui sont compatibles avec le profil multimédia.</i> tt:VideoSourceConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	

11.4.4 Obtention des options de configuration de source vidéo

Cette opération retourne les options disponibles en cas de reconfiguration des paramètres de source vidéo. Si une configuration de source vidéo est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Le NVT doit prendre en charge la liste des options de paramètre de source vidéo disponibles (pour un profil et une configuration donnés) grâce à la commande GetVideoSourceConfigurationOptions (voir Tableau 145).

Tableau 145 – Commande GetVideoSourceConfigurationOptions

GetVideoSourceConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetVideoSourceConfiguration-OptionsRequest	Ce message contient les jetons facultatifs d'une configuration de source vidéo et un profil multimédia. ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées. <i>ProfileToken spécifie un profil multimédia existant avec lequel les options doivent être compatibles.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetVideoSourceConfiguration-OptionsResponse	<i>Ce message contient les options de configuration vidéo. Si une configuration de source vidéo est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Si aucun jeton n'est spécifié, les options doivent être considérées comme étant génériques pour le dispositif.</i> tt:VideoSourceConfigurationOptions Options [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée n'existe pas.	

11.4.5 Modification d'une configuration de source vidéo

Cette opération permet de modifier une configuration de source vidéo. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du NVT. Les flux en cours d'exécution utilisant cette configuration peuvent immédiatement être mis à jour conformément aux nouveaux paramètres. Il n'est pas garanti que les modifications prennent effet, sauf si le client demande un nouvel URI de flux et redémarre les flux affectés. Les méthodes NVC de modification d'un flux en cours d'exécution sont hors du domaine d'application de la présente norme. Le NVT doit prendre en charge la modification de paramètres de source vidéo grâce à la commande SetVideoSourceConfiguration (voir Tableau 146).

Tableau 146 – Commande SetVideoSourceConfiguration

SetVideoSourceConfiguration		Demande-Réponse
Nom du message	Description	
SetVideoSourceConfiguration-Request	<i>L'élément Configuration contient la configuration de source vidéo modifiée. La configuration doit exister dans le NVT.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:VideoSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoSourceConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.	

11.5 Configuration de codeur vidéo

Un VideoEncoderConfiguration contient les paramètres suivants de configuration du codage de données vidéo:

- Encoder – codage utilisé pour les données vidéo;
- Resolution – résolution de pixel des données vidéo codées;
- Quality – détermine la qualité de la vidéo. Une valeur élevée dans la plage de qualité prise en charge indique une qualité supérieure;
- RateControl – définit les paramètres de configuration du débit binaire [kbps], un paramètre EncodingInterval (intervalle de codage et de transmission des images) et un paramètre FrameRateLimit [fps] de configuration du taux de trame de sortie;
- Spécificités MPEG4/H264 – définit le profil de codage et la longueur de GOV [trame].

La structure de VideoEncoderConfiguration contient également des paramètres de multidiffusion et un délai d'attente de session pour définir le comportement de transmission vidéo continue. Si un VideoEncoderConfiguration est utilisé à l'intérieur d'un profil, la valeur

de son paramètre UseCount est augmentée pour indiquer qu'une modification de cette configuration peut affecter d'autres utilisateurs.

11.5.1 Obtention de configurations de codeur vidéo

Cette opération permet de répertorier toutes les configurations de codeur vidéo *existantes* d'un NVT. Cette commande permet de répertorier *toutes* les configurations de codeur vidéo configurées d'un dispositif. A priori, il n'est pas nécessaire que le client dispose d'informations relatives aux configurations de codeur vidéo pour utiliser la commande. Le NVT doit prendre en charge la liste des configurations de codeur vidéo disponibles grâce à la commande GetVideoEncoderConfigurations (voir Tableau 147).

Tableau 147 – Commande GetVideoEncoderConfigurations

GetVideoEncoderConfigurations		Demande-Réponse
Nom du message	Description	
GetVideoEncoderConfigurations-Request	<i>Ceci est un message vide.</i>	
GetVideoEncoderConfigurations-Response	<i>Ce message contient une liste de toutes les configurations de codeur vidéo existantes dans le NVT.</i> tt:VideoEncoderConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

11.5.2 Obtention de configuration de codeur vidéo

Si le jeton de configuration de codeur vidéo est déjà connu, la configuration de codeur peut être extraite grâce à la commande GetVideoEncoderConfiguration. Le NVT doit prendre en charge l'extraction d'une configuration de codeur vidéo spécifique grâce à la commande GetVideoEncoderConfiguration (voir Tableau 148).

Tableau 148 – Commande GetVideoEncoderConfiguration

GetVideoEncoderConfiguration		Demande-Réponse
Nom du message	Description	
GetVideoEncoderConfiguration-Request	<i>Ce message contient le jeton de la configuration de codeur vidéo demandée.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoEncoderConfiguration-Response	<i>Ce message contient la VideoEncoderConfiguration demandée avec le jeton correspondant.</i> tt:VideoEncoderConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	

11.5.3 Obtention de configurations de codeur vidéo compatibles

Cette opération permet de répertorier toutes les configurations de codeur vidéo du NVT compatibles avec un certain profil multimédia. Chacune des configurations retournées doit être un paramètre d'entrée valide pour la commande AddVideoEncoderConfiguration sur le profil multimédia. Le résultat varie suivant les fonctionnalités, configurations et paramètres du dispositif. Le NVT doit prendre en charge la liste des configurations de codeur vidéo compatibles (avec un profil spécifique) grâce à la commande GetCompatibleVideoEncoderConfigurations (voir Tableau 149).

Tableau 149 – Commande GetCompatibleVideoEncoderConfigurations

GetCompatibleVideoEncoderConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleVideoEncoderConfigurationsRequest	<i>Contient le jeton d'un profil multimédia existant.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleVideoEncoderConfigurationsResponse	<i>Contient une liste de configurations de codeur vidéo qui sont compatibles avec le profil multimédia en question.</i> tt:VideoEncoderConfiguration Configurations [0]non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	

11.5.4 Obtention d'options de configuration de codeur vidéo

Cette opération retourne les options disponibles en cas de reconfiguration des paramètres de codeur vidéo. Le NVT doit prendre en charge la liste des options de paramètre vidéo disponibles (pour un profil et une configuration donnés) grâce à la commande GetVideoEncoderConfigurationOptions (voir Tableau 150).

Tableau 150 – Commande GetVideoEncoderConfigurationOptions

GetVideoEncoderConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetVideoEncoderConfiguration-OptionsRequest	Ce message contient les jetons facultatifs d'une configuration de codeur vidéo et un profil multimédia. ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées. <i>ProfileToken spécifie un profil multimédia existant avec lequel les options doivent être compatibles.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetVideoEncoderConfiguration-OptionsResponse	<i>Ce message contient les options de configuration vidéo. Si une configuration de codeur vidéo est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Si aucun jeton n'est spécifié, les options doivent être considérées comme étant génériques pour le dispositif.</i> tt:VideoEncoderConfigurationOptions Options [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée n'existe pas.	

11.5.5 Modification d'une configuration de codeur vidéo

Cette opération permet de modifier une configuration de codeur vidéo. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du NVT. Les modifications des paramètres de multidiffusion doivent toujours être persistantes. Les flux en cours d'exécution utilisant cette configuration peuvent être immédiatement mis à jour en fonction des nouveaux paramètres, mais leur prise d'effet n'est pas garantie tant que le client n'a pas demandé une nouvelle URI de flux et redémarré tous les flux affectés. Si les nouveaux paramètres invalident les paramètres déjà négociés à l'aide de RTSP (en modifiant le type de codec, par exemple), le NVT ne doit pas les appliquer aux flux existants. Il doit plutôt continuer le flux à l'aide des anciens paramètres ou arrêter l'envoi de données sur les flux concernés.

Les méthodes NVC de modification d'un flux en cours d'exécution sont hors du domaine d'application de la présente norme. Le NVT doit prendre en charge la modification de paramètres de codeur vidéo grâce à la commande SetVideoEncoderConfiguration (voir Tableau 151).

Tableau 151 – Commande SetVideoEncoderConfiguration

SetVideoEncoderConfiguration		Demande-Réponse
Nom du message	Description	
SetVideoEncoderConfiguration-Request	<i>L'élément Configuration contient la configuration de codeur vidéo modifiée. La configuration doit exister dans le NVT.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:VideoEncoderConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoEncoderConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.	

11.5.6 Obtention d'un nombre garanti d'instances de codeur vidéo

La commande GetGuaranteedNumberOfVideoEncoderInstance peut être utilisée pour demander le nombre minimal garanti d'instances de codeur vidéo (applications) par configuration de source vidéo. Un NVT doit prendre en charge cette commande. Cette commande a été ajoutée dans ONVIF 1.02. Voir Tableau 152.

Tableau 152 – Commande GetGuaranteedNumberOfVideoEncoderInstances

GetGuaranteedNumberOfVideoEncoderInstances		Demande-Réponse
Nom du message		Description
GetGuaranteedNumberOfEncoderInstancesRequest	<i>Cette demande contient un jeton pour la configuration de source vidéo.</i> tt: ReferenceToken ConfigurationToken [1][1]	
GetGuaranteedNumberOfEncoderInstancesResponse	<i>Ce message contient le TotalNumber minimal garanti d'instances de codeur (applications) par VideoSourceConfiguration. Si un dispositif limite le nombre d'instances pour des codecs vidéo respectifs, la réponse contient les informations quant au nombre de Jpeg, H264 et Mpeg4 qui peuvent être configurés simultanément. Dans tous les autres cas, le dispositif est capable de délivrer le TotalNumber de flux indépendamment du VideoCodec configuré simultanément.</i> xs:int TotalNumber [1][1] xs:int JPEG [0][1] xs:int H264 [0][1] xs:int MPEG4 [0][1]	
Codes de défaut		Description
env:Sender		<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>
ter:InvalidArgVal		
ter:NoConfig		

11.6 Source audio

11.6.1 Généralités

Une AudioSource représente une entrée audio non codée et déclare le nombre de canaux d'entrée.

11.6.2 Obtention de sources audio

Cette opération permet de répertorier toutes les sources audio disponibles du dispositif. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la liste des sources audio disponibles grâce à la commande GetAudioSources (voir Tableau 153).

Tableau 153 – Commande GetAudioSources

GetAudioSources		Demande-Réponse
Nom du message	Description	
GetAudioSourcesRequest	<i>Ce message est vide.</i>	
GetAudioSourcesResponse	<i>Contient une liste de structures décrivant toutes les sources audio disponibles du dispositif.</i> tt:AudioSource AudioSources [0][non limité]	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Le NVT ne prend pas en charge l'audio.</i>	

11.7 Configuration de source audio

Une AudioSourceConfiguration (configuration de source audio) contient une référence à une AudioSource (source audio) à utiliser pour entrée dans un profil multimédia. Si un AudioSourceConfiguration est utilisé à l'intérieur d'un profil, la valeur de son paramètre UseCount est augmentée pour indiquer qu'une modification de cette configuration peut affecter d'autres utilisateurs.

11.7.1 Obtention de configurations de source audio

Cette opération permet de répertorier toutes les configurations de source audio *existantes* d'un NVT. Cette commande permet de répertorier *toutes* les configurations de source audio d'un dispositif. A priori, il n'est pas nécessaire que le client dispose d'informations relatives aux configurations de source audio pour utiliser la commande. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la liste des configurations de source audio disponibles grâce à la commande GetAudioSourceConfigurations (voir Tableau 154).

Tableau 154 – Commande GetAudioSourceConfigurations

GetAudioSourceConfigurations		Demande-Réponse
Nom du message	Description	
GetAudioSourceConfigurations-Request	<i>Ceci est un message vide.</i>	
GetAudioSourceConfigurations-Response	<i>Ce message contient une liste de toutes les configurations de source audio existantes dans le NVT. Une configuration de source audio pointe toujours vers une source audio réelle avec l'élément SourceToken.</i> tt:AudioSourceConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Le NVT ne prend pas en charge l'audio.</i>	

11.7.2 Obtention de configuration de source audio

La commande GetAudioSourceConfiguration extrait les configurations de source audio si le jeton de configuration de source audio est déjà connu. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge l'extraction d'une configuration de source audio spécifique par l'intermédiaire de la commande GetAudioSourceConfiguration (voir Tableau 155).

Tableau 155 – Commande GetAudioSourceConfiguration

GetAudioSourceConfiguration		Demande-Réponse
Nom du message	Description	
GetAudioSourceConfiguration-Request	<i>Ce message contient le jeton de la configuration de source audio demandée. Une configuration de source audio pointe toujours vers une source audio réelle avec l'élément SourceToken.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioSourceConfiguration-Response	<i>Ce message contient l'AudioSourceConfiguration demandé avec le jeton correspondant.</i> tt:AudioSourceConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Le NVT ne prend pas en charge l'audio.</i>	

11.7.3 Obtention de configurations de source audio compatibles

Cette opération permet de demander toutes les configurations de codeur audio d'un dispositif compatibles avec un certain profil multimédia. Chacune des configurations retournées doit être un paramètre d'entrée valide pour la commande AddAudioSourceConfiguration sur le profil multimédia. Le résultat varie suivant les fonctionnalités, les configurations et les réglages dans le dispositif. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la liste des configurations de source audio compatibles (avec un profil spécifique) grâce à la commande GetCompatibleAudioSourceConfigurations (voir Tableau 156).

Tableau 156 – Commande GetCompatibleAudioSourceConfigurations

GetCompatibleAudioSourceConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleAudioSource-ConfigurationsRequest	<i>Contient le jeton d'un profil multimédia existant.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioSource-ConfigurationsResponse	<i>Contient une liste de configurations de source audio qui sont compatibles avec le profil multimédia.</i> tt:AudioSourceConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Le NVT ne prend pas en charge l'audio.</i>	

11.7.4 Obtention d'options de configuration de source audio

Cette opération retourne les options disponibles lorsque les paramètres de source audio sont reconfigurés. Si une configuration de source audio est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Un NVT qui prend en charge la transmission audio continue d'un NVT à un client doit prendre en charge la liste des options de paramètres audio disponibles (pour un profil et une configuration donnés) grâce à la commande GetAudioSourceConfigurationOptions (voir Tableau 157).

Tableau 157 – Commande GetAudioSourceConfigurationOptions

GetAudioSourceConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetAudioSourceConfiguration-OptionsRequest	Ce message contient les jetons facultatifs d'une configuration de source audio et un profil multimédia. ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées. <i>ProfileToken spécifie un profil multimédia existant avec lequel les options doivent être compatibles.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioSourceConfiguration-OptionsResponse	<i>Ce message contient les options de configuration audio. Si une configuration de source audio est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Si aucun jeton n'est spécifié, les options doivent être considérées comme étant génériques pour le dispositif.</i> tt:AudioSourceConfigurationOptions Options [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée n'existe pas.	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	Le NVT ne prend pas en charge l'audio.	

11.7.5 Modification d'une configuration de source audio

Cette opération modifie une configuration de source audio. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du NVT. Les flux en cours d'exécution utilisant cette configuration peuvent être immédiatement mis à jour en fonction des nouveaux paramètres, mais leur prise d'effet n'est pas garantie tant que le client n'a pas demandé une nouvelle URI de flux et redémarré tous les flux affectés. Si les nouveaux paramètres invalident les paramètres déjà négociés à l'aide de RTSP (en modifiant le type de codec, par exemple), le NVT ne doit pas les appliquer aux flux existants. Il doit plutôt

continuer le flux à l'aide des anciens paramètres ou arrêter l'envoi de données sur les flux concernés.

Les méthodes NVC de modification d'un flux en cours d'exécution sont hors du domaine d'application de la présente norme. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la configuration des paramètres de source audio grâce à la commande SetAudioSourceConfiguration (voir Tableau 158).

Tableau 158 – Commande SetAudioSourceConfiguration

SetAudioSourceConfiguration		Demande-Réponse
Nom du message	Description	
SetAudioSourceConfiguration-Request	<i>L'élément Configuration contient la configuration de source audio modifiée. La configuration doit exister dans le NVT.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AudioSourceConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioSourceConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	Le NVT ne prend pas en charge l'audio.	

11.8 Configuration de codeur audio

Un AudioEncoderConfiguration contient les paramètres suivants de codage des données audio:

- Encoder – codage utilisé pour les données audio;

- Bitrate – débit binaire de sortie [kbps];
- SampleRate – fréquence d'échantillonnage de sortie [kHz].

La structure d'AudioEncoderConfiguration contient en outre des paramètres de multidiffusion et un délai d'attente de session pour définir le comportement de transmission audio continue.

Si un AudioEncoderConfiguration est utilisé à l'intérieur d'un profil, la valeur de son paramètre UseCount est augmentée pour indiquer qu'une modification de cette configuration peut affecter d'autres utilisateurs.

11.8.1 Obtention de configurations de codeur audio

Cette opération permet de répertorier toutes les configurations de codeur audio existantes d'un dispositif. A priori, il n'est pas nécessaire que le client dispose d'informations relatives aux configurations de codeur audio pour utiliser la commande. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la liste des configurations de codeur audio disponibles grâce à la commande GetAudioEncoderConfigurations (voir Tableau 159).

Tableau 159 – Commande GetAudioEncoderConfigurations

GetAudioEncoderConfigurations		Demande-Réponse
Nom du message	Description	
GetAudioEncoderConfigurations-Request	<i>Ceci est un message vide.</i>	
GetAudioEncoderConfigurations-Response	<i>Ce message contient une liste de toutes les configurations de codeur audio existantes dans le NVT.</i> tt:AudioEncoderConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env:Receiver	<i>Le NVT ne prend pas en charge l'audio.</i>	
ter:ActionNotSupported		
ter:AudioNotSupported		

11.8.2 Obtention de configuration de codeur audio

La commande GetAudioEncoderConfiguration extrait la configuration de codeur si le jeton de configuration de codeur audio est connu. Un NVT qui prend en charge la transmission audio continue d'un NVT à un client doit prendre en charge la liste des configurations spécifiques de codeur audio grâce à la commande GetAudioEncoderConfiguration (voir Tableau 160).

Tableau 160 – Commande GetAudioEncoderConfiguration

GetAudioEncoderConfiguration		Demande-Réponse
Nom du message	Description	
GetAudioEncoderConfiguration-Request	<i>Ce message contient le jeton de la configuration de codeur audio demandée.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioEncoderConfiguration-Response	<i>Ce message contient l'AudioEncoderConfiguration demandé avec le jeton correspondant.</i> tt:AudioEncoderConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Le NVT ne prend pas en charge l'audio.</i>	

11.8.3 Obtention de configurations de codeur audio compatibles

Cette opération permet de demander toutes les configurations de codeur audio du NVT compatibles avec un certain profil multimédia. Chacune des configurations retournées doit être un paramètre d'entrée valide pour la commande AddAudioEncoderConfiguration sur le profil multimédia. Le résultat varie suivant les fonctionnalités, les configurations et les réglages dans le dispositif. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la liste des configurations de codeur audio compatibles (avec un profil spécifique) grâce à la commande GetCompatibleAudioEncoderConfigurations (voir Tableau 161).

Tableau 161 – Commande GetCompatibleAudioEncoderConfigurations

GetCompatibleAudioEncoderConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleAudioEncoderConfigurationsRequest	<i>Contient le jeton d'un profil multimédia existant.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioEncoderConfigurationsResponse	<i>Contient une liste de configurations de codeur audio qui sont compatibles avec le profil multimédia en question.</i> tt:AudioEncoderConfiguration Configurations [0]non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	<i>Le NVT ne prend pas en charge l'audio.</i>	

11.8.4 Obtention d'options de configuration de codeur audio

Cette opération retourne les options disponibles lorsque les paramètres de codeur audio sont reconfigurés. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la liste des options de paramètres de codeur audio disponibles (pour un profil et une configuration donnés) grâce à la commande GetAudioEncoderConfigurationOptions (voir Tableau 162).

Tableau 162 – Commande GetAudioEncoderConfigurationOptions

GetAudioEncoderConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetAudioEncoderConfiguration-OptionsRequest	Ce message contient les jetons facultatifs d'une configuration de codeur audio et un profil multimédia. ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées. <i>ProfileToken spécifie un profil multimédia existant avec lequel les options doivent être compatibles.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioEncoderConfiguration-OptionsResponse	<i>Ce message contient les options de configuration audio. Si une configuration de codeur audio est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Si aucun jeton n'est spécifié, les options doivent être considérées comme étant génériques pour le dispositif.</i> tt:AudioEncoderConfigurationOptions Options [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le jeton de profil demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée n'existe pas.	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	Le NVT ne prend pas en charge l'audio.	

11.8.5 Modification de configurations de codeur audio

Cette opération permet de modifier une configuration de codeur audio. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du NVT. Les modifications des paramètres de multidiffusion doivent toujours être persistantes. Les flux en cours d'exécution utilisant cette configuration peuvent immédiatement être mis à jour conformément aux nouveaux paramètres. Il n'est pas garanti que les modifications prennent effet, sauf si le client demande un nouvel URI de flux et redémarre les flux affectés. Les méthodes NVC de modification d'un flux en cours d'exécution sont hors du domaine

d'application de la présente norme. Un NVT qui prend en charge la transmission audio continue d'un NVT vers un client doit prendre en charge la configuration des paramètres de codeur audio grâce à la commande SetAudioEncoderConfiguration (voir Tableau 163).

Tableau 163 – Commande SetAudioEncoderConfiguration

SetAudioEncoderConfiguration		Demande-Réponse
Nom du message	Description	
SetAudioEncoderConfiguration-Request	<i>L'élément Configuration contient la configuration de codeur audio modifiée. La configuration doit exister dans le NVT.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AudioEncoderConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioEncoderConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.	
env:Receiver ter:ActionNotSupported ter:AudioNotSupported	Le NVT ne prend pas en charge l'audio.	

11.9 Configuration d'analyse vidéo

L'élément VideoAnalyticsConfiguration contient les paramètres d'un *moteur d'analyse* et d'un *moteur de règles* (voir 4.12). Par conséquent, le moteur d'analyse est composé de plusieurs modules qui peuvent être gérés par la partie de module d'analyse du service d'analyse. De manière similaire, le moteur de règles est composé de plusieurs règles qui peuvent être gérées par la partie de moteur de règle du service d'analyse. Les commandes suivantes sont introduites pour gérer la configuration d'analyse vidéo complète de manière atomique. Par exemple, la commande ModifyVideoAnalyticsConfiguration modifie la configuration de moteur d'analyse et de règle dans une opération atomique. Lorsqu'une configuration d'analyse vidéo

est présente dans un profil, la configuration de métadonnées peut activer la transmission continue de la description de scène dans les flux RTP (voir 11.10).

Un dispositif est susceptible de NE PAS permettre de référencer le même VideoAnalyticsConfiguration de plusieurs profils multimédia avec différents VideoSourceConfiguration. Si le dispositif le permet, il doit générer des descriptions de scène individuelles pour chaque profil, le système de coordonnées d'une description de scène étant associé à un VideoSourceConfiguration spécifique. Des règles de masquage et géométriques concernent en outre le système de coordonnées de VideoSourceConfiguration. Cela PEUT nécessiter de traiter séparément l'ensemble des analyses vidéo pour chaque VideoSourceConfiguration, même si elles font référence au même VideoSource.

Étant donné que les options d'un VideoAnalyticsConfiguration sont dynamiques et souvent spécifiques au fournisseur, elles peuvent être extraites par l'intermédiaire du service d'analyse vidéo uniquement.

11.9.1 Obtention de configurations d'analyse vidéo

Cette opération permet de répertorier les configurations d'analyse vidéo d'un dispositif. Cette commande permet de répertorier *toutes* les analyses vidéo configurées dans un dispositif. A priori, il n'est pas nécessaire que le client dispose d'informations relatives aux analyses vidéo pour utiliser la commande. Un dispositif qui prend en charge l'analyse vidéo doit prendre en charge la liste des configurations d'analyses vidéo disponibles grâce à la commande GetVideoAnalyticsConfigurations (voir Tableau 164).

Tableau 164 – Commande GetVideoAnalyticsConfigurations

GetVideoAnalyticsConfigurations		Demande-Réponse	
Nom du message	Description		
GetVideoAnalyticsConfigurations-Request	Ce message est vide.		
GetVideoAnalyticsConfigurations-Response	Ce message contient une liste de toutes les configurations d'analyse vidéo existantes dans le dispositif.		
	tt:VideoAnalyticsConfiguration	Configurations	[0][non limité]
Codes de défaut	Description		
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	Le dispositif ne prend pas en charge l'analyse vidéo.		

11.9.2 Obtention de configuration d'analyse vidéo

La commande GetVideoAnalyticsConfiguration extrait la configuration d'analyse vidéo si le jeton d'analyse vidéo est connu. Un dispositif qui prend en charge l'analyse vidéo doit prendre en charge la liste des configurations d'analyses vidéo spécifiques grâce à la commande GetVideoAnalyticsConfiguration (voir Tableau 165).

Tableau 165 – Commande GetVideoAnalyticsConfiguration

GetVideoAnalyticsConfiguration		Demande-Réponse
Nom du message	Description	
GetVideoAnalyticsConfiguration-Request	<i>Ce message contient le jeton d'une configuration d'analyse vidéo existante.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoAnalyticsConfiguration-Response	<i>Ce message contient la configuration d'analyse vidéo demandée.</i> tt:VideoAnalyticsConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	<i>Le dispositif ne prend pas en charge l'analyse vidéo.</i>	

11.9.3 Obtention de configurations d'analyse vidéo compatibles

Cette opération permet de demander toutes les configurations d'analyse vidéo du dispositif compatibles avec un certain profil multimédia. Chacune des configurations retournées doit être un paramètre d'entrée valide pour la commande AddVideoAnalyticsConfiguration sur le profil multimédia. Le résultat varie suivant les fonctionnalités, les configurations et les réglages dans le dispositif. Un dispositif qui prend en charge l'analyse vidéo doit prendre en charge la liste des configurations d'analyse vidéo compatibles (avec un profil spécifique) grâce à la commande GetCompatibleVideoAnalyticsConfigurations (voir Tableau 166).

Tableau 166 – Commande GetCompatibleVideoAnalyticsConfigurations

GetCompatibleVideoAnalyticsConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleVideoAnalytics-ConfigurationsRequest	<i>Contient le jeton d'un profil multimédia existant.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleVideoAnalytics-ConfigurationsResponse	<i>Contient une liste de configurations d'analyse vidéo qui sont compatibles avec le profil multimédia en question.</i> tt:VideoAnalyticsConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	<i>Le dispositif ne prend pas en charge l'analyse vidéo.</i>	

11.9.4 Modification d'une configuration d'analyse vidéo

Une configuration d'analyse vidéo est modifiée en utilisant cette commande. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du dispositif ou non. Les flux en cours d'exécution utilisant cette configuration doivent immédiatement être mis à jour conformément aux nouveaux paramètres. Sinon, des incohérences peuvent survenir entre la description de scène traitée par le moteur de règles et les notifications générées par le moteur d'analyse et le moteur de règles qui référencent le même jeton de configuration d'analyse vidéo. Un dispositif qui prend en charge l'analyse vidéo doit prendre en charge la configuration des paramètres d'analyse vidéo grâce à la commande SetVideoAnalyticsConfiguration (voir Tableau 167).

Tableau 167 – Commande SetVideoAnalyticsConfiguration

SetVideoAnalyticsConfiguration		Demande-Réponse
Nom du message	Description	
SetVideoAnalyticsConfiguration-Request	<i>L'élément Configuration contient la configuration d'analyse vidéo modifiée. La configuration doit exister dans le dispositif.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:VideoAnalyticsConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetVideoAnalyticsConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:NoConfig	La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.	
env:Sender ter:ActionNotSupported ter:VideoAnalyticsNot-Supported	Le dispositif ne prend pas en charge l'analyse vidéo.	

11.10 Configuration de métadonnées

Un MetadataConfiguration contient des paramètres de sélection des données à inclure dans le flux de métadonnées. Les choix comprennent le statut PTZ, la position PTZ, les événements tels que définis par un abonnement et les données d'analyse. Les données d'abonnement d'événement sont décrites en 15.5. Les paramètres d'analyse définissent les données à inclure à partir de la partie de moteur d'analyse du profil (voir 0).

La structure contient également les paramètres de multidiffusion utilisés pour configurer et commander la multidiffusion du flux de métadonnées. Un paramètre de délai d'attente de session définit le délai d'attente de session (voir 12.2.1.1.1)

Si un MetadataConfiguration est utilisé à l'intérieur d'un profil, la valeur de son paramètre UseCount est augmentée pour indiquer qu'une modification de cette configuration peut affecter d'autres utilisateurs.

11.10.1 Obtention de configurations de métadonnées

Cette opération permet de répertorier toutes les configurations de métadonnées *existantes*. A priori, il n'est pas nécessaire que le client dispose d'informations relatives aux métadonnées pour utiliser la commande. Un NVT ou un autre dispositif qui prend en charge la transmission en continu de métadonnées doit prendre en charge la liste des configurations de métadonnées existantes grâce à la commande GetMetadataConfigurations (voir Tableau 168).

Tableau 168 – Commande GetMetadataConfigurations

GetMetadataConfigurations		Demande-Réponse
Nom du message	Description	
GetMetadataConfigurations-Request	<i>Ce message est vide.</i>	
GetMetadataConfigurations-Response	<i>Ce message contient une liste de toutes les configurations de métadonnées existantes dans le dispositif.</i> tt:MetadataConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

11.10.2 Obtention de configuration de métadonnées

La commande GetMetadataConfiguration permet d'extraire la configuration de métadonnées si le jeton de métadonnées est connu. Un NVT ou un autre dispositif qui prend en charge la transmission en continu de métadonnées doit prendre en charge la liste des configurations de métadonnées spécifiques grâce à la commande GetMetadataConfiguration (voir Tableau 169).

Tableau 169 – Commande GetMetadataConfiguration

GetMetadataConfiguration		Demande-Réponse
Nom du message	Description	
GetMetadataConfiguration-Request	<i>Ce message contient le jeton d'une configuration de métadonnées existante.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetMetadataConfiguration-Response	<i>Ce message contient la configuration de métadonnées demandée.</i> tt:MetadataConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	

11.10.3 Obtention de configurations de métadonnées compatibles

Cette opération demande toutes les configurations de métadonnées du dispositif compatibles avec un certain profil multimédia. Chacune des configurations retournées doit être un paramètre d'entrée valide pour la commande AddMetadataConfiguration sur le profil multimédia. Le résultat varie suivant les fonctionnalités, les configurations et les réglages dans le dispositif. Un NVT ou un autre dispositif qui prend en charge la transmission en continu de métadonnées doit prendre en charge la liste des configurations de métadonnées (avec un profil spécifique) compatibles grâce à la commande GetCompatibleMetadataConfigurations (voir Tableau 170).

Tableau 170 – Commande GetCompatibleMetadataConfigurations

GetCompatibleMetadataConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleMetadata-ConfigurationsRequest	<i>Contient le jeton d'un profil multimédia existant.</i> tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleMetadata-ConfigurationsResponse	<i>Contient une liste de configurations de métadonnées qui sont compatibles avec le profil multimédia donné.</i> tt:MetadataConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	

11.10.4 Obtention d'options de configuration de métadonnées

Cette opération retourne les options disponibles de modification de la configuration de métadonnées. Un NVT ou un autre dispositif qui prend en charge la transmission en continu de métadonnées doit prendre en charge la liste des options de paramètre de métadonnées disponibles (pour un profil et une configuration donnés) grâce à la commande GetMetadataConfigurationOptions (voir Tableau 171).

Tableau 171 – Commande GetMetadataConfigurationOptions

GetMetadataConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetMetadataConfiguration-OptionsRequest	Ce message contient les jetons optionnels d'une configuration de métadonnées et un profil multimédia. ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées. ProfileToken spécifie un profil multimédia existant avec lequel les options doivent être compatibles. tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetMetadataConfiguration-OptionsResponse	<i>Ce message contient les options de configuration de métadonnées. Si une configuration de métadonnées est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Si aucun jeton n'est spécifié, les options doivent être considérées comme étant génériques pour le dispositif.</i> tt:MetadataConfigurationOptions Options [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le jeton de profil demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée n'existe pas.	

11.10.5 Modification d'une configuration de métadonnées

Cette opération modifie une configuration de métadonnées. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du dispositif. Les modifications des paramètres de multidiffusion doivent toujours être persistantes. Les flux en cours d'exécution en utilisant cette configuration peuvent immédiatement être mis à jour conformément aux nouveaux réglages. Il n'est pas garanti que les modifications prennent effet, sauf si le client demande un nouvel URI de flux et redémarre les flux affectés. Les méthodes NVC de modification d'un flux en cours d'exécution sont hors du domaine d'application de la présente norme. Un NVT ou un autre dispositif qui prend en charge la transmission en continu de métadonnées doit prendre en charge la configuration des

paramètres de métadonnées existantes grâce à la commande SetMetadataConfiguration (voir Tableau 172).

Tableau 172 – Commande SetMetadataConfiguration

SetMetadataConfiguration		Demande-Réponse
Nom du message	Description	
SetMetadataConfiguration-Request	<i>L'élément Configuration contient des paramètres de multidiffusion, ainsi qu'un ensemble de filtres déterminant les données à inclure dans le flux de métadonnées.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:MetadataConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetMetadataConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.	

11.11 Sorties audio

11.11.1 Généralités

La sortie audio représente les sorties audio physiques qui peuvent être connectées à un haut-parleur.

11.11.2 Obtention de sorties audio

Cette commande permet de répertorier toutes les sorties audio disponibles d'un dispositif. Un NVT comportant une ou plusieurs sorties audio physiques doit prendre en charge la liste des sorties audio disponibles grâce à la commande GetAudioOutputs (voir Tableau 173).

Tableau 173 – Commande GetAudioOutputs

GetAudioOutputs		Demande-Réponse
Nom du message	Description	
GetAudioOutputsRequest	<i>Ceci est un message vide.</i>	
GetAudioOutputsResponse	<i>Contient une liste des structures décrivant toutes les sorties audio disponibles du dispositif. Si un dispositif ne contient aucun AudioOutputs, une liste vide est retournée.</i> tt:AudioOutput AudioOutputs [0][non limité]	
Codes de défaut	Description	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio ou AudioOutputs n'est pas pris en charge par le NVT</i>	

11.12 Configuration de sortie audio

La configuration de sortie audio contient les paramètres suivants:

- SourceToken: référence à une sortie audio existante.
- OutputLevel: paramètre de configuration du volume de la sortie.
- SendPrimacy: paramètre qui peut être utilisé pour les NVT dotés d'entrée/de sortie audio bidirectionnelle non simultanée pour configurer la direction de transmission active (voir 11.14).

Si un AudioOutputConfiguration est utilisé à l'intérieur d'un profil, la valeur de son paramètre UseCount est augmentée pour indiquer qu'une modification de cette configuration peut affecter d'autres utilisateurs.

11.12.1 Obtention de configurations de sortie audio

Cette commande permet de répertorier tous les AudioOutputConfigurations d'un dispositif. A priori, il n'est pas nécessaire que le NVC dispose d'informations relatives aux configurations audio pour utiliser cette commande. Un NVT qui peut assurer une sortie audio doit prendre en charge la liste des AudioOutputConfigurations grâce à cette commande (voir Tableau 174).

Tableau 174 –GetAudioOutputConfiguration

GetAudioOutputConfigurations		Demande-Réponse
Nom du message	Description	
GetAudioOutputConfigurationsRequest	<i>Ceci est un message vide.</i>	
GetAudioOutputConfigurationsResponse	<i>Contient une liste des AudioOutputConfigurations disponibles sur le dispositif</i> tt:AudioOutputConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env: Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio ou AudioOutputs n'est pas pris en charge par le dispositif</i>	

11.12.2 Obtention d'une configuration de sortie audio

Si le jeton de configuration de sortie audio est déjà connu, la configuration de sortie peut être extraite grâce à la commande GetAudioOutputConfiguration. Un NVT doté d'une ou de plusieurs sorties audio doit prendre en charge l'extraction d'une configuration de sortie audio spécifique grâce à la commande GetAudioOutputConfiguration (voir Tableau 175).

Tableau 175 –GetAudioOutputConfiguration

GetAudioOutputConfiguration		Demande-Réponse
Nom du message	Description	
GetAudioOutputConfigurationRequest	<i>Ce message contient le jeton de la configuration AudioOutput demandée.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioOutputConfigurationResponse	<i>Ce message contient l'AudioOutputConfiguration demandé avec le jeton correspondant.</i> tt:AudioOutputConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	
env: Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	<i>Audio ou AudioOutputs n'est pas pris en charge par le dispositif</i>	

11.12.3 Obtention de configurations de sortie audio compatibles

Cette commande permet de répertorier toutes les configurations de sortie audio d'un dispositif compatibles avec un certain profil multimédia. Chaque configuration retournée doit être une entrée valide pour la commande AddAudioOutputConfiguration. Un NVT doté d'une ou de plusieurs sorties audio doit prendre en charge la liste des AudioOutputConfigurations (avec un profil spécifique) compatibles grâce à la commande GetCompatibleAudioOutputConfigurations (voir Tableau 176).

Tableau 176 – GetCompatibleAudioOutputConfiguration

GetCompatibleAudioOutputConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleAudioOutputConfigurations Request	Contient le jeton d'un profil multimédia existant. tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioOutputConfigurations Response	Contient une liste des configurations de sortie audio compatibles avec le profil multimédia donné. tt:AudioOutputConfiguration Configurations [0][non limité]	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	Audio ou AudioOutputs n'est pas pris en charge par le dispositif	

11.12.4 Obtention d'options de configuration de sortie audio

Cette opération retourne les options disponibles de configuration d'une sortie audio. Un NVT doté d'une ou de plusieurs sorties audio doit prendre en charge la liste des options de configuration de sortie audio disponibles (pour un profil et une configuration donnés) grâce à la commande GetAudioOutputConfigurationOptions (voir Tableau 177).

Tableau 177 – Commande GetAudioOutputConfigurationOptions

GetAudioOutputConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetAudioOutputConfiguration-OptionsRequest	<i>Ce message contient les jetons facultatifs d'une configuration de sortie audio et un profil multimédia.</i> <i>ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées.</i> <i>ProfileToken spécifie un profil multimédia existant avec lequel les options doivent être compatibles.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioOutputConfiguration-OptionsResponse	<i>Ce message contient les options de configuration de sortie audio. Si une configuration de sortie audio est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Si aucun jeton n'est spécifié, les options doivent être considérées comme étant génériques pour le dispositif.</i> tt:AudioOutputConfigurationOptions Options [1][1]	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoProfile		Le ProfileToken (jeton de profil) demandé n'existe pas.
env:Sender ter:InvalidArgVal ter:NoConfig		La configuration demandée n'existe pas.
env:Receiver ter:ActionNotSupported ter:AudioOutputNotSupported		Audio ou AudioOutputs n'est pas pris en charge par le dispositif

11.12.5 Modification de configuration de sortie audio

Cette opération modifie une configuration de sortie audio. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du dispositif. Un NVT doté

d'une ou de plusieurs sorties audio doit prendre en charge la modification des paramètres de sortie audio grâce à la commande SetAudioOutputConfiguration (voir Tableau 178).

Tableau 178 – SetAudioOutputConfiguration

SetAudioOutputConfiguration		Demande-Réponse
Nom du message	Description	
SetAudioOutputConfiguration-Request	<i>L'élément Configuration contient la configuration de sortie audio modifiée. La configuration doit exister dans le dispositif.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AudioOutputConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioOutputConfiguration-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify	Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.	
env: Receiver ter:ActionNotSupported ter:AudioOutputNotSupported	Audio ou AudioOutputs n'est pas pris en charge par le dispositif	

11.13 Configuration de décodeur audio

La configuration de décodeur audio ne contient aucun paramètre de configuration du décodage. Un décodeur doit décoder toutes les données qu'il reçoit (en fonction de ces fonctionnalités).

Si un AudioDecoderConfiguration est utilisé à l'intérieur d'un profil, la valeur de son paramètre UseCount est augmentée pour indiquer qu'une modification de cette configuration peut affecter d'autres utilisateurs.

11.13.1 Obtention de configurations de décodeur audio

Cette commande permet de répertorier tous les AudioDecoderConfigurations d'un dispositif.

A priori, il n'est pas nécessaire que le NVC dispose d'informations relatives aux configurations de décodeur audio pour utiliser la commande. Un NVT qui peut décoder l'audio doit prendre en charge la liste des AudioOutputConfigurations grâce à cette commande (voir Tableau 179).

Tableau 179 – GetAudioDecoderConfigurations

GetAudioDecoderConfigurations		Demande-Réponse
Nom du message	Description	
GetAudioDecoderConfigurationsRequest	<i>Ceci est un message vide.</i>	
GetAudioDecoderConfigurationsResponse	<i>Contient une liste des AudioDecoderConfigurations disponibles sur le dispositif</i> tt:AudioDecoderConfiguration Configurations [0][non limité]	
Codes de défaut		Description
env:Receiver	<i>Audio ou le décodage Audio n'est pas pris en charge par le dispositif</i>	
ter:ActionNotSupported		
ter:AudioDecodingNotSupported		

11.13.2 Obtention d'une configuration de décodeur audio

Si le jeton de configuration de décodeur audio est déjà connu, la configuration de décodeur peut être extraite grâce à la commande GetAudioDecoderConfiguration. Un NVT qui peut décoder l'audio doit prendre en charge l'extraction d'une configuration de décodeur audio spécifique grâce à la commande GetAudioDecoderConfiguration (voir Tableau 180).

Tableau 180 – GetAudioDecoderConfiguration

GetAudioDecoderConfiguration		Demande-Réponse
Nom du message	Description	
GetAudioDecoderConfigurationRequest	<i>Ce message contient le jeton de la configuration AudioDecoder demandée.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAudioDecoderConfigurationResponse	<i>Ce message contient l'AudioDecoderConfiguration demandé avec le jeton correspondant.</i> tt:AudioDecoderConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	<i>Audio ou le décodage Audio n'est pas pris en charge par le dispositif</i>	

11.13.3 Obtention de configurations de décodeur audio compatibles

Cette opération permet de répertorier toutes les configurations de décodeur audio du dispositif compatibles avec un certain profil multimédia. Chacune des configurations retournées doit être un paramètre d'entrée valide pour la commande AddAudioDecoderConfiguration sur le profil multimédia. Un NVT qui peut décoder l'audio doit prendre en charge la liste des configurations de décodeur audio compatibles (avec un profil spécifique) grâce à la commande GetCompatibleAudioDecoderConfigurations (voir Tableau 181).

Tableau 181 – GetCompatibleAudioDecoderConfigurations

GetCompatibleAudioDecoderConfigurations		Demande-Réponse
Nom du message	Description	
GetCompatibleAudioDecoderConfigurations Request	Contient le jeton d'un profil multimédia existant. tt:ReferenceToken ProfileToken [1][1]	
GetCompatibleAudioDecoderConfigurations Response	Contient une liste de configurations de décodeur audio compatibles avec le profil multimédia donné. tt:AudioDecoderConfiguration Configurations [0][non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	Audio ou le décodage Audio n'est pas pris en charge par le dispositif	

11.13.4 Obtention d'options de configuration de décodeur audio

Cette commande permet de répertorier les fonctionnalités de décodage audio pour un profil et une configuration donnés d'un dispositif. Un dispositif qui peut décoder l'audio doit prendre en charge l'extraction des AudioDecoderConfigurationOptions grâce à cette commande (voir Tableau 182).

Tableau 182 – GetAudioDecoderConfigurationOptions

GetAudioDecoderConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetAudioDecoderConfiguration-OptionsRequest	Ce message contient les jetons facultatifs d'une configuration de décodeur audio et un profil multimédia. ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées. <i>ProfileToken spécifie un profil multimédia existant avec lequel les options doivent être compatibles.</i> tt:ReferenceToken ConfigurationToken [0][1] tt:ReferenceToken ProfileToken [0][1]	
GetAudioDecoderConfiguration-OptionsResponse	Ce message contient les options de configuration de décodeur audio. Si une configuration de décodeur audio est spécifiée, les options doivent concerner cette configuration particulière. Si un profil multimédia est spécifié, les options doivent être compatibles avec ce profil multimédia. Si aucun jeton n'est spécifié, les options doivent être considérées comme étant génériques pour le dispositif. tt:AudioDecoderConfigurationOptions Options [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le ProfileToken (jeton de profil) demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée n'existe pas.	
env:Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported	Audio ou le décodage Audio n'est pas pris en charge par le dispositif	

11.13.5 Modification d'une configuration de décodeur audio

Cette opération permet de modifier une configuration de décodeur audio. Le drapeau ForcePersistence indique si les modifications doivent persister après le redémarrage du dispositif. Un NVT qui peut décoder l'audio doit prendre en charge la modification des

paramètres de décodeur audio grâce à la commande SetAudioDecoderConfiguration (voir Tableau 183).

Tableau 183 – SetAudioDecoderConfiguration

SetAudioDecoderConfiguration		Demande-Réponse
Nom du message	Description	
SetAudioDecoderConfiguration-Request	<i>L'élément Configuration contient la configuration AudioDecoder modifiée. La configuration doit exister dans le dispositif.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AudioDecoderConfiguration Configuration [1][1] xs:boolean ForcePersistence [1][1]	
SetAudioDecoderConfiguration-Response	Ce message est vide.	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoConfig		La configuration n'existe pas.
env:Sender ter:InvalidArgVal ter:ConfigModify		Les paramètres de configuration ne peuvent pas être définis.
env:Receiver ter:Action ter:ConfigurationConflict		Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.
env: Receiver ter:ActionNotSupported ter:AudioDecodingNotSupported		Audio ou le décodage Audio n'est pas pris en charge par le dispositif

11.14 Modes de voie audio

Une voie audio PEUT prendre en charge différents types de transmission audio. En mode de fonctionnement bidirectionnel simultané, aucun traitement particulier n'est requis. En mode de

fonctionnement bidirectionnel non simultané, il est nécessaire de permuter le sens de transmission direction.

Un paramètre Send-Primacy facultatif à l'intérieur d'AudioOutputConfiguration indique la direction actuellement en cours. Un NVC peut basculer d'un mode à l'autre en attribuant une valeur à AudioOutputConfiguration.

Les modes suivants sont définis pour Send-Primacy:

- www.onvif.org/ver20/HalfDuplex/Server
Le serveur peut envoyer des données audio au client. Dans ce mode, le client ne doit pas envoyer de données audio par voie de retour au NVT.
- www.onvif.org/ver20/HalfDuplex/Client
Le serveur peut envoyer des données audio au serveur par la voie de retour. Dans ce mode, le NVT ne doit pas envoyer de données audio par voie de retour au client.
- www.onvif.org/ver20/HalfDuplex/Auto
La manière de traiter l'envoi et la réception de données audio revient au dispositif.

L'annulation de l'écho acoustique est hors du domaine d'application de l'ONVIF.

11.15 URI de flux

11.15.1 Généralités

Un flux multimedia peut être mis en place et contrôlé par un protocole RTSP.

11.15.2 Demande d'URI de flux

Cette opération demande un URI qui peut être utilisé pour initier un flux multimédia en direct en utilisant RTSP en tant que protocole de commande. L'URI retournée doit rester définitivement valide, même en cas de modification du profil. ValidUntilConnect, ValidUntilReboot et Timeout Parameter doivent être définis en conséquence (ValidUntilConnect=false, ValidUntilReboot=false, timeout=PT0S). Le NVT doit prendre en charge l'extraction d'un URI de flux multimédia pour un profil multimédia spécifique grâce à la commande GetStreamUri (voir Tableau 184).

Pour assurer la totale compatibilité avec d'autres services ONVIF, il convient qu'un dispositif ne génère pas d'URI comportant plus de 128 octets.

Tableau 184 – Commande GetStreamUri

GetStreamUri		Demande-Réponse
Nom du message	Description	
GetStreamUriRequest	<i>L'élément StreamSetup contient deux parties. Le StreamType définit si un flux multimédia à diffusion unilatérale ou multidiffusion est demandé. Transport spécifie une chaîne de protocoles de transport définissant la tunnellation du flux multimédia via différents protocoles réseau.</i> <i>L'élément ProfileToken indique le profil multimédia à utiliser et définit la configuration du contenu du flux.</i> tt:StreamSetup StreamSetup [1][1] tt:ReferenceToken ProfileToken [1][1]	
GetStreamUriResponse	<i>Contient l'URI stable à utiliser pour demander le flux multimédia ainsi que les paramètres définissant la durée de vie de l'URI. La valeur "false" doit être attribuée aux paramètres ValidUntilConnect et ValidUntilReboot, la valeur PTOS devant être attribuée au paramètre de délai d'attente afin d'indiquer que cet URI de flux est définitivement valide, même en cas de modification du profil.</i> xs:anyURI Uri [1][1] xs:boolean InvalidAfterConnect [1][1] xs:boolean InvalidAfterReboot [1][1] xs:duration Timeout [1][1]	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoProfile		Le profil multimédia n'existe pas.
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup		La spécification de la partie StreamType ou Transport dans StreamSetup n'est pas prise en charge.
env:Sender ter:OperationProhibited ter:StreamConflict		La spécification de la partie StreamType ou Transport dans StreamSetup génère un conflit avec d'autres flux.
env:Receiver ter:Action ter:IncompleteConfiguration		Le profil multimédia spécifié contient des sources inutilisées ou des codeurs de configuration sans source correspondante.

11.16 Instantané

11.16.1 Généralités

Une seule image snapshot, composée d'une seule trame vidéo peut être obtenue.

11.16.2 Demande d'URI d'instantané

Un client de réseau utilise la commande GetSnapshotUri pour obtenir un instantané JPEG depuis le NVT. L'URI retournée doit rester définitivement valide, même en cas de modification du profil. ValidUntilConnect, ValidUntilReboot et Timeout Parameter doivent être définis en conséquence (ValidUntilConnect=false, ValidUntilReboot=false, timeout=PT0S). L'URI peut être utilisé pour l'acquisition d'une image JPEG par l'intermédiaire d'une opération GET HTTP. Le format d'image est toujours JPEG, quel que soit le réglage de codage dans le profil multimédia. Un NVT doit prendre en charge cette commande (voir Tableau 185).

Tableau 185 – Commande GetSnapshotUri

GetSnapshotUri		Demande-Réponse
Nom du message	Description	
GetSnapshotUriRequest	<i>L'élément ProfileToken indique le profil multimédia à utiliser et définit la source et les dimensions de l'instantané.</i> tt:ReferenceToken ProfileToken [1][1]	
GetSnapshotUriResponse	<i>Contient l'URI à utiliser pour l'acquisition d'un instantané au format JPEG, ainsi que les paramètres définissant la durée de vie de l'URI. La valeur "false" doit être attribuée aux paramètres ValidUntilConnect et ValidUntilReboot, la valeur PT0S devant être attribuée au paramètre de délai d'attente afin d'indiquer que cet URI de flux est définitivement valide, même en cas de modification du profil.</i> xs:anyURI Uri [1][1] xs:boolean InvalidAfterConnect [1][1] xs:boolean InvalidAfterReboot [1][1] xs:duration Timeout [1][1]	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoProfile		<i>Le profil multimédia n'existe pas.</i>
env:Receiver ter:Action ter:IncompleteConfiguration		<i>Le profil multimédia spécifié ne contient pas de référence à une configuration de codeur vidéo ou à une configuration de source vidéo.</i>

11.17 Multidiffusion

Voir 12.1 pour une discussion détaillée de la transmission continue en multidiffusion NVT et client.

11.17.1 Démarrage de transmission continue en multidiffusion

Cette commande permet de démarrer la transmission continue en multidiffusion à l'aide du profil multimédia spécifié d'un NVT. La transmission continue jusqu'à ce que la commande StopMulticastStreaming soit appelée pour le même profil. La transmission continue doit continuer après un redémarrage du NVT tant qu'une demande StopMulticastStreaming n'est pas reçue. Les adresses, port et TTL de multidiffusion sont configurés dans un VideoEncoderConfiguration, AudioEncoderConfiguration et MetadataConfiguration, respectivement. Un NVT qui prend en charge la transmission continue en multidiffusion vidéo, audio ou de métadonnées doit prendre en charge le démarrage d'un flux en multidiffusion grâce à la commande StartMulticastStreaming (voir Tableau 186).

Tableau 186 – Commande StartMulticastStreaming

StartMulticastStreaming		Demande-Réponse
Nom du message	Description	
StartMulticastStreaming-Request	<i>Contient le jeton du profil qui est utilisé pour définir le flux multidiffusion.</i> tt:ReferenceToken ProfileToken [1][1]	
StartMulticastStreaming-Response	Ce message est vide.	
Codes de défaut	Description	
env:Sender	Le profil n'existe pas.	
ter:InvalidArgVal		
ter:NoProfile		
env:Receiver	Le profil multimédia spécifié ne contient pas de référence à un codeur vidéo, une configuration de source vidéo, une source audio, une configuration de codeur audio ou une configuration de métadonnées	
ter:Action		
ter:IncompleteConfiguration		

11.17.2 Arrêt de transmission continue en multidiffusion

Cette commande arrête la transmission continue en multidiffusion à l'aide du profil multimédia spécifié d'un NVT. Un NVT qui prend en charge la transmission continue en multidiffusion vidéo, audio ou de métadonnées doit prendre en charge l'arrêt d'un flux multidiffusion grâce à la commande StopMulticastStreaming (voir Tableau 187).

Tableau 187 – Commande StopMulticastStreaming

StopMulticastStreaming		Demande-Réponse
Nom du message	Description	
StopMulticastStreaming-Request	<i>Contient le jeton du profil qui est utilisé pour définir le flux multidiffusion.</i> tt:ReferenceToken ProfileToken [1][1]	
StopMulticastStreaming-Response	<i>Ce message est vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le profil n'existe pas.</i>	
env:Receiver ter:Action ter:IncompleteConfiguration	<i>Le profil multimédia spécifié ne contient pas de référence à un codeur vidéo, une configuration de source vidéo, une source audio, une configuration de codeur audio ou une configuration de métadonnées</i>	

11.18 Points de synchronisation

11.18.1 Généralités

Les points de synchronisation permettent aux clients de décoder et d'utiliser correctement toutes les données après le point de synchronisation.

11.18.2 Définition de point de synchronisation

Par exemple, si un flux vidéo est configuré avec une distance de trame I élevée et qu'un client perd un paquet, le client n'affiche pas la vidéo tant que la trame I suivante n'est pas transmise. Dans de tels cas, le client peut demander un point de synchronisation qui impose au NVT d'ajouter une trame I dès que possible. Les clients peuvent demander des points de synchronisation pour des profils. Le NVT doit ajouter des points de synchronisation pour tous les flux associés à ce profil.

De même, un point de synchronisation est utilisé pour obtenir une mise à jour sur le PTZ total ou le statut d'événement par l'intermédiaire du flux de métadonnées.

Si un flux vidéo est associé au profil, une trame I doit être ajoutée à ce flux vidéo. Si un flux d'événement est associé au profil, la demande de point de synchronisation doit être gérée comme indiqué en 15.6). Si un flux de métadonnées de PTZ est associé au profil, la position PTZ doit être répétée dans le flux de métadonnées.

Un NVT qui prend en charge MPEG-4 ou H.264 doit prendre en charge la demande de trame I grâce à la commande SetSynchronizationPoint (voir Tableau 188).

Tableau 188 – Commande SetSynchronizationPoint

SetSynchronizationPoint		Demande-réponse
Nom du message	Description	
SetSynchronizationPointRequest	Contient une référence de profil pour laquelle un point de synchronisation est demandé. tt:ReferenceToken ProfileToken [1][1]	
SetSynchronizationPointResponse	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le profil n'existe pas.	

11.19 Codes de défaut spécifiques au service

Le Tableau 189 ci-dessous présente les codes de défaut spécifiques au service multimédia. De plus, chaque commande peut également générer un défaut générique (voir Tableau 6).

Les défauts spécifiques sont définis en tant que sous-code d'un défaut générique (voir 5.11.2.1). Le sous-code générique parent est le *sous-code* en haut de chaque ligne ci-dessous, le *sous-code* de défaut spécifique se trouvant en bas de la cellule.

Tableau 189 – Codes de défaut spécifiques au service multimédia

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Receiver	ter:ActionNotSupported	Pas de fonctionnalité audio	Le NVT ne prend pas en charge l'audio.
	ter:AudioNotSupported		
env:Receiver	ter:Action	Nombre maximal atteint	Le nombre maximal de profils pris en charge a été atteint.
	ter:MaxNVTProfiles		
env:Receiver	ter:ActionNotSupported	Pas de fonctionnalité de sortie audio	Audio ou AudioOutputs n'est pas pris en charge par le NVT
	ter:AudioOutputNotSupported		
env:Receiver	ter:ActionNotSupported	Pas de fonctionnalité de décodage audio	Audio ou le décodage Audio n'est pas pris en charge par le NVT
	ter:AudioDecodingNotSupported		
env:Receiver	ter:Action	Configuration incomplète	Les entités requises par cette action sont manquantes dans le profil spécifié.
	ter:IncompleteConfiguration		
env:Receiver	ter:Action	Conflit d'utilisation des nouveaux paramètres	Les nouveaux réglages entrent en conflit avec d'autres utilisations de la configuration.
	ter:ConfigurationConflict		
env:Sender	ter:InvalidArgVal	Le jeton de profil existe déjà.	Un profil avec le jeton ProfileToken existe déjà.
	ter:ProfileExists		
env:Sender	ter:InvalidArgVal	Le jeton de configuration n'existe pas.	La configuration demandée indiquée par le ConfigurationToken n'existe pas.
	ter:NoConfig		
env:Sender	ter:InvalidArgVal	Le jeton de profil n'existe pas.	Le ProfileToken (jeton de profil) demandé n'existe pas.
	ter:NoProfile		
env:Sender	ter:Action	Un profil fixe ne peut pas être supprimé.	Le profil fixe ne peut pas être supprimé.
	ter:DeletionOfFixedProfile		
env:Sender	ter:InvalidArgVal	Les paramètres ne peuvent pas être définis.	Les paramètres de configuration ne peuvent pas être définis.
	ter:ConfigModify		
env:Sender	ter:ActionNotSupported	Pas de fonctionnalité d'analyse vidéo	Le NVT ne prend pas en charge l'analyse vidéo.
	ter:VideoAnalyticsNot-Supported		
env:Sender	ter:InvalidArgVal	Configuration de flux non valide	La spécification de la partie StreamType ou Transport dans StreamSetup n'est pas prise en charge.
	ter:InvalidStreamSetup		
env:Sender	ter:OperationProhibited	Conflit de flux	La spécification de la partie StreamType ou Transport dans StreamSetup génère un conflit avec d'autres flux.
	ter:StreamConflict		

12 Transmission continue en temps réel

Cet article décrit la transmission continue en temps réel de vidéo, d'audio et de métadonnées. Il n'existe pas de service *spécifique* associé à la transmission continue en temps réel. Les configurations en temps réel via les commandes de service Web sont définies dans le service multimédia et ReceiverService.

12.1 Protocole de flux multimédia

12.1.1 Format de transport

Le protocole RTP (Real-Time Transport) est un protocole de transfert multimédia (voir 12.1.2). Les quatre paragraphes suivants décrivent le transfert de données RTP.

12.1.1.1 Transfert de données RTP via UDP

UDP présente la plus faible charge supplémentaire (overhead) et est capable de transférer des données en temps réel d'une manière efficace. Un dispositif doit prendre en charge le protocole RTP/UDP, et il convient qu'il prenne en charge la multidiffusion RTP/UDP.

12.1.1.2 RTP/TCP

En cas de perte de paquet au cours du transfert multimédia via UDP, la norme autorise le transfert de données RTP via TCP en tant que méthode alternative de transport de média. Cependant, un dispositif PEUT prendre en charge l'option basée sur RTP/TCP. Si le dispositif prend en charge le protocole RTP/TCP, ce protocole doit être conforme à la norme [RFC 4571] (Framing Real-time Transport Protocol and RTP Control Protocol [RTCP] Packets over Connection-Oriented Transport).

12.1.1.3 RTP/RTSP/TCP

Il convient que le dispositif prenne en charge le transfert multimédia en utilisant RTP/RTSP pour traverser un pare-feu à l'aide d'un tunnel RTSP. Ce protocole doit être conforme à la norme [RFC 2326], 10.12.

12.1.1.4 RTP/RTSP/HTTP/TCP

Le flux de données doit être envoyé via HTTP pour traverser un pare-feu. Un dispositif doit prendre en charge le transfert multimédia en utilisant RTP/RTSP/HTTP/TCP. De plus, si un dispositif prend en charge TLS1.0, le flux de données doit être envoyé ou reçu via HTTPS pour traverser un pare-feu, et un dispositif doit prendre en charge le transfert multimédia à l'aide de RTP/RTSP/HTTPS/TCP.

Ce protocole doit être conforme à la norme [RFC 2326] (RTSP, 12.2.1.1: Embedded [Interleaved] Binary Data).

Cette méthode de tunnellation doit également être conforme au QuickTime disponible auprès d'Apple Inc. Les parties obligatoires du document suivant doivent être mises en œuvre par un NVT.

<http://developer.apple.com/quicktime/icefloe/dispatch028.html>

12.1.2 Transport de support

12.1.2.1 RTP

Le protocole RTP permet le transfert en temps réel des flux multimédia entre deux points terminaux. Le protocole RTP prend en charge le reclassement, la stabilisation et la synchronisation multimédia. La présentation de l'en-tête RTP est telle qu'illustrée à la Figure 14.

Tous les flux multimédia transférés par le protocole RTP doivent être conformes aux normes [RFC 3550], [RFC 3551], [RFC 3984], [RFC 3016] et JPEG via RTP (voir 12.1.3).

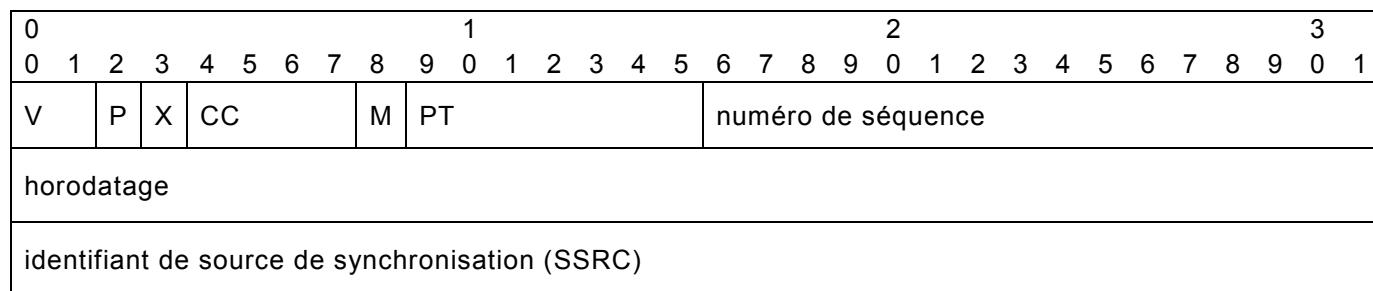


Figure 14 – En-tête RTP

IEC 2754/13

Un en-tête RTP doit être renseigné avec les valeurs du Tableau 190.

Tableau 190 – Valeur d'en-tête RTP

Champ d'en-tête	Valeur	Description
Version (V): 2 bits	2	
Padding (P): 1 bit	0/1	Si la charge utile comprend un octet de remplissage, il convient de lui attribuer la valeur "1"
Extension (X): 1 bit	0/1	Dépend de l'utilisation de l'extension d'en-tête RTP. La présente spécification définit deux scénarii dans lesquels une extension d'en-tête RTP peut être utilisée pour transmettre des informations supplémentaires: 1) "JPEG sur RTP" (voir 12.1.3). 2) Lecture (voir Article 21) Si l'extension d'en-tête est utilisée, une valeur doit être attribuée au bit Extension.
CSRC count (CC): 4 bits	0	
Marker (M): 1 bit	0/1	L'utilisation doit être conforme aux normes RFC correspondantes ([RFC 3984] pour H.264 Video, par exemple), à la présente Norme, comme "JPEG sur RTP", par exemple (voir 12.1.3) ou à la transmission RTP en continu de métadonnées (voir 12.1.2.1).
Payload type (PT): 7 bits	Voir [RFC 3551] Article 6.	
Sequence Number: 16 bits		Il convient que la valeur initiale du "numéro de séquence" soit aléatoire (imprévisible) afin de rendre plus difficiles les attaques à texte clair connu sur le chiffrement. Ce numéro est incrémenté d'une unité pour chaque paquet de données RTP envoyé.
timestamp: 32 bits		Il convient que la valeur initiale d'"horodatage" soit aléatoire (imprévisible) afin de rendre plus difficiles les attaques à texte clair connu sur chiffrement. Voir 12.1.2.2 pour plus de détails sur la synchronisation multimédia. L'utilisation de l'horodatage dépend du codec.
SSRC 32 bits		Source de synchronisation pour le flux de données. La présente spécification n'impose aucune restriction sur l'utilisation de ce champ.

RTP pour flux de métadonnées

Les flux de métadonnées sont également transportés par RTP. L'utilisation du type de charge utile, du marqueur et de l'horodatage pour l'en-tête RTP du flux de métadonnées est définie de la manière suivante:

- Un type de charge utile dynamique (96-127) doit être utilisé pour le type de charge utile assigné dans le processus d'une configuration de session RTSP.
- La valeur "1" doit être attribuée au bit de marqueur RTP lorsque le document XML est fermé.
- Il est RECOMMANDÉ d'utiliser un horodatage RTP représentant le temps de création du paquet RTP avec une fréquence d'horloge RTP de 90 000 Hz. Seuls des horodatages UTC doivent être utilisés dans le flux de métadonnées. La synchronisation de flux de données vidéo et audio est assurée à l'aide de RTCP.

La charge utile de métadonnées est un document XML avec le nœud racine `tt:MetaDataStream`. La taille du document XML n'est pas limitée. Lorsqu'un point de synchronisation (voir 11.18.2) est demandé pour le flux, le document XML précédent doit être fermé et un nouveau document initié. Il est RECOMMANDÉ d'initier de nouveaux documents XML au bout de 1 seconde, au plus tard. L'horodatage RTP du flux de métadonnées n'a pas de signification spécifique. Le flux de métadonnées multiplexe les métadonnées provenant de différentes sources. La présente spécification définit des paramètres fictifs pour la description de scène de l'analyse vidéo, le statut PTZ du contrôleur PTZ et les notifications de la configuration d'événement. Un dispositif peut choisir laquelle de ces parties il convient de multiplexer dans les métadonnées pendant la configuration multimédia (voir 11.2.10). Chaque partie peut apparaître plusieurs fois dans un ordre aléatoire dans le document. Une connexion de métadonnées peut être bidirectionnelle grâce au mécanisme de voie de retour (voir 12.3).

Le flux de métadonnées contient les éléments suivants:

- VideoAnalyticsStream;
- PTZStream;
- EventStream.

Les paramètres fictifs des différentes sources de métadonnées présentent la structure XML suivante:

```
<xs:complexType name="VideoAnalyticsStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="Frame" type="tt:Frame"/>
    ...
  </xs:choice>
</xs:complexType>

<xs:complexType name="PTZStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="PTZStatus"/>
    ...
  </xs:choice>
</xs:complexType>

<xs:complexType name="EventStream">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="wsnt:NotificationMessage"/>
    ...
  </xs:choice>
</xs:complexType>
```

Voici un exemple de document XML de métadonnées:

```

<?xml version="1.0" encoding="UTF-8"?>
<tt:MetaDataStream xmlns:tt="http://www.onvif.org/ver10/schema">
  <tt:VideoAnalytics>
    <tt:Frame UtcTime="2008-10-10T12:24:57.321">
      ...
    </tt:Frame>
    <tt:Frame UtcTime="2008-10-10T12:24:57.621">
      ...
    </tt:Frame>
  </tt:VideoAnalytics>
</tt:MetaDataStream>

<?xml version="1.0" encoding="UTF-8"?>
<tt:MetaDataStream xmlns:tt="http://www.onvif.org/ver10/schema">
  <tt:Event>
    <wsnt:NotificationMessage>
      <wsnt:Message>
        <tt:Message UtcTime= "2008-10-10T12:24:57.628">
          ...
        </tt:Message>
      </wsnt:Message>
    </wsnt:NotificationMessage>
  </tt:Event>
</tt:MetaDataStream>

```

12.1.2.2 RTCP

Le protocole RTCP produit un retour sur la qualité du service fourni par RTP et la synchronisation de différents flux multimédia. Le protocole RTCP doit être conforme à la norme [RFC 3550]. Le protocole utilise des rapports expéditeur (SR) et des rapports de récepteur (RR) pour acheminer des informations entre le serveur et le client comme illustré à la Figure 15.

Pour une demande de retour, il convient que les normes [RFC 4585] et [RFC 5104] soient prises en charge.

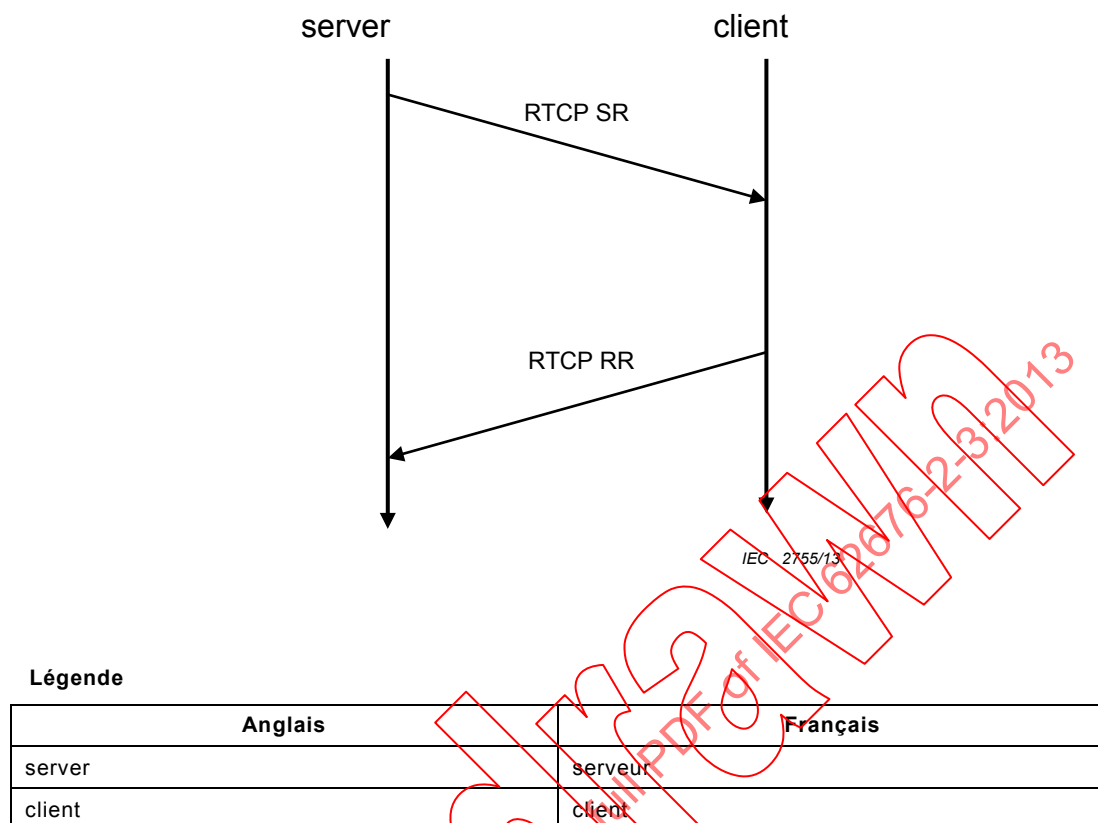


Figure 15 – Séquence RTCP

Synchronisation multimédia

Un client PEUT recevoir des flux audio et vidéo simultanément depuis plusieurs dispositifs. Dans ce cas, chaque flux utilise une horloge différente (de l'acquisition de données à la réception de paquet). Les rapports RTCP SR (Sender Report) sont utilisés pour synchroniser différents flux multimédia. Les RTCP SR doivent satisfaire à la norme [RFC 3550].

Le paquet RTCP Sender Report (SR) contient des champs pour l'horodatage RTP et pour un horodatage en temps absolu (date et heure absolues, NTP [Network Time Protocol, protocole de temps réseau] 64 bits). Voir Figure 16.

Un dispositif doit prendre en charge RTCP SR pour la synchronisation multimédia. Il convient que le client utilise RTCP pour la synchronisation multimédia.

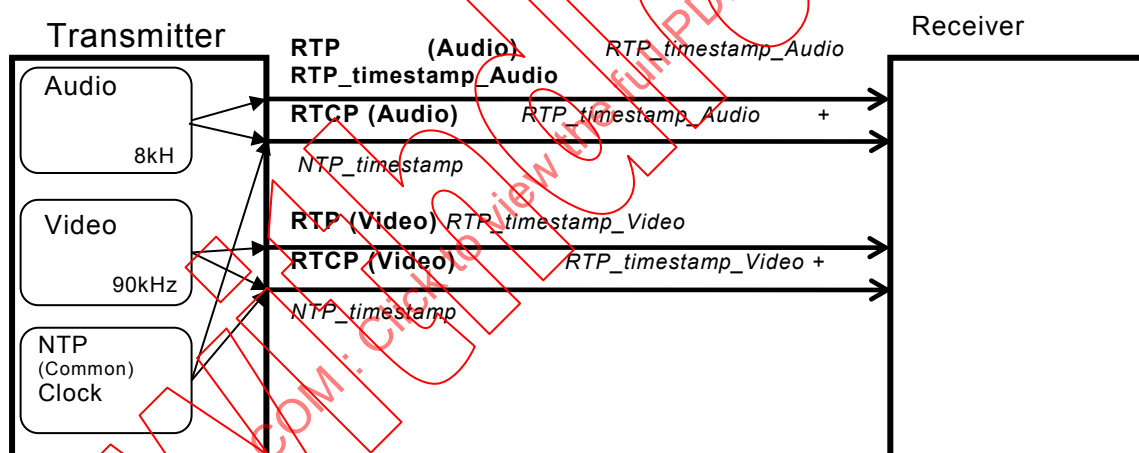
[illegible]

Figure 16 – RTCP Sender Report

IEC 2756/13

Il convient que l'horloge absolue soit commune dans le dispositif et il convient de déterminer correctement chaque valeur d'horodatage. Le client peut synchroniser différents flux multimédia avec la temporisation appropriée reposant sur les horodatages d'horloge RTP et d'horloge absolue (voir Figure 17).

En présence de plusieurs dispositifs, il convient que l'horodatage NTP soit commun à chacun d'eux et que le serveur NTP soit requis dans le système.



IEC 2757/13

Légende

Anglais	Français
Transmitter	Émetteur
Video	Vidéo
NTP (common) clock	Horloge NTP (commune)
Receiver	récepteur

Figure 17 – Synchronisation multimédia

12.1.3 Point de synchronisation

Les points de synchronisation permettent aux clients de décoder et d'utiliser correctement toutes les données après le point de synchronisation. Un point de synchronisation PEUT être

³ Le client peut obtenir des informations sur la "disponibilité de serveur NTP" à partir des dispositifs grâce à la commande GetNTP. Voir 8.2.5.

demandé par un client en cas d'erreur de décodeur (suite à la perte de paquet, par exemple) pour forcer le dispositif à ajouter une trame I dès que possible ou pour demander le PTZ ou l'état des événements en cours.

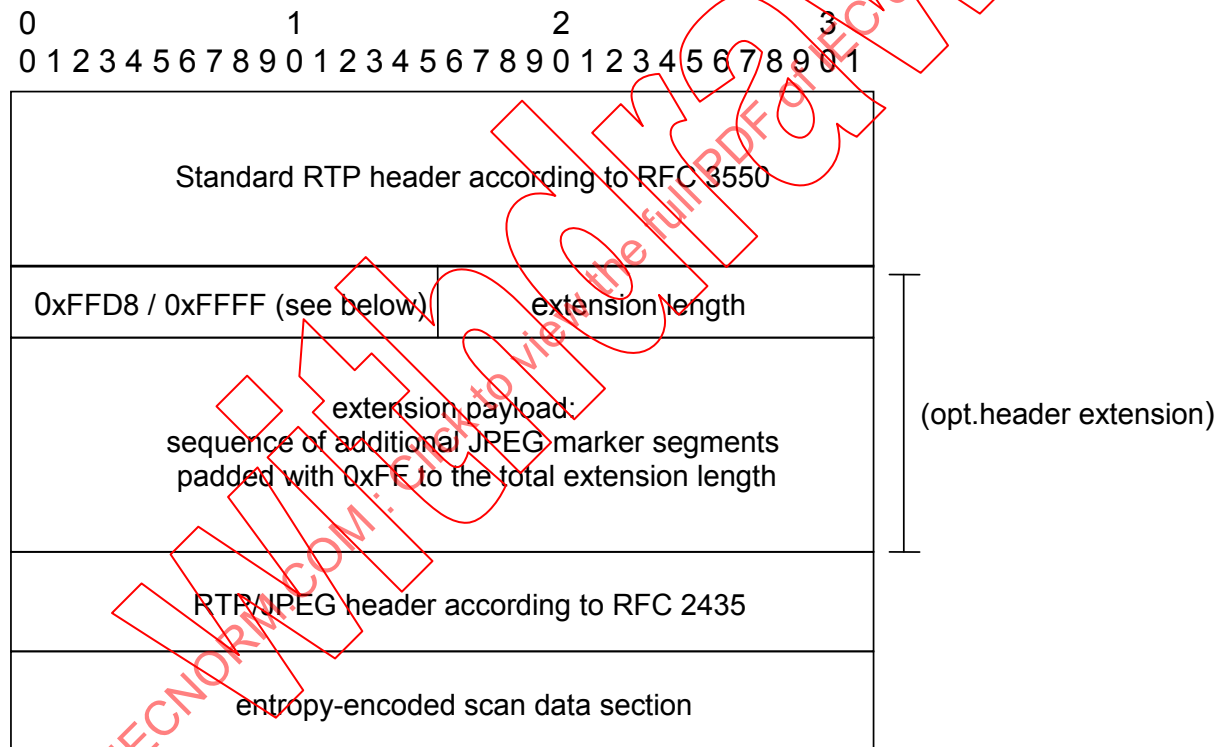
Outre les méthodes basées sur le Service Web (voir 11.18.2 et 15.6) qu'un dispositif doit prendre en charge, la présente Norme RECOMMANDE l'utilisation de messages PLI décrits dans la norme [RFC 4585] pour demander un point de synchronisation.

12.1.4 JPEG via RTP

12.1.4.1 Structure globale de paquet

La syntaxe de transmission des flux JPEG est conforme à la norme [RFC 2435]. Elle permet d'incorporer des données supplémentaires, au-delà des limitations de la norme [RFC 2435], en utilisant une extension d'en-tête RTP facultative (voir ci-dessous) avec certains des paquets RTP. Toutefois, cette option modifie la sémantique exacte des trames contenant ce type de paquet.

Le format global du paquet RTP JPEG est présenté à la Figure 18.



Anglais	Français
Standard RTP header according to RFC 3550	En-tête RTP normalisé conforme à la norme RFC 3550
(see below)	(voir ci-dessous)
extension length	longueur d'extension
(opt.header extension)	(extension opt.header)
extension payload: sequence of additional JPEG marker segments padded with 0xFF to the total extension length	charge utile d'extension: segments de marqueur JPEG remplis de 0xFF jusqu'à la longueur d'extension totale
RTP/JPEG header according to RFC 2435	En-tête RTP/JPEG conforme à la norme RFC 2435
entropy-encoded scan data section	section de données de balayage codées par entropie

Figure 18 – Structure de paquet RTP/JPEG (seul le contenu type est présenté pour la charge utile d'extension)

Afin de distinguer une extension d'en-tête RTP facultative des autres extensions d'en-tête possibles, les 16 premiers bits (les deux premiers octets de l'en-tête d'extension à quatre octets) d'un RTP doivent avoir la valeur 0xFFD8 (marqueur SOI JPEG) pour le paquet initial et 0xFFFF pour les autres paquets RTP dans une trame.

Comme requis par la norme [RFC 3550], la présence de l'extension d'en-tête facultative doit être signalée par le bit X de l'en-tête RTP. Le champ de longueur d'extension dans l'extension d'en-tête compte le nombre d'éléments 32 bits suivants en tant que charge utile d'extension. Par exemple, un champ de longueur zéro suivant l'en-tête d'extension de 32 bits représente une extension d'en-tête vide.

La section de données de balayage codées par entropie peut être absente de tous les paquets RTP. Toutefois, un en-tête RTP/JPEG complet doit être présent dans le paquet initial de chaque trame et de tous les paquets contenant une section de données de balayage codées par entropie, sinon il peut être absent.

Il convient d'utiliser le champ de décalage de fragment dans l'en-tête RTP/JPEG, conformément à la norme [RFC 2435], comme si aucune extension d'en-tête n'était présente. De plus, si un paquet ne contient pas de segment de données de balayage codées par entropie, mais qu'il contient une extension d'en-tête, le champ de décalage de fragment ne doit pas être nul si des paquets contenant une section de données de balayage codées par entropie pour la même trame ont été transmis. Si le paquet initial d'une trame ne contient pas d'extension d'en-tête, conformément à la présente Norme, son champ de décalage de fragment doit être nul, sinon il convient qu'il le soit. Tous les paquets comprenant un en-tête RTP/JPEG avec un décalage de fragment nul et une valeur Q comprise entre 128 et 255 doivent contenir un en-tête de table de quantification conformément à 3.1.8 de la norme [RFC 2435], les autres paquets ne doivent PAS contenir cet en-tête.

12.1.4.2 Spécification de décodage logique

Pour la spécification de décodage, il est supposé que l'ordre des paquets originaux dans le flux RTP a été restauré conformément à la numérotation de séquence RTP.

Si le paquet initial d'une trame ne contient pas d'extension d'en-tête RTP comme spécifié ci-dessus, les décodeurs doivent générer l'en-tête de balayage complet et effectuer le décodage comme spécifié par la norme [RFC 2435]. Les sections de données de balayage et les charges utiles d'une extension d'en-tête conforme à la présente spécification, jusqu'au paquet RTP suivant inclus avec son bit de marqueur défini, doivent être concaténées au fur et à mesure de leur apparition dans le flux, en ignorant leurs valeurs de décalage de fragment.

Sinon (au moins une extension d'en-tête vide comme spécifié ci-dessus est présente dans le paquet initial d'une trame), les règles suivantes s'appliquent pour chaque trame de ce type:

- Si le paquet initial d'une trame ne contient pas de segment de données de balayage codées par entropie, mais contient une extension d'en-tête comme spécifié ci-dessus, les décodeurs doivent concaténer sa charge utile d'extension d'en-tête avec une/des charge(s) utile(s) d'extension d'en-tête (éventuellement vide(s) ou inexistante(s)) conforme à la présente spécification des paquets suivants jusqu'au premier paquet inclus avec le bit marqueur RTP défini ou contenant un segment de données de balayage codées par entropie.
- La charge utile d'extension d'en-tête (séquence) RTP initiale concaténée doit être logiquement préfixée avec un marqueur SOI JPEG (0xFFD8).
- Si la valeur Q de l'en-tête de balayage RTP/JPEG dans le paquet initial d'une trame n'est pas nulle, les tables de quantification doivent être pré-initialisées conformément aux règles de la norme [RFC 2435]. Si Q est égal à zéro, les tables de quantification doivent être copiées à partir de la trame précédente, permettant aux marqueurs DQT présents dans cette charge utile d'extension d'en-tête (séquence) initiale de les remplacer.
- S'il s'agit de la trame initiale d'une séquence, les tables de Huffman doivent être pré-initialisées conformément à la norme [RFC 2435]. Les tables de Huffman de toutes les trames suivantes doivent être copiées à partir de la trame précédente, afin de pouvoir remplacer les trames par des marqueurs DHT dans la charge utile d'extension d'en-tête (séquence) initiale.
- Si la charge utile d'extension d'en-tête (séquence) RTP initiale ne fournit aucun marqueur DRI, mais que l'en-tête RTP/JPEG du paquet initial d'une trame contient un marqueur de redémarrage RTP/JPEG, un marqueur DRI correspondant aux règles de la norme [RFC 2435] doit être ajouté à la charge utile d'extension d'en-tête (séquence) initiale. Sinon, si l'extension d'en-tête (séquence) RTP initiale fournit un marqueur DRI, le marqueur doit être prioritaire sur tout autre marqueur de redémarrage RTP/JPEG conformément à la norme [RFC 2435] pour la même trame. Toutefois, pour la compatibilité avec les décodeurs conformes à la norme [RFC 2435] uniquement, il convient normalement que les codeurs utilisent un marqueur de redémarrage RTP/JPEG avec des valeurs cohérentes, si des intervalles de redémarrage sont à utiliser.
- Les marqueurs DRI ne doivent PAS être déduits des trames précédentes.
- Si la charge utile d'extension d'en-tête (séquence) RTP initiale ne fournit aucun marqueur SOF, qui sinon est prioritaire, un marqueur SOF doit lui être ajouté avec les valeurs suivantes:
 - Si les champs de largeur et de hauteur de l'en-tête RTP/JPEG sont nuls, le marqueur SOF de la trame précédente doit être utilisé.
 - Sinon, il doit être déduit conformément aux règles de la norme [RFC 2435].

Toutefois, tant que la taille d'image (arrondie par excès) est dans la plage spécifiée dans la norme [RFC 2435], il convient que les codeurs spécifient une taille d'image dans l'en-tête RTP/JPEG cohérente avec les valeurs d'un en-tête SOF supplémentaire.

- Si la charge utile d'extension d'en-tête (séquence) initiale ne fournit aucun marqueur SOS, un marqueur correspondant doit être déduit conformément à la norme [RFC 2435] et ajouté à celle-ci, sinon le marqueur SOS de l'extension est prioritaire.

Un marqueur SOS ne doit PAS être déduit de trames précédentes.

Si le marqueur SOS est présent et qu'il n'est pas suivi par des données de balayage codées par entropie dans l'extension, il doit être le marqueur final dans la charge utile (séquence) d'extension initiale d'une trame. Le remplissage nécessaire avec des octets 0xFF ne doit PAS suivre ce marqueur mais PEUT le précéder.

- Les données de balayage codées par entropie restantes et les charges utiles d'extensions d'en-tête doivent être logiquement ajoutées dans le même ordre que celui de leur apparition dans le flux RTP jusqu'à la fin de la trame, comme indiqué par le bit marqueur RTP. Un marqueur EOI final doit également être ajouté s'il n'est pas encore présent dans la séquence logique de cette trame.

Pour chaque trame, la séquence obtenue jusqu'au premier marqueur EOI inclus (éventuellement ajouté) doit être un flux JPEG valide (éventuellement abrégé), générant une image complète issue du processus de décodage de cette trame. La définition des données qui suivent ce premier marqueur EOI pour chaque trame est hors du domaine d'application de la présente norme.

12.1.4.3 Espaces colorimétriques et facteurs d'échantillonnage pris en charge

Il convient qu'un émetteur utilise uniquement l'échelle de gris et l'espace colorimétrique YCbCr. Un client doit prendre en charge l'échelle de gris et YCbCr.

Les facteurs d'échantillonnage pour YCbCr doivent correspondre aux valeurs prises en charge par la norme [RFC 2435]. Par exemple, un facteur d'échantillonnage de 4:2:0 (préférentiel) ou 4:2:2.

12.1.4.4 Gestion du format d'affichage des pixels

Le format d'affichage des pixels des fichiers JPEG peut être spécifié dans le marqueur JFIF. Si le format d'affichage des pixels est différent du facteur normalisé 1:1 et 1:2 conformément à la norme [RFC 2435], il convient que ce marqueur soit transmis dans la charge utile d'extension d'en-tête (séquence) initiale de chaque trame pour spécifier le format d'affichage des pixels (pour des données entrelacées à base de champ).

12.1.4.5 Gestion de la vidéo entrelacée

La vidéo entrelacée est codée par deux champs indépendants et signalée comme il est spécifié dans la norme [RFC 2435] dans l'en-tête RTP/JPEG.

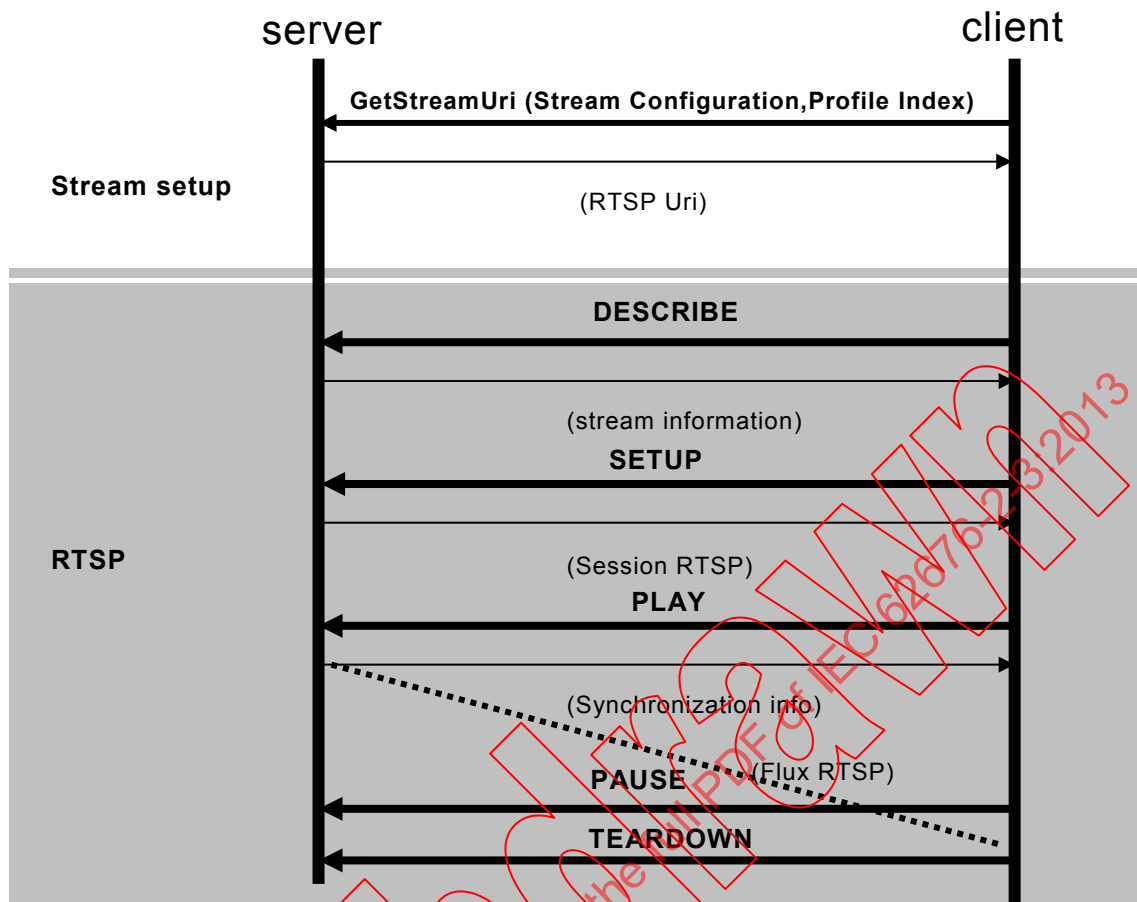
Les deux champs doivent utiliser les mêmes espaces colorimétriques, facteurs d'échantillonnage et format d'affichage des pixels.

Il convient de ne PAS utiliser le codage entrelacé si la trame a été initialement balayée progressivement.

12.2 Protocole de contrôle multimédia

12.2.1 Contrôle de flux

Le flux multimédia est contrôlé à l'aide du protocole défini dans l'URI. L'ensemble du contrôle de flux est illustré à la Figure 19. L'URI est retourné en réponse à la commande GetStreamUri définie en 11.15.2.



Légende

IEC 2759/13

Anglais	Français
Server	Serveur
(Stream Configuration, Profile Index)	(configuration de flux, index de profil)
Stream setup	Configuration de flux
Stream information	Informations de flux
RTSP session	Session RTSP
Synchronization info	Infos de synchronisation
RTSP stream	Flux RTSP

Figure 19 – Contrôle de flux

12.2.1.1 RTSP

Tous les dispositifs et clients doivent prendre en charge RTSP ([RFC 2326]) pour l'initiation de session et le contrôle de la lecture (voir Tableau 191). RTSP doit utiliser TCP comme protocole de transport, le port TCP par défaut du trafic RTSP étant le port 554. Le protocole SDP (Session Description Protocol) doit être utilisé pour fournir des informations relatives au flux multimédia et SDP doit satisfaire à la norme [RFC 4566].

Tableau 191 – Méthodes RTSP

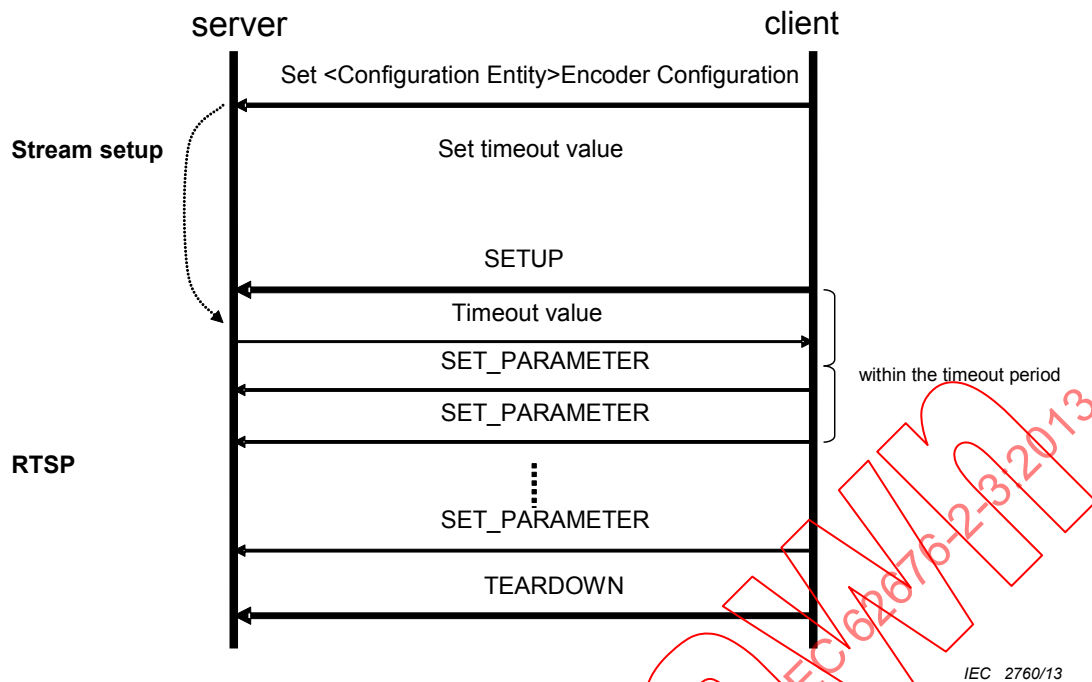
Méthode	Sens	SPEC	Description
OPTIONS	R->T T->R	M X	Requis pour obtenir une fonctionnalité de méthode optionnelle et autoriser différentes versions dans le futur.
DESCRIBE	R->T	M	Requis pour extraire des paramètres multimédia dans le profil désigné.
ANNOUNCE	R->T T->R	X	
SETUP	R->T	M	Requis pour définir des paramètres de session.
PLAY	R->T	M	Requis pour démarrer un flux multimédia.
PAUSE	R->T	O	Requis pour arrêter temporairement un flux multimédia. Afin de gérer des flux multiples dans un réseau à bande passante étroite, par suspension du flux RTP, le trafic peut être correctement contrôlé en réduisant les données redondantes et les congestions du trafic réseau peuvent être évitées.
TEARDOWN	R->T	M	Requis pour libérer une session de média.
GET_PARAMETER	R->T T->R	O	
SET_PARAMETER	R->T T->R	O O	Méthode facultative permettant de maintenir une session RTSP active (direction R-T uniquement).
REDIRECT	T->R	X	
RECORD	R->T	X	
Légende: X: Non pris en charge, M: Obligatoire, O: Facultatif			

12.2.1.1.1 Méthode "keep-alive " pour session RTSP

Un client RTSP maintient la session RTSP active ("keep-alive") et empêche l'expiration du délai d'attente de session (voir [RFC 2326] 12.37). Cette spécification recommande les méthodes suivantes pour maintenir RTSP actif pour la transmission continue par diffusion individuelle ou multidiffusion.

- 1) Le client peut éventuellement définir le paramètre Timeout (en secondes) en utilisant la commande `Set<configurationEntity>EncoderConfiguration` définie en 11.2. Sinon, la valeur par défaut "60" est utilisée.
- 2) Dans toutes les réponses RTSP SETUP, il convient qu'un émetteur inclue la valeur de Timeout conformément à la norme [RFC 2326], 12.37, et qu'il utilise la valeur de Timeout pour le maintien "keep-alive".
- 3) Pour maintenir la session RTSP active, un client doit appeler le serveur RTSP à l'aide d'une méthode RTSP ou envoyer des rapports de récepteur RTCP. SET_PARAMETER est la méthode RTSP RECOMMANDÉE.

La Figure 20 représente un exemple typique du contrôle de flux.



IEC 2760/13

Légende

Anglais	Français
server	serveur
Stream setup	Configuration de flux
Set timeout value	Définir la valeur du délai d'attente
Timeout value	Valeur du délai d'attente
within the timeout period	dans la période de délai d'attente

Figure 20 – Méthode "Keep Alive"

12.2.1.1.2 Synchronisation audio et vidéo RTSP

Pour que les clients puissent immédiatement commencer la synchronisation des flux audio et vidéo, puis calculer les horodatages TUC absolus des paquets entrants aux fins d'enregistrement, il convient qu'un émetteur inclue les champs d'en-tête suivants dans la réponse RTSP PLAY:

- Range ([RFC 2326], 12.29). Cela doit inclure une heure de début en unités d'horloge ([RFC 2326] 3.7), pas en unités SMPTE ou NPT.
- RTP-Info ([RFC 2326], 12.33). Cela doit inclure une valeur rtpime correspondant à l'heure de début indiquée dans l'en-tête Range.

Exemple:

```

client->>server:    PLAY rtsp://example.com/onvif_camera/video RTSP/1.0
                   CSeq: 4
                   Range: npt=now-
                   Session: 12345678

server->>client:    RTSP/1.0 200 OK
                   CSeq: 4
                   Session: 12345678
                   Range: 20100217T143720.257Z-
                   RTP-Info: url=rtsp://example.com/onvif_camera/video;
seq=1234;rtptime=3450012
    
```

12.2.1.1.3 Session RTSP pour un flux de métadonnées

Dans le cas d'un flux de métadonnées, il convient que la description SDP "application" soit utilisée dans la réponse DESCRIBE pour le type de média et il convient que "vnd.onvif.metadata" soit utilisé pour coder un nom.

Voici un exemple d'échange de message RTSP DESCRIBE entre un serveur RTSP (serveur) et un client an RTSP (client):

```
client->server:    DESCRIBE rtsp://example.com/onvif_camera RTSP/1.0
                  CSeq: 1

server->client:    RTSP/1.0 200 OK
                  CSeq: 1
                  Content-Type: application/sdp
                  Content-Length: XXX
                  v=0
                  o=- 2890844256 2890842807 IN IP4 172.16.2.93
                  s=RTSP Session
                  m=audio 0 RTP/AVP 0
                  a=control:rtsp://example.com/onvif_camera /audio
                  m=video 0 RTP/AVP 26
                  a=control:rtsp://example.com/onvif_camera /video
                  m=application 0 RTP/AVP 107
                  a=control:rtsp://example.com/onvif_camera/metadata
                  a=recvonly
                  a=rtpmap
                  a=rtpmap:107 vnd.onvif.metadata/90000
```

12.2.1.1.4 Exemple de message RTSP

Cet exemple illustre le transfert de message entre un client RTSP (client) et un serveur RTSP (serveur). Le client demande un flux audio et vidéo depuis le dispositif. L'URI de flux "rtsp://example.com/onvif_camera" peut être extrait grâce à la commande GetStreamUri. Voir 11.15.2.

```
client->server:    DESCRIBE                                rtsp://example.com/onvif_camera
RTSP/1.0

                  CSeq: 1

server->client:    RTSP/1.0 200 OK
                  CSeq: 1
                  Content-Type: application/sdp
                  Content-Length: XXX
                  v=0
                  o=- 2890844256 2890842807 IN IP4 172.16.2.93
                  s=RTSP Session
                  m=audio 0 RTP/AVP 0

                  a=control:rtsp://example.com/onvif_camera/audio
                  m=video 0 RTP/AVP 26

                  a=control:rtsp://example.com/onvif_camera/video

client->server:    SETUP rtsp://example.com/onvif_camera/audio RTSP/1.0
                  CSeq: 2
                  Transport:    RTP/AVP;unicast;client_port=8002-
8003
```

```

server->client:      RTSP/1.0 200 OK
                      CSeq: 2
                      Transport:    RTP/AVP;unicast;client_port=8002-
8003;
                      server_port=9004-9005
                      Session: 12345678; timeout=60

client->server:      SETUP rtsp://example.com/onvif_camera/video RTSP/1.0
                      CSeq: 3
                      Transport:    RTP/AVP;unicast;client_port=8004-
8005
                      Session: 12345678

server->client:      RTSP/1.0 200 OK
                      CSeq: 3
                      Transport:    RTP/AVP;unicast;client_port=8004-
8005;
                      server_port=9006-9007
                      Session: 12345678; timeout=60

client->server:      PLAY rtsp://example.com/onvif_camera RTSP/1.0
                      CSeq: 4
                      Range: npt=now-
                      Session: 12345678

server->client:      RTSP/1.0 200 OK
                      CSeq: 4
                      Session: 12345678
RTP-Info:            url=rtsp://example.com/onvif_camera/video;
seq=1234;rtptime=3450012, url=rtsp://example.com/onvif_camera/audio;
seq=22434;rtptime=1234566

client->server:      TEARDOWN rtsp://example.com/onvif_camera RTSP/1.0
                      CSeq: 5
                      Session: 12345678

server->client:      RTSP/1.0 200 OK
                      CSeq: 5
                      Session: 12345678

```

12.2.1.2 RTSP via HTTP

Le protocole RTSP via HTTP/HTTPS doit être pris en charge pour traverser un pare-feu. Voir 12.1.1.4 RTP/RTSP/HTTP/TCP.

12.3 Connexion de la voie de retour

Le présent paragraphe explique comment une connexion bidirectionnelle peut être établie entre un client et un serveur. La gestion de connexion de voie de retour est assurée à l'aide du protocole RTSP [RFC 2326]. Un mécanisme est donc présenté, indiquant qu'un client souhaite générer une connexion de voie de retour. RTSP offre des balises de caractéristique dotées de ces fonctions.

Un dispositif qui prend en charge les connexions bidirectionnelles (les connexions audio ou de métadonnées, par exemple) doit prendre en charge les extensions RTSP présentées.

12.3.1 Balise RTSP Require

La norme RTSP [RFC 2326] peut être étendue à l'aide d'objets d'en-tête supplémentaires. C'est la raison pour laquelle une balise Require est introduite pour gérer les ajouts de fonctionnalités particuliers (voir [RFC 2326], 1.5 Extending Rtsp et 12.32 Require).

La balise Require permet de déterminer la prise en charge de cette caractéristique. L'en-tête doit être inclus dans une demande lorsqu'il est demandé au serveur de comprendre cette caractéristique pour exécuter correctement la demande.

Un dispositif qui prend en charge la voie de retour doit comprendre la balise correspondante:

- www.onvif.org/ver20/backchannel

Un client RTSP qui souhaite générer une connexion RTSP avec une voie de retour de données doit inclure l'en-tête Require dans ses demandes.

12.3.2 Configuration des connexions pour une connexion bidirectionnelle

Un client doit inclure la balise de caractéristique dans sa demande DESCRIBE pour indiquer qu'une connexion de données bidirectionnelle doit être établie.

Un serveur qui comprend cette balise Require doit inclure un flux multimédia supplémentaire dans son fichier SDP tel que configuré dans son profil multimédia.

Un serveur RTSP qui ne comprend pas la balise de caractéristiques de voie de retour ou qui ne prend pas en charge les connexions de données bidirectionnelles doit répondre par un code d'erreur 551 *Option not supported* (Option non prise en charge) conformément à la norme RTSP. Le client peut ensuite tenter d'établir une connexion RTSP sans voie de retour.

Un fichier SDP est utilisé pour décrire la session. Le serveur doit inclure l'attribut a=sendonly ou a=recvonly dans chaque section multimédia du fichier SDP pour indiquer dans quelle direction vont être envoyées les données multimédia.

Le serveur doit répertorier tous les codecs de décodage pris en charge en tant que section multimédia, et le client choisit celui à utiliser.

12.3.2.1 Exemple 1: Serveur sans prise en charge de voie de retour

```
Client - Server: DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                  CSeq: 1
                  User-Agent: ONVIF Rtsp client
                  Accept: application/sdp
                  Require: www.onvif.org/ver20/backchannel

Server - Client:  RTSP/1.0 551 Option not supported
                  CSeq: 1
                  Unsupported: www.onvif.org/ver20/backchannel
```

12.3.2.2 Exemple 2: Serveur avec prise en charge de voie de retour ONVIF

```
Client - Server: DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                  CSeq: 1
                  User-Agent: ONVIF Rtsp client
                  Accept: application/sdp
                  Require: www.onvif.org/ver20/backchannel

Server - Client:  RTSP/1.0 200 OK
                  CSeq: 1
                  Content-Type: application/sdp
                  Content-Length: xxx

                  v=0
o= 2890842807 IN IP4 192.168.0.1
s=RTSP Session with audiobackchannel
m=video 0 RTP/AVP 26
a=control:rtsp://192.168.0.1/video
a=recvonly
```

```

                                m=audio 0 RTP/AVP 0
a=control:rtsp://192.168.0.1/audio
a=recvonly
m=audio 0 RTP/AVP 0
a=control:rtsp://192.168.0.1/audioback
a=rtpmap:0 PCMU/8000
a=sendonly

```

Ce fichier SDP décrit de manière exhaustive la session RTSP. Le serveur donne au client ses URL de contrôle pour configurer les flux.

A l'étape suivante, le client peut configurer les sessions:

```

Client - Server:      SETUP rtsp://192.168.0.1/video RTSP/1.0
                        CSeq: 2
                        Transport: RTP/AVP;unicast;client_port=4588-
4589

Server - Client:      RTSP/1.0 200 OK
                        CSeq: 2
Session: 123124;timeout=60
Transport:RTP/AVP;unicast;client_port=4588-4589; server_port=6256-6257

Client - Server:      SETUP rtsp://192.168.0.1/audio RTSP/1.0
                        CSeq: 3
Session: 123124
Transport: RTP/AVP;unicast;client_port=4578-4579

Server - Client:      RTSP/1.0 200 OK
                        CSeq: 3
Session: 123124;timeout=60
Transport:RTP/AVP;unicast;client_port=4578-4579; server_port=6276-6277

Client - Server:      SETUP rtsp://192.168.0.1/audioback RTSP/1.0
                        CSeq: 4
Session: 123124
Transport: RTP/AVP;unicast;client_port=6296-6297
Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 200 OK
                        CSeq: 4
Session: 123124;timeout=60
Transport:RTP/AVP;unicast;client_port=6296-6297; server_port=2346-2347

```

La troisième demande de configuration établit la connexion de voie de retour audio.

À l'étape suivante, le client démarre la session en envoyant une demande PLAY.

```

Client - Server:      PLAY rtsp://192.168.0.1 RTSP/1.0
                        CSeq: 5
                        Session: 123124
Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 200 OK
                        CSeq: 5
                        Session: 123124;timeout=60

```

Après avoir reçu la réponse OK à cette demande PLAY, le client PEUT commencer à envoyer les données audio au serveur. Il ne doit pas commencer l'envoi des données au serveur avant d'avoir reçu la réponse.

L'en-tête Require indique qu'une interprétation particulière de la commande PLAY est nécessaire. La commande couvre le début du flux audio et vidéo entre le NVT et le client, et le début de la connexion audio du client au serveur.

Pour mettre fin à la session, le client envoie une requête TEARDOWN.

```
Client - NVT:          TEARDOWN rtsp://192.168.0.1 RTSP/1.0
                      CSeq: 6
                      Session: 123124
Require: www.onvif.org/ver20/backchannel

NVT - Client:         RTSP/1.0 200 OK
                      CSeq: 6
                      Session: 123124
```

12.3.3 Transmission continue en multidiffusion

Si le client tente d'envoyer ses données en multidiffusion, il utilise le paramètre de transport de la demande SETUP pour indiquer au serveur l'adresse et le port de multidiffusion.

Exemple: Configuration multidiffusion

```
Client - Server:       SETUP rtsp://192.168.0.1/audioback RTSP/1.0
                      CSeq: 4
Session: 123124
Transport: RTP/AVP;multicast;destination=224.2.1.1;port=60000-60001;ttl=128
Require: www.onvif.org/ver20/backchannel

Server - Client:      RTSP/1.0 200 OK
                      CSeq: 4
Session: 123124;timeout=60
Transport: RTP/AVP;multicast;destination=224.2.1.1;port=60000-60001;ttl=128;mode="PLAY"
```

12.4 Gestion des erreurs

Les erreurs de protocole RTSP et HTTP sont classées dans différentes catégories (par exemple, les codes de statut 1xx, 2xx, 3xx, 4xx et 5xx respectivement). Le dispositif et le client doivent prendre en charge et gérer ces codes de statut. Pour les définitions de code de statut RTSP, voir le document [RFC 2326], 11.0. Pour les définitions de code de statut HTTP, voir HTTP/1.1 [RFC 2616], 10.0.

13 Configuration du récepteur

Ce service offre des commandes de gestion des objets Receiver, qui sont utilisés pour recevoir des flux multimédia provenant d'autres dispositifs. Un objet Receiver contient des informations relatives à la configuration du flux, au mode du récepteur et à l'URI du flux (MediaUri). Un dispositif doit au moins prendre en charge les URI multimédia de 128 octets. La fonctionnalité Receiver - MaximumRTSPURILength indique la longueur maximale prise en charge par le dispositif. Le service de récepteur doit être mis en œuvre par les dispositifs pouvant recevoir des flux multimédia.

L'adresse IP ou DNS de l'URI émettrice donnée au récepteur est l'adresse que le dispositif qui héberge le service de récepteur utilise pour accéder au dispositif émetteur. Par exemple, s'il faut que le client communique par l'intermédiaire d'un routeur NAT pour accéder à l'émetteur et au récepteur, l'adresse de l'émetteur que le client donne au récepteur (dans ce cas, une adresse de réseau local) peut être différente de celle que le client utilise pour accéder à l'émetteur (dans ce cas, une adresse de réseau externe).

Un dispositif doit prendre en charge le transfert RTP via RTP (voir 12.1.1.1) et le transfert RTP via RTSP/HTTP/TCP (voir 12.1.1.4). Un dispositif peut prendre en charge d'autres protocoles de transfert RTP et doit indiquer celui qu'il prend en charge avec les capacités appropriées (voir la catégorie Récepteur du Tableau 11).

13.1 Persistance

Tous les objets créés dans le service de récepteur doivent être persistants, c'est-à-dire qu'ils doivent résister à un cycle d'alimentation. De même, toutes les données de configuration des objets doivent être persistantes.

13.2 Modes du récepteur

Un récepteur peut fonctionner en trois modes distincts:

- Toujours connecté – le récepteur tente de maintenir une connexion permanente au point terminal configuré;
- Jamais connecté – le récepteur ne tente pas de se connecter;
- Connexion automatique – le récepteur se connecte à la demande des utilisateurs des flux multimédia.

13.3 Commandes du récepteur

Le présent paragraphe décrit les commandes proposées par le service de récepteur.

13.3.1 Obtention des récepteurs

Cette opération permet de répertorier tous les récepteurs qui existent sur le dispositif. Le service de récepteur doit prendre en charge cette commande (voir Tableau 192).

Tableau 192 – Commande GetReceivers

GetReceivers	
Nom du message	Description
GetReceiversRequest	Ce message est vide.
GetReceiversResponse	Contient une liste des récepteurs. tt:Receiver Receivers [0][non limité]
Codes de défaut	Description
Pas de codes de défaut spécifiques.	

13.3.2 Obtention du récepteur

Cette opération permet d'extraire les détails d'un récepteur spécifique dont le jeton est connu du client. Le service de récepteur doit prendre en charge cette commande (voir Tableau 193).

Tableau 193 – Commande GetReceiver

GetReceiver	
Nom du message	Description
GetReceiverRequest	Contient le jeton du récepteur demandé. tt:ReferenceToken ReceiverToken [1][1]
GetReceiverResponse	Contient les détails du récepteur demandé. tt:Receiver Receiver [1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	Le récepteur indiqué par ReceiverToken n'existe pas.

13.3.3 Création de récepteur

Cette opération permet de créer un récepteur. Le service de récepteur doit prendre en charge cette commande (voir Tableau 194).

Tableau 194 – Commande CreateReceiver

CreateReceiver	
Nom du message	Description
CreateReceiverRequest	Contient la configuration initiale du récepteur. tt:ReceiverConfiguration Configuration [1][1]
CreateReceiverResponse	Contient les détails du récepteur créé. tt:Receiver Receiver [1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup	La configuration spécifiée n'est pas valide.
env:Receiver ter:Action ter:MaxReceivers	Le nombre maximal de récepteurs pris en charge a été atteint.

13.3.4 Suppression de récepteur

Cette opération permet de supprimer un récepteur existant. Un récepteur NE PEUT PAS être supprimé s'il est en cours d'utilisation. Le service de récepteur doit prendre en charge cette commande (voir Tableau 195).

Tableau 195 – Commande DeleteReceiver

DeleteReceiver	
Nom du message	Description
DeleteReceiverRequest	Contient le jeton du récepteur à supprimer tt:ReferenceToken ReceiverToken [1][1]
DeleteReceiverResponse	Ce message est vide.
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	Le récepteur indiqué par ReceiverToken n'existe pas.
env: Receiver ter:Action ter:CannotDeleteReceiver	Le récepteur spécifié ne peut pas être supprimé (il est en cours d'utilisation, par exemple).

13.3.5 Configuration de récepteur

Cette opération permet de configurer un récepteur. Le service de récepteur doit prendre en charge cette commande (voir Tableau 196)

Tableau 196 – Commande ConfigureReceiver

ConfigureReceiver	
Nom du message	Description
ConfigureReceiverRequest	Contient le jeton du récepteur demandé et la nouvelle configuration. tt:ReferenceToken ReceiverToken [1][1] tt:ReceiverConfiguration Configuration [1][1]
ConfigureReceiverResponse	Ce message est vide.
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	Le récepteur indiqué par ReceiverToken n'existe pas.
env:Sender ter:InvalidArgVal ter:BadConfiguration	La configuration spécifiée n'est pas valide.

13.3.6 SetReceiverMode

Cette opération peut être utilisée pour définir le mode du récepteur indépendamment du reste de sa configuration. Le service de récepteur doit prendre en charge cette commande (voir Tableau 197).

Tableau 197 – Commande SetReceiverMode

SetReceiverMode	
Nom du message	Description
SetReceiverModeRequest	Contient le jeton du récepteur demandé et le nouveau mode. tt:ReferenceToken ReceiverToken [1][1] tt:ReceiverMode ReceiverMode [1][1]
SetReceiverModeResponse	Ce message est vide.
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	Le récepteur indiqué par ReceiverToken n'existe pas.

13.3.7 GetReceiverState

Cette opération permet de déterminer si le récepteur est actuellement déconnecté, connecté ou s'il tente de se connecter. Le service de récepteur doit prendre en charge cette commande (voir Tableau 198).

Tableau 198 – Commande GetReceiverState

GetReceiverState	
Nom du message	Description
GetReceiverStateRequest	Contient le jeton du récepteur demandé. tt:ReferenceToken ReceiverToken [1][1]
GetReceiverStateResponse	Contient le statut en cours du récepteur. tt:ReceiverState State [1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:UnknownToken	Le récepteur indiqué par ReceiverToken n'existe pas.

13.4 Événements

Le service de récepteur doit expédier les événements grâce au service d'événement. Il doit être capable de générer les événements figurant dans le présent paragraphe à chaque fois que la condition qui déclenche l'événement se produit.

13.4.1 ChangeState

À chaque fois qu'un récepteur change de statut, le dispositif doit expédier l'événement suivant:

```
Topic: tns1: Receiver/ChangeState
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="ReceiverToken" Type="tt:ReceiverToken"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItemDescription Name="NewState" Type="tt:ReceiverState"/>
    <tt:SimpleItemDescription Name="MediaUri" Type="tt:MediaUri" minOccurs="0"/>
  </tt>Data>
</tt:MessageDescription>
```

13.4.2 Échec de la connexion

Si un récepteur ne peut établir de connexion, le dispositif doit expédier l'événement suivant:

```
Topic: tns1: Receiver/ConnectionFailed
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="ReceiverToken" Type="tt:ReceiverToken"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItemDescription Name="MediaUri" Type="tt:MediaUri"/>
  </tt>Data>
</tt:MessageDescription>
```

13.5 Codes de défaut spécifiques au service

Le Tableau 199 présente les codes de défaut spécifiques au service d'affichage. De plus, chaque commande peut également générer un défaut générique (voir Tableau 6).

Les défauts spécifiques sont définis en tant que sous-code d'un défaut générique (voir 5.11.2.1). Le sous-code générique parent est le *sous-code* en haut de chaque ligne ci-dessous, le *sous-code* de défaut spécifique se trouvant en bas de la cellule.

Tableau 199 – Codes de défaut spécifiques au service

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
Env:Sender	ter:InvalidArgVal	Le récepteur n'existe pas.	Le récepteur indiqué par ReceiverToken n'existe pas.
	ter:UnknownToken		
Env:Sender	ter:Action	Le nombre maximal de récepteurs a été atteint.	Le nombre maximal de récepteurs pris en charge a été atteint.
	ter:MaxReceivers		
Env:Sender	ter:InvalidArgVal	StreamSetup n'est pas pris en charge.	La spécification de la partie StreamType ou Transport dans ReceiverConfiguration StreamSetup n'est pas prise en charge.
	ter:InvalidStreamSetup		
Env:Sender	ter:Action	Le récepteur ne peut pas être supprimé.	Le récepteur spécifié ne peut pas être supprimé (il est en cours d'utilisation, par exemple).
	ter:CannotDeleteReceiver		

14 Service d'affichage

Un dispositif d'affichage comporte un nombre fixe de sorties vidéo, chacune d'elles pouvant être associée à un moniteur. Un client peut demander les sorties vidéo du dispositif utilisant le service DeviceIO. Chacune de ces sorties est configurée selon présentation (un seul

affichage ou division d'écran, par exemple). La présentation définit un certain nombre de panneaux vidéo, chacun occupant une zone de l'affichage physique.

Un affichage vidéo en réseau PEUT également comporter un nombre fixe d'entrées et de sorties audio. Chacune de ces sorties PEUT être associée à un panneau. La liaison d'une entrée ou d'une sortie audio à un panneau met automatiquement en correspondance les flux audio et vidéo provenant d'un dispositif émetteur avec les sorties correctes. Le panneau contient également un pointeur vers un récepteur dans lequel sont stockées les informations nécessaires à la connexion du dispositif d'affichage à un émetteur.

Le service d'affichage offre des fonctions de configuration des panneaux et de description et modification de la présentation du dispositif d'affichage. Les présentations et fonctionnalités de codage possibles d'une sortie vidéo peuvent être demandées.

Un dispositif d'affichage doit prendre en charge le service d'affichage tel que défini en C.5.

14.1 Panneaux

Un panneau est une zone d'affichage du moniteur associée à une sortie vidéo. Un panneau contient un `PaneConfiguration` qui décrit les entités qui lui sont associées. `PaneConfiguration` comprend:

Pane Token: Identifiant unique dans le dispositif d'affichage.

Pane Name: Nom de configuration.

AudioOutputToken: Pointeur vers la sortie audio associée au panneau. Un client peut extraire les sorties audio disponibles d'un dispositif grâce à la commande `GetAudioOutputs` du service `DeviceIO`.

AudioSourceToken: Pointeur vers la source audio associée au panneau. La connexion audio entre un dispositif d'affichage et le NVT est établie à l'aide du mécanisme de voie de retour. Un client peut extraire les sources audio disponibles d'un dispositif grâce à la commande `GetAudioSources` du service `DeviceIO`.

AudioEncoderConfiguration: Configuration du codeur audio comprenant le codec, le débit binaire et le taux d'échantillonnage.

ReceiverToken: Pointeur vers un récepteur détenant les informations nécessaires à la réception de données provenant d'un émetteur. Ce récepteur peut être connecté, et l'affichage vidéo en réseau affiche les données reçues sur les sorties spécifiées. Un client peut extraire les récepteurs disponibles grâce à la commande `GetReceivers` du service de récepteur.

Il faut qu'un client configure le panneau en fonction de la connexion à établir en attribuant une valeur à `AudioOutput` et/ou `AudioSourceToken`. Si aucun jeton n'est défini, la session correspondante n'est pas établie.

La modification de `PaneConfiguration` ou des paramètres d'un récepteur référencé ne doit pas affecter la connexion RTSP. Si un client tente d'appliquer les nouveaux paramètres, il doit redémarrer la connexion RTSP.

La présentation du panneau (voir 14.2) de la sortie vidéo définit si un panneau est visible et à quel endroit il se trouve (position, taille). Le récepteur doit uniquement établir une connexion RTSP pour recevoir des données si le panneau est visible. Les modifications de présentation ne doivent PAS affecter les flux en cours d'exécution.

14.1.1 GetPaneConfigurations

Cette commande permet de répertorier tous les panneaux définis d'un dispositif pour une sortie vidéo spécifiée (que le panneau soit visible ou pas à un instant donné). Un dispositif d'affichage doit prendre en charge l'extraction de ses panneaux configurés grâce à cette commande (voir Tableau 200).

Tableau 200 – GetPaneConfigurations

GetPaneConfigurations		Demande-Réponse
Nom du message	Description	
GetPaneConfigurationsRequest	<i>L'élément VideoOutput spécifie la sortie vidéo dont les PaneConfigurations sont demandés.</i> tt:ReferenceToken VideoOutput[1][1]	
GetPaneConfigurationsResponse	<i>Contient une liste des panneaux définis du VideoOutput spécifié. Chaque VideoOutput contient au moins un PaneConfiguration.</i> tt:PaneConfiguration PaneConfiguration [1][non limité]	
Codes de défaut	Description	
env:Sender	<i>La sortie vidéo demandée n'existe pas.</i>	
ter:InvalidArgVal		
ter:NoVideoOutput		

14.1.2 GetPaneConfiguration

Si le jeton de panneau est déjà connu, cette commande peut être utilisée pour obtenir la configuration du panneau. Un dispositif d'affichage doit prendre en charge l'extraction d'une configuration de panneau spécifique grâce à cette commande (voir Tableau 201).

Tableau 201 – GetPaneConfiguration

GetPaneConfiguration		Demande-Réponse
Nom du message	Description	
GetPaneConfigurationRequest	<i>Ce message contient le jeton du panneau dont la configuration est demandée. Il contient également un jeton VideoOutput qui spécifie la sortie vidéo contenant le panneau demandé.</i> <i>Ce message contient également le jeton du panneau demandé.</i> tt:ReferenceToken VideoOutput[1][1] tt:ReferenceToken Pane[1][1]	
GetPaneConfigurationResponse	<i>Contient le PaneConfiguration demandé</i> tt:PaneConfiguration PaneConfiguration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoPane	<i>Le panneau demandé n'existe pas</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie vidéo demandée n'existe pas.</i>	

14.1.3 SetPaneConfigurations

Cette commande permet de modifier la configuration de tous les panneaux existants en une étape. Le message contient tous les PaneConfigurations (modifiés et non modifiés) de la sortie vidéo. Un dispositif d'affichage doit prendre en charge la modification de la configuration de ses panneaux grâce à cette commande (voir Tableau 202).

Tableau 202 – SetPaneConfigurations

SetPaneConfigurations		Demande-Réponse
Nom du message	Description	
SetPaneConfigurationsRequest	<i>Ce message contient la configuration de tous les panneaux du VideoOutput spécifié.</i> <i>L'élément PaneConfiguration contient la configuration modifiée.</i> tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][non limité]	
SetPaneConfigurationsResponse	<i>Ce message est vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>La configuration ne peut pas être définie</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie video demandée n'existe pas.</i>	

14.1.4 SetPaneConfiguration

Cette commande modifie la configuration d'un seul panneau. Un dispositif d'affichage doit prendre en charge la modification de la configuration d'un seul panneau grâce à cette commande (voir Tableau 203).

Tableau 203 – SetPaneConfiguration

SetPaneConfiguration		Demande-Réponse
Nom du message	Description	
SetPaneConfigurationRequest	<i>Ce message contient le jeton de la sortie vidéo et le nouveau PaneConfiguration.</i> tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][1]	
SetPaneConfigurationResponse	<i>Ce message est vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoPane	<i>Le panneau demandé n'existe pas</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie vidéo demandée n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>La configuration ne peut pas être définie</i>	

14.1.5 CreatePaneConfiguration

Cette commande crée la configuration d'un panneau. Un dispositif d'affichage qui prend en charge la création dynamique des panneaux doit prendre en charge la création d'une configuration de panneau grâce à cette commande (voir Tableau 204).

Tableau 204 – CreatePaneConfiguration

CreatePaneConfiguration		Demande-Réponse
Nom du message	Description	
CreatePaneConfigurationRequest	<i>Ce message contient le jeton de la sortie vidéo et le nouveau PaneConfiguration.</i> tt:ReferenceToken VideoOutput[1][1] tt:PaneConfiguration PaneConfiguration[1][1]	
CreatePaneConfigurationResponse	<i>Ce message est vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:MaxNumberOfPane	<i>Le nombre maximal de panneaux a été atteint</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie vidéo demandée n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>La configuration ne peut pas être définie</i>	

14.1.6 DeletePaneConfiguration

Cette commande supprime la configuration d'un panneau. Un dispositif d'affichage qui prend en charge la suppression dynamique des panneaux doit prendre en charge la suppression d'une configuration de panneau grâce à cette commande (voir Tableau 205).

Tableau 205 – DeletePaneConfiguration

DeletePaneConfiguration		Demande-Réponse
Nom du message	Description	
DeletePaneConfigurationRequest	<i>Ce message contient le jeton de la sortie vidéo et le nouveau PaneConfiguration.</i> tt:ReferenceToken VideoOutput[1][1] tt:ReferenceToken PaneToken[1][1]	
DeletePaneConfigurationResponse	<i>Ce message est vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:FixedPane	<i>Cette configuration de panneau ne peut pas être supprimée.</i>	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie vidéo demandée n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoPane	<i>La configuration de panneau demandée n'existe pas.</i>	

14.2 Présentation

La présentation attribue une configuration de panneau à une certaine zone de l'écran. Les paramètres de présentation affectent une sortie vidéo spécifique. La présentation est composée d'une liste de PaneConfigurations et de leurs zones d'affichage associées. Si le dispositif prend en charge le chevauchement des panneaux, l'ordre d'affichage des panneaux à l'écran est défini par l'ordre des PaneConfigurations dans la liste. Le premier panneau de la liste est celui qui s'affiche en avant-plan.

Un dispositif PEUT fournir un nombre fixe de présentations prises en charge, ou la présentation peut être configurée librement.

14.2.1 GetLayout

Cette commande renvoie la présentation en cours d'une sortie vidéo. Un dispositif d'affichage doit prendre en charge l'extraction de sa présentation grâce à cette commande (voir Tableau 206).

Tableau 206 – GetLayout

GetLayout	Demande-Réponse
Nom du message	Description
GetLayoutRequest	<i>Contient le VideoOutputToken de la sortie à laquelle est connecté l'écran.</i> tt:ReferenceToken VideoOutput[1][1]
GetLayoutResponse	<i>Contient la présentation en cours du VideoOutput.</i> tt:Layout Layout[1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie vidéo demandée indiquée avec VideoOutput n'existe pas.</i>

14.2.2 SetLayout

L'opération SetLayout peut être utilisée pour modifier la présentation d'un affichage (passage d'une vue simple à une vue d'écran divisée, par exemple). Un dispositif d'affichage doit prendre en charge la modification de la présentation grâce à cette commande (voir Tableau 207).

Tableau 207 – SetLayout

SetLayout	Demande-Réponse
Nom du message	Description
SetLayoutRequest	<i>Ce message contient le jeton de la sortie vidéo et la présentation modifiée.</i> tt:ReferenceToken VideoOutput [1][1] tt:Layout Layout[1][1]
SetLayoutResponse	<i>Ceci est un message vide.</i>
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:InvalidLayout	<i>La présentation ne peut pas être définie.</i>
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>La sortie vidéo demandée indiquée avec VideoOutput n'existe pas.</i>

14.3 DisplayOptions

14.3.1 Généralités

Les options d'affichage contiennent les présentations prises en charge (LayoutOptions) et les fonctionnalités de décodage et de codage (CodingCapabilities) du dispositif. La commande GetDisplayOptions renvoie les fonctionnalités de présentation et de codage d'un VideoOutput.

14.3.2 LayoutOptions

L'élément LayoutOptions décrit les présentations fixes et prédéfinies d'un dispositif. Si le dispositif n'offre pas de présentations fixes et permet de paramétrer librement la présentation, cet élément est vide.

14.3.3 CodingCapabilities

L'affichage vidéo en réseau peut décoder les flux audio et vidéo à l'aide d'algorithmes de décodage adaptés. L'affichage vidéo en réseau prend en charge des décodeurs audio et vidéo, des débits binaires et une résolution en fonction des choix du fabricant.

Afin d'assurer l'interopérabilité entre les différents dispositifs, la présente norme précise les profils de décodeur suivants:

Le NVD doit prendre en charge JPEG QVGA.

Le NVD doit prendre en charge G.711 μ Law (s'il prend en charge l'audio)

Il s'agit des mêmes codecs que ceux qui sont obligatoires pour le NVT.

Il n'existe aucun paramètre de configuration d'un décodeur. Un décodeur doit décoder tout le contenu (en fonction de ses fonctionnalités) qu'il reçoit. En cas d'erreurs de décodage, il convient que le décodeur tente de demander un point de synchronisation et de poursuivre le décodage. Il doit générer un événement tel que défini en 14.4.2.

L'élément CodingCapabilities donne une indication sur les fonctionnalités de décodage et de codage du dispositif.

14.3.4 GetDisplayOptions

Cette commande permet de répertorier les fonctionnalités de présentation et de codage d'une sortie vidéo. Un dispositif d'affichage doit prendre en charge l'extraction de ses DisplayOptions grâce à cette commande (voir Tableau 208).

Tableau 208 – GetDisplayOptions

GetDisplayOptions		Demande-Réponse
Nom du message	Description	
GetDisplayOptionsRequest	<i>Contient un PaneToken qui indique le profil multimédia à supprimer.</i> tt:ReferenceToken VideoOutput[1][1]	
GetDisplayOptionsResponse	<i>Ce message contient le jeton de la sortie vidéo à laquelle sont destinées les options.</i> tt:LayoutOptions LayoutOptions[0][1] tt:CodingCapabilities CodingCapabilities[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoVideoOutput	<i>Le VideoOutputToken demandé n'existe pas</i>	

14.4 Événements

14.4.1 Généralités

Le service d'affichage doit expédier les événements grâce au service d'événement.

14.4.2 Décodage d'événement d'erreur

Le dispositif doit pouvoir générer l'événement suivant à chaque fois qu'il reçoit un train de bits qu'il ne sait pas décoder.

Plusieurs raisons expliquent pourquoi le décodeur ne peut pas décoder le train de bits. Les codes d'erreur ci-dessous sont définis et doivent être utilisés par le dispositif pour informer le client des erreurs du décodeur:

- 1) "codec non pris en charge ou profil de codec non pris en charge" – le dispositif ne peut pas décoder le train de bits car il ne prend pas en charge le codec ou le profil. Il convient que le client tente de reconfigurer l'émetteur en fonction de l'élément CodingCapabilities du dispositif;
- 2) "erreur de paquet" – Il manque des paquets dans le train de bits ou le train de bits contient des paquets imprévus.

D'autres codes spécifiques au fournisseur sont également admis.

```

Topic: tns1:VideoDecoder/DecodingError
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="VideoOutputToken"
      Type="tt:ReferenceToken"/>
  
```

```

</tt:Source>
<tt:Data>
  <tt:SimpleItemDescription Name="PaneReference"
Type="tt:ReferenceToken"/>
  <tt:SimpleItemDescription Name="Error"
Type="xs:string" minOccurs="0"/>
</tt:Data>
</tt:MessageDescription>

```

14.5 Codes de défaut spécifiques au service

Le Tableau 209 présente les codes de défaut spécifiques au service d'affichage. De plus, chaque commande peut également générer un défaut générique (voir Tableau 6).

Les défauts spécifiques sont définis en tant que sous-code d'un défaut générique (voir 5.11.2.1). Le sous-code générique parent est le *sous-code* en haut de chaque ligne ci-dessous, le *sous-code* de défaut spécifique se trouvant en bas de la cellule.

Tableau 209 – Codes de défaut spécifiques au service

Code de défaut	de	Sous-code parent	Raison de défaut	Description
		Sous-code		
env:Sender		ter:InvalidArgVal	La sortie vidéo n'existe pas	La sortie vidéo demandée n'existe pas.
		ter:NoVideoOutput		
env:Sender		ter:InvalidArgVal	PaneConfiguration est fixe.	Cette configuration de panneau ne peut pas être supprimée.
		ter:FixPane		
env:Sender		ter:InvalidArgVal	Le nombre maximal de panneaux a été atteint.	Aucun panneau ne peut être créé, car le nombre maximal de panneaux a été atteint
		ter:MaximumNumberOfPanels		
env:Sender		ter:InvalidArgVal	PaneConfiguration n'existe pas	La configuration de panneau demandée n'existe pas.
		ter:NoPane		
env:Sender		ter:InvalidArgVal	La configuration ne peut pas être définie.	La configuration demandée n'est pas prise en charge par le dispositif.
		ter:invalidConfig		
env:Sender		ter:Action	La présentation ne peut pas être définie.	La présentation demandée n'est pas prise en charge par le dispositif.
		ter:InvalidLayout		

15 Traitement des événements

Un événement est une action ou occurrence détectée par un dispositif à laquelle un client peut s'abonner. Les événements sont gérés par l'intermédiaire du service d'événement. Un

dispositif doit fournir des services d'événement comme défini en C.6. Un NVT et un NVC doivent prendre en charge [WS-Addressing] pour les services d'événement.

Le traitement des événements de la présente Norme repose sur les spécifications [WS-BaseNotification] et [WS-Topics]. La présente Norme requiert la mise en œuvre de l'interface de notification de base (voir 15.1) qui est entièrement conforme à la spécification [WS-BaseNotification]. De plus, le dispositif doit mettre en œuvre l'interface Real-time Pull-Point Notification (notification de point d'extraction en temps réel) et l'interface Notification Streaming (transmission continue de notification) décrites en 15.2 et en 15.3, respectivement.

La présente Norme décrit les extensions de message de notification permettant à un client de suivre les propriétés d'objet (propriétés d'objet d'analyse vidéo, par exemple) grâce aux événements. Les propriétés sont définies en 15.4.

La description de la charge utile d'événement et leur filtrage dans les abonnements est présentée en 15.5. Le paragraphe 15.6 explique comment le client peut demander un point de synchronisation à l'aide de l'une des trois interfaces de notification. Le paragraphe 15.7 décrit l'intégration de rubriques. Le paragraphe 15.9 décrit la gestion des défauts.

Le dernier paragraphe illustre de manière détaillée l'utilisation de l'interface de notification Real-Time Pull-Point comprenant le filtrage des messages et la définition des rubriques. Des exemples pour l'interface de notification de base peuvent être trouvés dans la spécification correspondante [WS-BaseNotification].

15.1 Interface de notification de base

Le paragraphe 15.1.1 présente brièvement l'interface de notification de base de la spécification [WS-BaseNotification]. Le paragraphe 15.1.2 résume les interfaces obligatoires et facultatives de la spécification [WS-BaseNotification].

15.1.1 Généralités

Les entités logiques suivantes participent au profil de notification:

- client: met en œuvre l'interface NotificationConsumer;
- service d'événement: met en œuvre l'interface NotificationConsumer;
- gestionnaire d'abonnements: met en œuvre l'interface BaseSubscriptionManager.

Il convient d'instancier le service d'événement et le gestionnaire d'abonnements sur un dispositif.

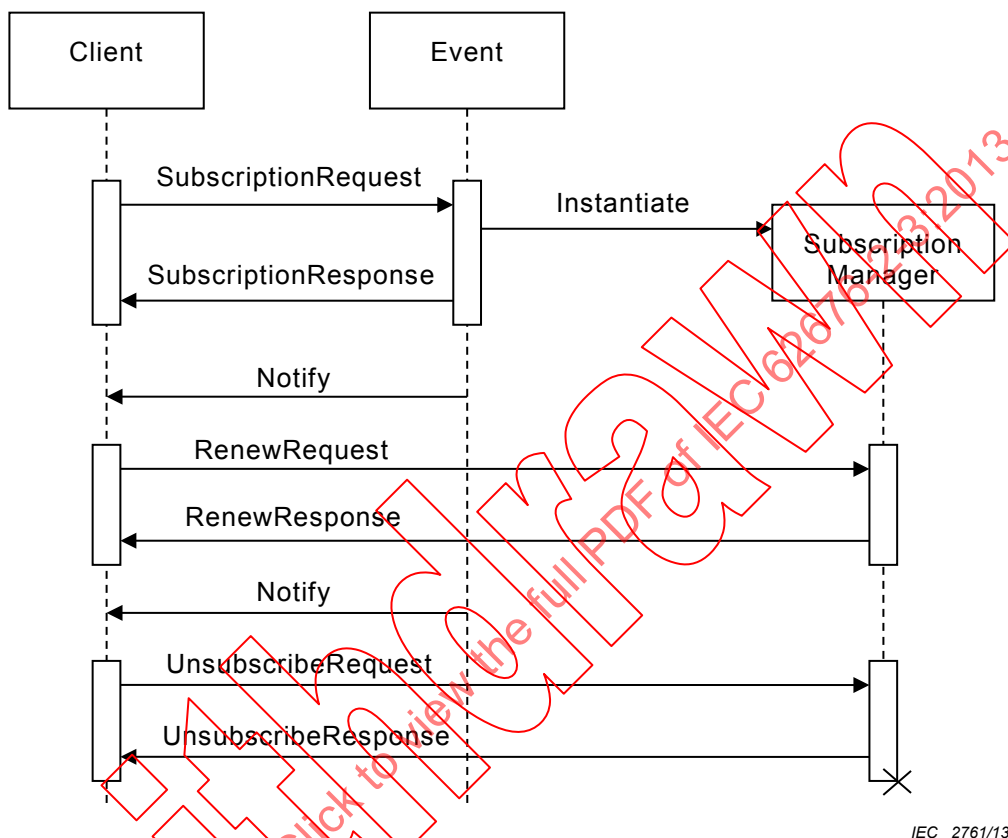
Des messages types échangés entre les entités sont présentés dans le schéma de séquence de la Figure 21. Le client établit d'abord une connexion au service d'événement. Ensuite, il peut s'abonner pour certaines notifications en envoyant une SubscriptionRequest. Si le service d'événement accepte l'abonnement, il instancie de manière dynamique un SubscriptionManager représentant l'abonnement. Le service d'événement doit retourner le WS-Endpoint-Address du SubscriptionManager dans la SubscriptionResponse.

Afin de transmettre des notifications correspondant à l'abonnement, une autre connexion est établie entre le service d'événement et le client. Grâce à cette connexion, le service d'événement envoie un message de notification unilatéral à l'interface NotificationConsumer du client. Les notifications correspondantes peuvent être envoyées à tout moment au client par le service d'événement, tant que l'abonnement est actif.

Pour contrôler l'abonnement, le client adresse directement le SubscriptionManager retourné dans la SubscriptionResponse. Dans la SubscriptionRequest, le client peut spécifier un temps d'expiration. Le SubscriptionManager est automatiquement détruit lorsque le temps d'expiration est atteint. Des RenewRequests peuvent être initiées par le client afin de retarder

le temps d'expiration. Le client peut également terminer le SubscriptionManager de manière explicite en envoyant une UnsubscribeRequest. Après un désabonnement réussi, le SubscriptionManager n'existe plus.

L'interaction entre un EventService et un SubscriptionManager n'est pas décrite de manière plus détaillée par la spécification [WS-BaseNotification] et dépend de la mise en œuvre du dispositif.



Légende

Anglais	Français
Event	Événement
Instantiate	Instanciation
Notify	notification

Figure 21 – Schéma de séquence de l'interface de notification de base

15.1.2 Exigences

Le présent paragraphe décrit de manière détaillée les interfaces de [WS-BaseNotification] qu'un dispositif doit fournir.

Un dispositif satisfaisant à l'ONVIF doit prendre en charge l'interface NotificationProducer de la spécification [WS-BaseNotification]. En conséquence, les propriétés de ressource NotificationProducer sont FACULTATIVES (voir 15.5). Le dispositif doit prendre en charge les filtres TopicExpression et MessageContent avec au moins les dialectes décrits en 15.5.5 et en 15.7.3. Si le dispositif n'accepte pas l'InitialTerminationTime d'un abonnement, il doit en fournir un valide dans le message de défaut. Le dispositif doit être capable de produire des notifications en utilisant l'enveloppe Notify de la spécification [WS-BaseNotification]. La SubscriptionPolicy wsnt:UseRaw est FACULTATIVE pour le dispositif. Bien que [WS-

BaseNotification] ait un CurrentTime et un TerminationTime en tant qu'éléments facultatifs dans un SubscribeResponse, un dispositif satisfaisant à l'ONVIF doit les énumérer dans des SubscribeResponse. Le dispositif PEUT répondre à une demande GetCurrentMessage par un message de défaut indiquant qu'aucun message actuel n'est disponible sur la rubrique demandée.

La mise en œuvre de l'interface Pull-Point de la spécification [WS-BaseNotification] sur un dispositif est FACULTATIVE.

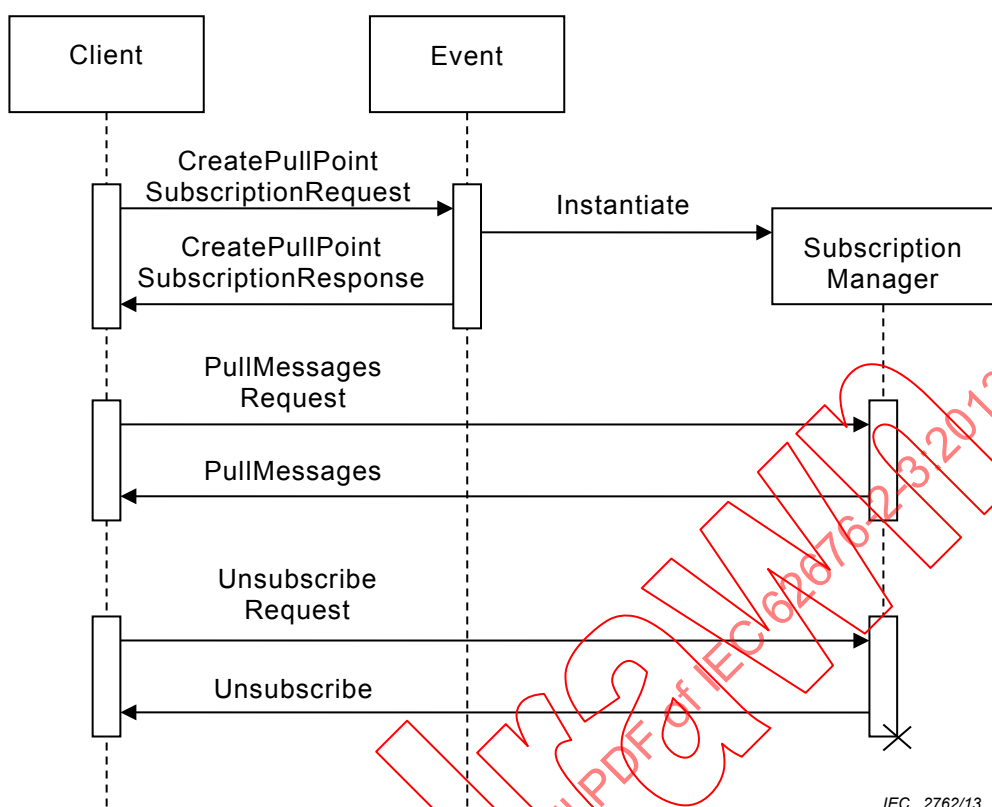
Le dispositif satisfaisant à l'ONVIF doit mettre en œuvre l'interface de gestionnaire d'abonnement de base de la spécification [WS-BaseNotification] composée des opérations Renew (Renouveler) et Unsubscribe (Désabonner). L'interface de gestionnaire d'abonnements pausable est FACULTATIVE. La mise en œuvre des abonnements en tant que WS-Resources est FACULTATIVE.

15.2 Interface de notification Real-time Pull-Point

Le présent paragraphe présente l'interface de notification Real-time Pull-Point (point d'extraction en temps réel). Il s'agit d'une interface de notification conviviale de pare-feu qui permet le groupage en temps réel et initie toutes les communications de client.

Cette interface est utilisée de la façon suivante:

- 1) le client demande au dispositif un PullPointSubscription avec le message CreatePullPointSubscriptionRequest. La demande contient une description détaillée de l'abonnement. La ConsumerReference doit être ignorée, contrairement à l'abonnement de l'interface de notification de base (voir 15.1);
- 2) le dispositif évalue l'abonnement et retourne une CreatePullPointSubscriptionResponse lorsque l'abonnement est accepté ou l'un des codes de défaut;
- 3) si l'abonnement est accepté, la réponse contient un WS-EndpointReference pour un SubscriptionManager. Ce WS-Endpoint doit produire une opération PullMessages, utilisée par le client pour extraire des notifications et par l'interface de gestionnaire d'abonnements de base décrite dans la spécification [WS-BaseNotification]. L'interface de gestionnaire d'abonnements de base est constituée d'opérations PullMessages, Renew et Unsubscribe. Le schéma de séquence de l'interaction est décrit à la Figure 22. La PullMessagesRequest contient les paramètres Timeout et MessageLimit;



Légende

Anglais	Français
Event	Événement
Instantiate	Instanciation
Notify	notification
Unsubscribe	Désabonnement

Figure 22 – Schéma de séquence pour l'interface de notification Real-time Pull-Point

- 4) le dispositif doit répondre immédiatement avec des notifications qui ont été agrégées au nom du client. S'il n'existe pas de notifications agrégées, le dispositif attend sa réponse tant qu'une notification n'est pas produite pour le client ou que le délai d'attente spécifié n'est pas dépassé. Dans tous les cas, la réponse contient, au plus, le nombre de notifications spécifié par le paramètre MessageLimit. Le client peut grouper les notifications en temps réel au démarrage d'une nouvelle PullMessagesRequest immédiatement après chaque PullMessagesResponse;
- 5) si aucun temps d'expiration ou temps d'expiration relatif n'est défini dans la CreatePullPointSubscriptionRequest, chaque PullMessagesRequest doit être interprétée comme un maintien actif ("keep-alive") pour le PullPointSubscription correspondant. Le temps d'expiration est recalculé en fonction du temps d'expiration relatif, s'il est disponible, ou d'une valeur par défaut interne du dispositif. Afin d'informer le client du temps d'expiration mis à jour, la PullMessagesReponse doit contenir les éléments CurrentTime et TerminationTime. Lorsque la PullMessagesRequest est utilisée en tant que maintien actif ("keep-alive") pour le PullPointSubscription correspondant, la RenewRequest, définie par la spécification [WS-BaseNotification], peut ne pas être appelée par un client. Cependant, le dispositif doit la prendre en charge pour le PullPointSubscription.

15.2.1 Création d'abonnement de point d'extraction

Le dispositif doit fournir la commande CreatePullPointSubscription donnée dans le Tableau 210.

Tableau 210 – Commande CreatePullPointSubscription

CreatePullPointSubscription		Demande-réponse
Nom du message	Description	
CreatePullPointSubscriptionRequest	<i>Ce message contient les mêmes éléments que la SubscriptionRequest (demande d'abonnement) de la spécification [WS-BaseNotification] sans la ConsumerReference (référence de consommateur):</i> wsnt:FilterType Filter [0][1] wsnt:AbsoluteOrRelativeTimeType InitialTerminationTime [0][1] xs:any SubscriptionPolicy [0][1]	
CreatePullPointSubscriptionResponse	<i>La réponse contient les mêmes éléments que la SubscriptionResponse de la [WS-BaseNotification].</i> wsa:EndpointReferenceType SubscriptionReference [1][1] xs:dateTime CurrentTime [1][1] xs:dateTime TerminationTime [1][1]	
Codes de défaut	Description	
	<i>Les mêmes défauts que pour la demande d'abonnement de la spécification [WS-BaseNotification] sont utilisés.</i>	

15.2.2 Messages Pull

Le dispositif doit produire la commande PullMessages suivante pour tous les points terminaux SubscriptionManager retournés par la commande CreatePullPointSubscription (voir Tableau 211).

Tableau 211 – Commande PullMessages

PullMessages		Demande-réponse
Nom du message	Description	
PullMessagesRequest	<i>Ce message doit être adressé à un SubscriptionManager afin d'extraire des notifications:</i> xs:duration Timeout [1][1] xs:int MessageLimit [1][1]	
PullMessagesResponse	<i>La réponse contient une liste de notifications conjointement avec un TerminationTime mis à jour pour le SubscriptionManager:</i> xs:dateTime CurrentTime [1][1] xs:dateTime TerminationTime [1][1] wsnt:NotificationMessageHolderType [0][non limité] NotificationMessage	
PullMessagesFaultResponse	<i>Le délai d'attente (Timeout) dépasse la limite supérieure prise en charge par le dispositif. Le message de défaut doit contenir les limites supérieures pour les deux paramètres.</i> xs:duration MaxTimeout[1][1] xs:int MaxMessageLimit[1][1]	
Codes de défaut	Description <i>Pas de codes de défaut spécifiques.</i>	

15.3 Interface de transmission en continu de notification

L'Article 10 décrit la création, la suppression et la modification de configurations de métadonnées. Certaines configurations de métadonnées peuvent contenir plusieurs abonnements dont la structure est identique à celle d'un abonnement de notification. Si une configuration de métadonnées contenant des abonnements a été attribuée à un profil, un dispositif utilise ce profil pour obtenir un flux RTP comprenant les notifications configurées en tant que métadonnées. La transmission en continu de notification via RTP doit être mise en œuvre par un dispositif satisfaisant à l'ONVIF.

La spécification [WS-BaseNotification] définit l'élément `wsnt:NotificationMessage` pour contenir la charge utile de message, la rubrique et le `ProducerReference`. La structure de ce message est identique à celle des demandes de notification directe (le format est décrit en 15.5). Plusieurs instances des éléments `wsnt:NotificationMessage` peuvent être placées dans un document de métadonnées présenté à la section Visualisation en temps réel.

Les notifications de transmission continue ne sont pas associées à des éléments `SubscriptionReference` explicites. Par conséquent, l'élément `wsnt:NotificationMessage` ne doit PAS contenir l'élément `SubscriptionReference`.

15.4 Propriétés

Une propriété est un ensemble de paires nom-valeur représentant un ensemble de données unique et adressable. Ces paires sont identifiées de manière unique par la combinaison de

leurs valeurs de rubrique, de source et de clé, et sont formatées comme des événements ordinaires. Une propriété contient également un drapeau supplémentaire indiquant s'il s'agit d'une propriété créée, modifiée ou supprimée.

- Si un client s'abonne à une rubrique représentant une certaine propriété, le dispositif doit informer le client de tous les objets comportant la propriété demandée et qui sont actifs lors de l'abonnement. Un client *peut* également demander à tout moment les valeurs de toutes les propriétés actuellement actives auxquelles le client s'est abonné, en demandant un point de synchronisation (voir 15.6).

L'interface de propriété est définie dans la présente Norme afin de grouper tous les événements associés à des propriétés et les présenter uniformément aux clients. Il est CONVIENT d'utiliser l'interface de propriété dans la mesure du possible. Le paragraphe 15.5 décrit de manière détaillée la structure des événements et des propriétés.

15.4.1 Exemple de propriété

L'exemple d'analyse vidéo suivant illustre le comportement dynamique des propriétés: L'interface de moteur de règles du détecteur d'analyse vidéo peut définir des champs. Ce type de champ de détecteur est décrit par un polygone dans le plan de l'image. Pour chaque objet de la scène, le moteur de règles détermine les objets se trouvant dans le polygone. Un client peut accéder à ces informations en s'abonnant à la propriété ObjectsInside correspondante du champ de détecteur. Chaque fois qu'un objet apparaît dans la scène, une propriété ObjectsInside est créée. Le client est informé par une notification "propriété créée" correspondante indiquant si l'objet apparaît à l'intérieur ou à l'extérieur du polygone. Chaque fois qu'un objet entre dans le polygone ou en sort, une notification "propriété modifiée" est générée indiquant que la propriété ObjectsInside pour cet objet a changé. Lorsqu'un objet sort de la scène, la propriété ObjectsInside correspondante est supprimée, et le client en est informé par une notification "propriété supprimée".

15.5 Structure des notifications

Le code suivant est le schéma de la notification wsnt:NotificationMessage [WS-BaseNotification].

```
<xs:complexType name="NotificationMessageHolderType" >
  <xs:sequence>
    <xs:element ref="wsnt:SubscriptionReference" minOccurs="0" />
    <xs:element ref="wsnt:Topic" minOccurs="0" />
    <xs:element ref="wsnt:ProducerReference" minOccurs="0" />
    <xs:element name="Message">
      <xs:complexType>
        <xs:sequence>
          <xs:any namespace="##any" processContents="lax" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:element name="NotificationMessage"

      type="wsnt:NotificationMessageHolderType"/>
```

Cela correspond à la structure XML suivante:

```
<wsnt:NotificationMessage>
  <wsnt:SubscriptionReference>
    wsa:EndpointReferenceType
  </wsnt:SubscriptionReference>
  <wsnt:Topic Dialect="xs:anyURI">
    ...
```

```

</wsnt:Topic>?
<wsnt:ProducerReference>
  wsa:EndpointReferenceType
</wsnt:ProducerReference>
<wsnt:Message>
  ...
</wsnt:Message>
</wsnt:NotificationMessage>

```

où l'élément `wsnt:Message` contient la charge utile de notification réelle. Le type XML de l'élément `Message` peut être spécifié dans une définition `TopicTree` (voir 15.7).

Le paragraphe 15.5.1 donne une vue d'ensemble des informations qu'un client extrait à l'aide de notifications. Le paragraphe 15.5.2 présente un formatage détaillé de la charge utile du message, et le paragraphe 15.5.4 un langage de description pour la charge utile du message. Le paragraphe 15.5.5 définit la grammaire utilisée dans un abonnement pour filtrer les notifications en fonction du contenu de leur message.

15.5.1 Informations de notification

Une notification répond au moins aux questions suivantes:

- Quand cela s'est-il produit?
- Qui a généré l'événement?
- Que s'est-il passé?

La réponse à la question "quand?" est obtenue en ajoutant un attribut temporel à l'élément `Message` de l'objet `NotificationMessage`. Le dispositif satisfaisant à l'ONVIF doit inclure l'attribut temporel dans l'élément `Message`.

La question "qui?" est divisée en deux parties. Une partie est le point terminal WS-Endpoint qui identifie le dispositif ou un service dans le dispositif dans lequel la notification a été générée. Par conséquent, il convient que WS-Endpoint soit spécifié dans l'élément `ProducerReference` du `NotificationMessage`. La deuxième partie est l'identification du composant dans WS-Endpoint, qui est responsable de la production de la notification. Selon le composant, plusieurs paramètres ou aucun peuvent être nécessaires pour identifier le composant de façon unique. Ces paramètres sont placés en tant qu'éléments (`Items`) dans l'élément `Source` du conteneur de `Message`.

La réponse à la question "que?" est obtenue en deux étapes. Premièrement, l'élément `Topic` du `NotificationMessage` est utilisé pour classer l'événement. Deuxièmement, des éléments sont ajoutés à l'élément `Data` du conteneur de `Message` afin de décrire les détails de l'événement.

Lorsque la rubrique pointe sur des propriétés (voir 15.4), le client utilise les éléments `NotificationProducer`, `Topic`, `Source` et les éléments `Key` facultatifs (voir 15.5) pour identifier la propriété. Ces valeurs doivent conduire à un identifiant unique.

Exemple d'événement

- L'exemple suivant illustre les différentes parties de la notification:

```

<wsnt:NotificationMessage>
  ...
  <wsnt:Topic Dialect="...Concrete">
    tns1:PTZController/PTZPreset/Reached
  </wsnt:Topic>
  <wsnt:Message>
    <tt:Message UtcTime="...">
      <tt:Source>

```

```

        <tt:SimpleItem                                Name="PTZConfigurationToken"
Value="PTZConfig1"/>
    </tt:Source>
    <tt>Data>
        <tt:SimpleItem Name="PresetToken"                Value="Preset5"/>
        <tt:SimpleItem                                Name="PresetName"
Value="ParkingLot"/>
    </tt>Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>

```

L'élément "PTZConfigurationToken" identifie de manière unique le composant responsable de la détection de l'événement. Dans cet exemple, le composant est un nœud PTZ référencé par la configuration PTZ "PTZConfig1". L'événement tns1:PTZController/PTZPreset/Reached indique que l'unité PTZ a atteint un préréglage. Le bloc de données contient les informations indiquant de quel préréglage il s'agit. Ainsi, le préréglage est identifié par un élément PresetToken "Preset5" appelé "PresetName".

15.5.2 Format de message

L'élément Message du NotificationMessage est défini dans [ONVIF Schema]. La définition est présentée ci-dessous⁴:

```

<xs:element name="Message" type="Message">
  <xs:element name="Message">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Source" type="tt:ItemList" minOccurs="0"/>
        <xs:element name="Key" type="tt:ItemList" minOccurs="0"/>
        <xs:element name="Data" type="tt:ItemList" minOccurs="0"/>
        ...
      </xs:sequence>
      <xs:attribute name="UtcTime" type="xs:time" use="required"/>
      <xs:attribute name="PropertyOperation" type="tt:PropertyOperationType"/>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="ItemList">
    <xs:sequence>
      <xs:element name="SimpleItem" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="Name" type="xs:string" use="required"/>
          <xs:attribute name="Value" type="xs:anySimpleType" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="ElementItem" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:any namespace="##any"/>
          </xs:sequence>
          <xs:attribute name="Name" type="xs:string" use="required"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="PropertyOperationType">

```

⁴ À noter que le schéma est inclus ici à titre informatif uniquement. [ONVIF Schema] contient la définition de schéma normative.


```

<xs:restriction base="xs:string">
  <xs:enumeration value="Initialized"/>
  <xs:enumeration value="Deleted"/>
  <xs:enumeration value="Changed"/>
</xs:restriction>
</xs:simpleType>

```

Les éléments de l'élément Message sont groupés en trois catégories: Source, Key et Data. Le groupe Key ne doit PAS être utilisé par des notifications qui ne sont pas associées à des propriétés. Plusieurs éléments Simple et Element peuvent être placés dans chaque groupe. Chaque élément a un nom et une valeur. Dans le cas d'un élément ElementItem, la valeur est exprimée par un élément XML dans l'élément ElementItem. Dans le cas d'un élément SimpleItem, la valeur doit être spécifiée par l'attribut de valeur. Le nom de tous les éléments doit être unique pour l'ensemble des éléments contenus dans un groupe du message en question.

Dans la mesure du possible, il est RECOMMANDÉ d'utiliser des éléments SimpleItem plutôt que des éléments ElementItem, étant donné que les éléments SimpleItem facilitent l'intégration de messages dans un client générique. Les informations de type exactes des éléments SimpleItem et ElementItem peuvent être extraites du TopicSet (voir 15.7), où chaque rubrique peut être complétée par une description de la charge utile du message.

L'élément PropertyOperation doit être présent si la notification concerne une propriété. Le mode d'opération "Initialized" (Initialisé) doit être utilisé pour informer un client de la création d'une propriété. Le mode d'opération "Initialized" doit être utilisé si un point de synchronisation a été demandé.

15.5.3 Exemple de propriété (suite)

L'exemple de 15.4.1 impliquait un élément Key facultatif. L'exemple dans le présent paragraphe illustre l'application d'éléments Key. Le moteur de règles peut contenir des règles FieldDetector. Ces règles définissent une propriété ObjectsInside pour chaque objet de la scène. Si un nouvel objet apparaît à l'extérieur d'un tel champ, la notification suivante est générée:

```

<wsnt:NotificationMessage>
  ...
  <wsnt:Topic Dialect="...Concrete">
    tns1:RuleEngine/FieldDetector/ObjectsInside
  </wsnt:Topic>
  <wsnt:Message>
    <tt:Message UtcTime="..." PropertyOperation="Initialized">
      <tt:Source>
        <tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
        <tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
        <tt:SimpleItem Name="Rule" Value="myImportantField"/>
      </tt:Source>
      <tt:Key>
        <tt:SimpleItem Name="ObjectId" Value="5"/>
      </tt:Key>
      <tt>Data>
        <tt:SimpleItem Name="IsInside" Value="false"/>
      </tt>Data>
    </tt:Message>
  </wsnt:Message>
</wsnt:NotificationMessage>

```

Les éléments Source décrivent la règle qui a généré la notification. Si la scène contient plusieurs objets, chacun d'eux a sa propre propriété ObjectsInside. Par conséquent, l'Object ID est utilisé en tant qu'élément Key supplémentaire afin de rendre la propriété unique. L'élément IsInside est une valeur booléenne indiquant si l'objet est à l'intérieur ou à l'extérieur du champ.

Si l'objet entre dans le champ, la règle génère un message "propriété modifiée" et s'apparente à l'exemple suivant:

```
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
  tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
  <tt:Message UtcTime="..." PropertyOperation="Changed">
    <tt:Source>
      <tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="Rule" Value="myImportantField"/>
    </tt:Source>
    <tt:Key>
      <tt:SimpleItem Name="ObjectId" Value="5"/>
    </tt:Key>
    <tt>Data>
      <tt:SimpleItem Name="IsInside" Value="true"/>
    </tt>Data>
  </tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
Finally, when the object leaves the scene, a "property deleted" message is
produced:
<wsnt:NotificationMessage>
...
<wsnt:Topic Dialect="...Concrete">
  tns1:RuleEngine/FieldDetector/ObjectsInside
</wsnt:Topic>
<wsnt:Message>
  <tt:Message UtcTime="..." PropertyOperation="Deleted">
    <tt:Source>
      <tt:SimpleItem Name="VideoSourceConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="1"/>
      <tt:SimpleItem Name="Rule" Value="myImportantField"/>
    </tt:Source>
    <tt:Key>
      <tt:SimpleItem Name="ObjectId" Value="5"/>
    </tt:Key>
  </tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
```

Dans ce cas, l'élément Data peut être omis car l'objet et sa propriété correspondante n'existent plus.

15.5.4 Langage de description de message

La structure de la charge utile de message a été présentée au paragraphe précédent. La structure contient trois groupes: Source, Key et Data. Chaque groupe contient un ensemble d'éléments SimpleItem et ElementItem. Pour chaque rubrique, un dispositif peut décrire l'élément faisant partie d'une notification générée par la rubrique en question, à l'aide d'un langage de description de message. Le langage de description suivant décrit les éléments de message obligatoires⁵:

```
<xs:complexType name="MessageDescription">
  <xs:sequence>
    <xs:element name="Source" type="tt:ItemListDescription"
      minOccurs="0"/>
    <xs:element name="Key" type="tt:ItemListDescription" minOccurs="0"/>
    <xs:element name="Data" type="tt:ItemListDescription" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

⁵ À noter que le schéma est inclus ici à titre informatif uniquement. [ONVIF Schema] contient la définition de schéma normative.

```

</xs:sequence>
<xs:attribute name="IsProperty" type="xs:boolean"/>
</xs:complexType>

<xs:complexType name="ItemListDescription">
  <xs:sequence>
    <xs:element name="SimpleItemDescription"
      minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="ElementItemDescription"
      minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

```

L'attribut Name d'un élément doit être unique pour l'ensemble des éléments, quel que soit le groupe (Source, Key, Data) dont il provient. La valeur "true" doit être affectée à l'attribut IsProperty lorsque le message décrit concerne une propriété. Toutefois, si le message ne concerne pas une propriété, le groupe Key ne doit PAS être présent. L'attribut Type d'un élément SimpleItemDescriptor doit correspondre à la définition de SimpleElement d'un schéma XML. De même, l'attribut Type d'un élément ElementItemDescriptor doit correspondre à une déclaration d'élément globale d'un schéma XML.

L'emplacement de tous les fichiers de schéma utilisés pour décrire des charges utiles de message figure dans le message GetEventPropertiesResponse (voir 15.8).

Exemple de description de message

Le code suivant est un exemple de description de message correspondant à l'exemple de propriété de 15.5.3.

```

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescriptionDescription
      Name="VideoSourceConfigurationToken"
      Type="tt:ReferenceToken"/>
    <tt:SimpleItemDescriptionDescription
      Name="VideoAnalyticsConfigurationToken"
      Type="tt:ReferenceToken"/>
    <tt:SimpleItemDescriptionDescription Name="Rule"
      Type="xs:string"/>
  </tt:Source>
  <tt:Key>
    <tt:SimpleItemDescriptionDescription Name="ObjectId"
      Type="tt:ObjectRefType"/>
  </tt:Key>
  <tt>Data>
    <tt:SimpleItemDescriptionDescription Name="IsInside"
      Type="xs:boolean"/>
  </tt>Data>
</tt:MessageDescription>

```

15.5.5 Filtre de contenu de message

Dans la demande d'abonnement, un client peut filtrer les notifications par TopicExpression (voir 15.7.3) et MessageContent. Pour ce dernier, le [WS-BaseNotification] propose le dialecte XPath 1.0. En raison de la structure de message spécifique requise par la présente

spécification, la spécification requiert un sous-ensemble de la syntaxe XPath 1.0. Un dispositif satisfaisant à l'ONVIF doit mettre en œuvre le sous-ensemble de XPath 1.0. Le dialecte correspondant peut être référencé avec l'URI suivant:

Dialect=http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter

Priorité et associativité:

L'opération "and" a une priorité plus élevée que l'opération "or". Ces deux opérations restent associatives.

La priorité et l'associativité des opérations "and" et "or" dans la définition grammaticale ci-dessous sont identiques aux spécifications XPath 1.0.

La structure des expressions se présente comme suit:

- [1] Expression::= BoolExpr | Expression 'and' Expression
| Expression 'or' Expression | '(' Expression ')' | 'not' '(' Expression ')'
- [2] BoolExpr::= 'boolean' '(' PathExpr ')'
- [3] PathExpr::= ['/' Prefix? 'SimpleItem' | '/' Prefix? 'ElementItem'] NodeTest
- [4] Prefix::= NamespacePrefix ':' | ''
- [5] NodeTest::= '[' AttrExpr ']'
- [6] AttrExpr::= AttrComp | AttrExpr 'and' AttrExpr | AttrExpr 'or' AttrExpr | '(' AttrExpr ')' | 'not' '(' AttrExpr ')'
- [7] AttrComp::= Attribute '=' '"' String '"'
- [8] Attribute::= '@Name' | '@Value'

Cette syntaxe permet de soumettre à essai la présence d'éléments SimpleItem ou ElementItem indépendamment du groupe auquel ils appartiennent (Source, Key ou Data). De plus, la valeur d'éléments SimpleItem peut être vérifiée. L'espace de noms de préfixe d'élément SimpleItem et ElementItem doit être "http://www.onvif.org/ver10/schema".

Enfin, des combinaisons booléennes aléatoires de ces essais sont possibles. Les expressions suivantes peuvent être formulées:

- Retourner uniquement les notifications qui contiennent une référence à la VideoSourceConfiguration "1"

```
boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken" and
@Value="1"] )
```

- Retourner uniquement les notifications qui contiennent une référence à une VideoAnalyticsConfiguration

```
not (
boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken"] )
)
```

- Retourner uniquement les notifications associées à VideoAnalyticsConfiguration "2" s'exécutant sur VideoSourceConfiguration "1"

```
boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken" and
```

```

        @Value="2"] )
    and
    boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken" and
        @Value="1"] )

```

- Retourner uniquement les notifications associées à VideoSourceConfiguration "1", mais pas à VideoAnalyticsConfigurations

```

    boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken" and
        @Value="1"] )
    and
    not(boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken"
    ] ))

```

- Retourner uniquement les notifications lorsque les objets entrent ou apparaissent dans "myImportantField"

```

    boolean(//tt:SimpleItem[@Name="IsInside" and @Value="true"] )
    and
    boolean(//tt:SimpleItem[@Name="Rule" and @Value="myImportantField"]
    )

```

15.6 Point de synchronisation

Les propriétés, décrites en 15.4, informent un client de la création, des modifications et de la suppression de propriétés de manière uniforme. Lorsqu'un client attend la synchronisation de ses propriétés avec les propriétés du dispositif, il peut demander un point de synchronisation qui répète le statut actuel de toutes les propriétés auxquelles un client s'est abonné. L'élément PropertyOperation de toutes les notifications générées est défini sur "Initialized" (voir 15.5). Le point de synchronisation est directement demandé au SubscriptionManager qui a été retourné dans SubscriptionResponse ou dans CreatePullPointSubscriptionResponse. La mise à jour de propriété est transmise via le transport de notification de l'interface de notification. L'opération suivante donnée dans le Tableau 212 doit être générée par tous les points terminaux de gestionnaire d'abonnement:

Tableau 212 – Commande SetSynchronizationPoint

SetSynchronizationPoint		Demande-Réponse
Nom du message	Description	
SetSynchronizationPoint-Request	<i>Ce message est vide.</i>	
SetSynchronizationPoint-Response	<i>Ce message est vide.</i>	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

Lorsqu'un client utilise l'interface de transmission continue de notification, il convient qu'il utilise l'opération SetSynchronizationPoint définie dans le service multimédia (voir 11.18).

15.7 Structure de rubrique

La présente Norme étend le cadre de Topic défini dans la spécification [WS-Topics]. Le paragraphe 15.7.1 décrit un espace de noms de rubrique ONVIF, qu'il convient d'utiliser comme base pour les rubriques spécifiques au fournisseur. L'Annexe A présente des exemples classiques de ce type d'extension.

Le paragraphe 15.7.2 définit une interface pour les propriétés de rubrique. L'interface doit être mise en œuvre par dispositif satisfaisant à l'ONVIF. Le paragraphe 15.7.3 intègre le langage de description de message défini en 15.5.4 dans la structure TopicSet. Toutes les rubriques développées à partir de l'espace de noms de rubrique ONVIF décrivent le type d'une rubrique conformément à 15.7.3. Le paragraphe 15.7.3 définit les dialectes d'expression de rubrique pris en charge par un dispositif.

15.7.1 Espaces de noms de rubrique ONVIF

La spécification [WS-Topics] distingue la définition d'une arborescence de rubriques appartenant à un certain espace de noms de rubrique et l'ensemble de rubriques pris en charge par un certain service Web. Cette distinction permet aux fournisseurs de se référer à un espace de noms de rubrique commun tout en utilisant seulement une partie des rubriques définies.

Si l'arborescence de rubriques d'un espace de noms de rubrique existant couvre uniquement un sous-ensemble des rubriques disponibles par un dispositif, l'arborescence de rubriques peut être développée en définissant un nouvel espace de noms de rubrique. Un nouvel espace de noms de rubrique est défini en ajoutant une nouvelle rubrique à un espace de noms de rubrique existant comme décrit dans la spécification [WS-Topics].

Les rubriques racines suivantes sont définies dans l'espace de noms ONVIF. Toutes les notifications faisant référence à ces rubriques doivent utiliser le format de message décrit en 15.5.2.

```
<wstop:TopicNamespace name="ONVIF"
  targetNamespace="http://www.onvif.org/ver10/topics" >
  <wstop:Topic name="Device"/>
  <wstop:Topic name="VideoSource"/>
  <wstop:Topic name="VideoEncoder"/>
  <wstop:Topic name="VideoAnalytics"/>
  <wstop:Topic name="RuleEngine"/>
  <wstop:Topic name="PTZController"/>
  <wstop:Topic name="AudioSource"/>
  <wstop:Topic name="AudioEncoder"/>
  <wstop:Topic name="UserAlarm"/>
  <wstop:Topic name="MediaControl"/>
  <wstop:Topic name="RecordingConfig"/>
  <wstop:Topic name="RecordingHistory"/>
  <wstop:Topic name="VideoOutput"/>
  <wstop:Topic name="AudioOutput"/>
  <wstop:Topic name="VideoDecoder"/>
  <wstop:Topic name="AudioDecoder"/>
  <wstop:Topic name="Receiver"/>
</wstop:TopicNamespace>
```

15.7.2 Informations de type de rubrique

Les informations de type sont placées au-dessous d'un élément de rubrique en ajoutant un élément MessageDescription de type MessageDescriptionType défini en 15.5.4. Les éléments de rubrique peuvent être identifiés par l'attribut wstop:topic attribute avec la valeur "true".

L'exemple suivant illustre comment des rubriques d'un élément TopicSet sont ajoutées avec des descriptions de message:

```
<tns1:RuleEngine wstop:topic="true">
  <tns1:LineDetector wstop:topic="true">
    <tns1:Crossed wstop:topic="true">
      <tt:MessageDescription>
        <tt:Source>
          <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
            Type="tt:ReferenceToken"/>
        </tt:Source>
      </tt:MessageDescription>
    </tns1:Crossed>
  </tns1:LineDetector>
</tns1:RuleEngine>
```

```

      <tt:SimpleItemDescription
Name="VideoAnalyticsConfigurationToken"
      Type="tt:ReferenceToken"/>
    </tt:Source>
    <tt:Data>
      <tt:SimpleItemDescription
Name="ObjectId"
      Type="tt:ObjectRefType"/>
    </tt:Data>
  </tt:MessageDescription>
</tns1:Crossed>
</tns1:LineDetector>
<tns1:FieldDetector wstop:topic="true">
  <tns1:ObjectsInside wstop:topic="true">
    <tt:MessageDescription IsProperty="true">
      <tt:Source>
        <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
      Type="tt:ReferenceToken"/>
        <tt:SimpleItemDescription
Name="VideoAnalyticsConfigurationToken"
      Type="tt:ReferenceToken"/>
        <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
      </tt:Source>
      <tt:Key>
        <tt:SimpleItemDescription
Name="ObjectId"
      Type="tt:ObjectRefType"/>
      </tt:Key>
      <tt:Data>
        <tt:SimpleItemDescription Name="IsInside" Type="xs:boolean"/>
      </tt:Data>
    </tt:MessageDescription>
  </tns1:ObjectsInside>
</tns1:FieldDetector>
</tns1:RuleEngine>

```

15.7.3 Filtre de rubrique

Le dispositif satisfaisant à l'ONVIF doit prendre en charge les expressions de rubrique concrètes définies dans la spécification [WS-Topics]. Cette spécification définit l'identification d'une rubrique spécifique dans des arborescences de rubriques. Le dialecte suivant doit être spécifié lorsqu'une expression de rubrique concrète est utilisée en tant que TopicExpression d'un filtre d'abonnement:

<http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete>

La syntaxe d'expression de rubrique suivante doit être prise en charge par un dispositif.

La syntaxe étend les expressions de rubrique concrètes par une opération "or" et une chaîne de correspondance de sous-arborescence de rubrique. Cette syntaxe étendue permet de sélectionner un TopicSet aléatoire dans un seul abonnement. La syntaxe est décrite de la même façon que les expressions de rubrique de la spécification [WS-Topics 1.3]:

[3] TopicExpression ::= TopicPath ('|' TopicPath)*

[4] TopicPath ::= RootTopic ChildTopicExpression* ('/.')?

[5] RootTopic ::= QName

Si un préfixe d'espace de noms est inclus dans l'élément RootTopic, il doit correspondre à une définition d'espace de noms de rubrique valide, et le nom local doit correspondre au nom d'une rubrique racine définie dans cet espace de noms.

[6] ChildTopicExpression ::= '/' ChildTopicName

[7] ChildTopicName ::= QName | NCName

Le NCName ou la partie locale du QName doit correspondre au nom d'une rubrique dans le chemin descendant depuis l'élément RootTopic, chaque barre oblique indiquant un autre niveau d'éléments de rubrique enfants dans le chemin.

Afin de référencer ce dialecte TopicExpression, l'URI suivant doit être utilisé:

Dialect=http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet

Si TopicExpression se termine par les caractères "//.", cela indique que TopicExpression correspond à une sous-arborescence de rubriques. Par exemple:

"tns1:RuleEngine/FieldDetector//."

Cela identifie la sous-arborescence composée de tns1:RuleEngine/FieldDetector et de tous ses descendants.

Les exemples suivants illustrent l'utilisation de l'élément topicExpression de ConcreteSet:

- Rechercher les notifications qui ont la rubrique VideoAnalytics en tant que rubrique parent:


```
<wsnt:TopicExpression Dialect =
  • "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet" >
    • tns1:VideoAnalytics//.
  • </wsnt:TopicExpression>
```
- Rechercher les notifications qui ont la rubrique VideoAnalytics ou RuleEngine en tant que rubrique parent:


```
<wsnt:TopicExpression Dialect =
  • "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet" >
    • tns1:VideoAnalytics//.|tns1:RuleEngine//.
  • </wsnt:TopicExpression>
```
- Rechercher les notifications générées par un LineDetector ou un FieldDetector:


```
<wsnt:TopicExpression Dialect =
  • "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
  • tns1:RuleEngine/FieldDetector//.|tns1:RuleEngine/LineDetector//.
  • </wsnt:TopicExpression>
```

15.8 Obtention de propriétés d'événement

La spécification [WS-BaseNotification] définit un ensemble de propriétés WS-ResourceProperties FACULTATIVES. Cette spécification ne requiert pas la mise en œuvre de l'interface WS-ResourceProperty. Au lieu de cela, l'interface directe suivante doit être mise en œuvre par un dispositif satisfaisant à l'ONVIF afin de fournir des informations sur les éléments FilterDialect, les fichiers de schéma et les rubriques pris en charge par le dispositif (voir Tableau 213).

Tableau 213 – Commande GetEventProperties

GetEventProperties		Demande-réponse
Nom du message	Description	
GetEventPropertiesRequest	<i>Ceci est un message vide.</i>	
GetEventPropertiesResponse	xs:anyURI TopicNamespaceLocation [1][non limité] xs:boolean FixedTopicSet [1][1] wstop:TopicSetType TopicSet [1][1] xs:anyURI TopicExpressionDialect [1][non limité] xs:anyURI MessageContentFilterDialect [1][non limité] xs:anyURI ProducerPropertiesFilterDialect [0][non limité] xs:anyURI MessageContentSchemaLocation [1][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

Un dispositif satisfaisant à l'ONVIF doit répondre et déclarer si son élément TopicSet est fixe ou non, et indiquer les rubriques fournies et les dialectes pris en charge.

Les éléments TopicExpressionDialect suivants sont obligatoires pour un dispositif satisfaisant à l'ONVIF (voir 15.7.3):

- <http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete>
- <http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet>
- Les éléments MessageContentFilterDialect sont obligatoires pour le dispositif satisfaisant à l'ONVIF (voir 15.5.5):

<http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter>

La présente spécification ne requiert pas la prise en charge d'un élément ProducerPropertiesDialect par un dispositif.

Le langage de description de contenu de message présenté en 15.5.4 permet de référencer des types spécifiques au fournisseur. Pour faciliter l'intégration de ces types dans une application client, l'élément GetEventPropertiesResponse doit énumérer tous les emplacements d'URI vers les fichiers de schéma dont les types sont utilisés dans la description de notifications, avec des éléments MessageContentSchemaLocation. Cette liste doit contenir au moins l'URI du fichier de schéma ONVIF.

15.9 Messages de défaut SOAP

Si un dispositif rencontre un échec pendant le traitement de messages [WS-BaseNotification] provenant d'un client ou d'un gestionnaire d'abonnements, il doit générer un message de défaut SOAP 1.2.

Tous les messages de défaut SOAP 1.2 doivent être générés conformément aux spécifications [WS-BaseNotification] et [WS-Topics].

15.10 Exemple de notification

L'exemple suivant est un profil de communication complet pour des notifications. Il utilise l'interface de notification Real-time Pull-Point pour recevoir des notifications.

15.10.1 GetEventPropertiesRequest

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/GetEventPropertiesRequest
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:GetEventProperties>
  </tet:GetEventProperties>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.2 GetEventPropertiesResponse

Dans cet exemple, la réponse du dispositif utilise l'espace de noms de rubrique ONVIF (dont la description peut être téléchargée à l'adresse <http://www.onvif.org/onvif/ver10/topics/topicsns.xml>). L'ensemble de rubriques ne change pas au cours du temps et est composé de la seule rubrique `tns1:RuleEngine/LineDetector/Crossed`. Le message associé à cette rubrique contient des informations sur le `VideoSourceConfigurationToken`, le `VideoAnalyticsConfigurationToken` et l'objet qui a traversé la limite. Le dispositif prend en charge deux éléments `TopicExpressionDialect`.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wstop="http://docs.basis-open.org/wsn/t-1"
  xmlns:wsnt="http://docs.basis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1"
  xmlns:tns1="http://www.onvif.org/ver10/topics"
  xmlns:tt="http://www.onvif.org/ver10/schema">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/GetEventPropertiesResponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:GetEventPropertiesResponse>
      <tet:TopicNamespaceLocation>
        http://www.onvif.org/onvif/ver10/topics/topicsns.xml
      </tet:TopicNamespaceLocation>
      <wsnt:FixedTopicSet>
        true
      </wsnt:FixedTopicSet>
      <wstop:TopicSet>
        <tns1:RuleEngine>
          <tns1:LineDetector>
            <tns1:Crossed wstop:topic="true">
              <tt:MessageDescription>
                <tt:Source>
                  <tt:SimpleItemDescription
                    Name="VideoSourceConfigurationToken"
                    Type="tt:ReferenceToken"/>
                  <tt:SimpleItemDescription
                    Name="VideoAnalyticsConfigurationToken"
                    Type="tt:ReferenceToken"/>
                </tt:Source>
                <tt:Data>
                  <tt:SimpleItemDescription Name="ObjectId">
```

```

Type="tt:ObjectRefType"/>
    </tt:Data>
  </tt:MessageDescription>
</tnsl:Crossed>
</tnsl:LineDetector>
</tnsl:RuleEngine>
</wstop:TopicSet>
<wsnt:TopicExpressionDialect>
  http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet
</wsnt:TopicExpressionDialect>
<wsnt:TopicExpressionDialect>
  http://docs.oasis-open.org/wsnt/t-1/TopicExpression/ConcreteSet
</wsnt:TopicExpressionDialect>
<wsnt:MessageContentFilterDialect>
  http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter
</wsnt:MessageContentFilterDialect>
<tt:MessageContentSchemaLocation>
  http://www.onvif.org/onvif/ver10/schema/onvif.xsd
</tt:MessageContentSchemaLocation>
</tet:GetEventPropertiesResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.10.3 CreatePullPointSubscription

Un client peut s'abonner à des notifications spécifiques avec les informations provenant des éléments TopicProperties. L'exemple XML suivant présente l'abonnement pour des notifications générées par le moteur de règles du dispositif. Le client réagit uniquement aux notifications qui référencent la VideoAnalyticsConfiguration "2" et la VideoSourceConfiguration "1". Le délai d'expiration de l'abonnement est d'une minute. Si l'abonnement n'est pas explicitement renouvelé ou que les messages ne sont pas extraits régulièrement, ils prennent fin à l'issue de ce délai.

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1"
  xmlns:tnsl="http://www.onvif.org/ver10/topics">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/CreatePullPointSubscriptionRequest
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:CreatePullPointSubscription>
      <tet:Filter>
        <wsnt:TopicExpression
          Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
            tnsl:RuleEngine//.
          </wsnt:TopicExpression>
        <wsnt:MessageContent
          Dialect="http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter">
            boolean(//tt:SimpleItem[@Name="VideoAnalyticsConfigurationToken"
              and @Value="2"] ) and
            boolean(//tt:SimpleItem[@Name="VideoSourceConfigurationToken"
              and @Value="1"] )
          </wsnt:MessageContent>
        </tet:Filter>
        <tet:InitialTerminationTime>
          PT1M
        </tet:InitialTerminationTime>
      </tet:CreatePullPointSubscription>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>

```

15.10.4 CreatePullPointSubscriptionResponse

Si le dispositif accepte l'objet Subscription, il retourne l'URI `http://160.10.64.10/Subscription?Idx=0` qui représente le point terminal de cet abonnement. De plus, le client est informé du `CurrentTime` du dispositif et du `TerminationTime` de l'abonnement créé.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/EventPortType/CreatePullPointSubscriptionResponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:CreatePullPointSubscriptionResponse>
      <tet:SubscriptionReference>
        <wsa:Address>
          http://160.10.64.10/Subscription?Idx=0
        </wsa:Address>
      </tet:SubscriptionReference>
      <wsnt:CurrentTime>
        2008-10-09T13:52:59
      </wsnt:CurrentTime>
      <wsnt:TerminationTime>
        2008-10-09T13:53:59
      </wsnt:TerminationTime>
    </tet:CreatePullPointSubscriptionResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.5 PullMessagesRequest

Le client envoie une `PullMessagesRequest` au point terminal indiqué dans la `CreatePullPointSubscriptionResponse` pour obtenir les notifications correspondant à un certain abonnement. L'exemple de demande suivant contient un délai d'attente (`Timeout`) de cinq (5) secondes et limite à deux (2) le nombre total de messages dans la réponse.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1" >
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/PullPointSubscription/PullMessagesRequest
    </wsa:Action>
    <wsa:To>http://160.10.64.10/Subscription?Idx=0</wsa:To>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:PullMessages>
      <tet:Timeout>
        PT5S
      </tet:Timeout>
      <tet:MessageLimit>
        2
      </tet:MessageLimit>
    </tet:PullMessages>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

15.10.6 PullMessagesResponse

La PullMessageResponse contient deux notifications qui correspondent à l'abonnement. La réponse informe le client que deux objets ont dépassé les limites correspondant aux règles "MyImportantFence1" et "MyImportantFence2".

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
  xmlns:tet="http://www.onvif.org/ver10/events/wsd1"
  xmlns:tns1="http://www.onvif.org/ver10/topics"
  xmlns:tt="http://www.onvif.org/ver10/schema">
  <SOAP-ENV:Header>
    <wsa:Action>
      http://www.onvif.org/ver10/events/wsd1/PullPointSubscription/PullMessagesR
      esponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tet:PullMessagesResponse>
      <tet:CurrentTime>
        2008-10-10T12:24:58
      </tet:CurrentTime>
      <tet:TerminationTime>
        2008-10-10T12:25:58
      </tet:TerminationTime>
      <wsnt:NotificationMessage>
        <wsnt:Topic
          Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
            tns1:RuleEngine/LineDetector/Crossed
          </wsnt:Topic>
          <wsnt:Message>
            <tt:Message UtcTime="2008-10-10T12:24:57.321">
              <tt:Source>
                <tt:SimpleItem Name="VideoSourceConfigurationToken"
                  Value="1"/>
                <tt:SimpleItem Name="VideoAnalyticsConfigurationToken"
                  Value="2"/>
                <tt:SimpleItem Value="MyImportantFence1" Name="Rule"/>
              </tt:Source>
              <tt:Data>
                <tt:SimpleItem Name="ObjectId" Value="15" />
              </tt:Data>
            </tt:Message>
          </wsnt:Message>
        </wsnt:NotificationMessage>
        <wsnt:NotificationMessage>
          <wsnt:Topic
            Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">
              tns1:RuleEngine/LineDetector/Crossed
            </wsnt:Topic>
            <wsnt:Message>
              <tt:Message UtcTime="2008-10-10T12:24:57.789">
                <tt:Source>
                  <tt:SimpleItem Name="VideoSourceConfigurationToken"
                    Value="1"/>
                  <tt:SimpleItem Name="VideoAnalyticsConfigurationToken"
                    Value="2"/>
                  <tt:SimpleItem Value="MyImportantFence2" Name="Rule"/>
                </tt:Source>
                <tt:Data>
                  <tt:SimpleItem Name="ObjectId" Value="19"/>
                </tt:Data>
              </tt:Message>
            </wsnt:Message>
          </wsnt:NotificationMessage>
        </wsnt:NotificationMessage>
      </wsnt:NotificationMessage>
    </tet:PullMessagesResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

```

    </tet:PullMessagesResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.10.7 UnsubscribeRequest

Il faut qu'un client mette fin explicitement à un abonnement avec une UnsubscribeRequest afin que le dispositif puisse libérer immédiatement des ressources. La demande s'adresse au point terminal d'abonnement retourné dans la CreatePullPointSubscriptionResponse.

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" >
  <SOAP-ENV:Header>
    <wsa:Action>
      http://docs.oasis-open.org/wsn/bw-
      2/SubscriptionManager/UnsubscribeRequest
    </wsa:Action>
    <wsa:To>http://160.10.64.10/Subscription?Idx=0</wsa:To>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <wsnt:Unsubscribe/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
UnsubscribeResponse
The Subscription Endpoint is no longer available once the device replies
with an UnsubscribeResponse.
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" >
  <SOAP-ENV:Header>
    <wsa:Action>
      http://docs.oasis-open.org/wsn/bw-
      2/SubscriptionManager/UnsubscribeResponse
    </wsa:Action>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <wsnt:UnsubscribeResponse/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

15.11 Codes de défaut spécifiques au service

Le service d'événement ne définit pas de défauts spécifiques au service autres que ceux définis dans la spécification [WS-BaseNotification].

16 Contrôle PTZ

Le contrôle PTZ comprend des opérations utilisées pour effectuer des commandes de basculement horizontal, basculement vertical et zoom de dispositif. Un dispositif avec fonctionnalité de basculement horizontal, basculement vertical et zoom doit prendre en charge le contrôle PTZ. De même, un dispositif avec seulement une fonctionnalité de zoom ou de basculement horizontal et vertical doit prendre en charge le contrôle PTZ. Le contrôle PTZ est défini en C.9. Les opérations obligatoires sont indiquées sous les descriptions de commande.

Le contrôle PTZ couvre un large éventail de dispositifs de caméra. Un modèle de caméra PTZ dome est supposé être capable de modifier la direction de visée de la caméra indépendamment du zoom.

Le mouvement PTZ est commandé en utilisant un modèle de système de coordonnées. Un dispositif permet de répertorier l'ensemble de systèmes de coordonnées qu'il prend en charge. Cette spécification présente un ensemble d'espaces de coordonnées génériques

applicable à un dispositif PTZ. D'autres systèmes de coordonnées qui sont plus appropriés peuvent être définis pour un matériel dome spécifique. Le contrôle PTZ est applicable aux dispositifs suivants:

- caméra dome ou PTZ;
- codeur vidéo en réseau avec caméra dome ou PTZ raccordée via un port série externe;
- caméra mégapixel fixe avec PTZ numérique;
- caméra fixe avec zoom.

La *structure de contrôle* PTZ est composée de trois blocs principaux:

- *PTZ Node (Nœud PTZ)* – entité PTZ de niveau inférieur mise en correspondance avec le dispositif PTZ et qui spécifie ses fonctionnalités;
- *PTZ Configuration (Configuration PTZ)* – Configuration PTZ des systèmes de coordonnées par défaut et vitesses par défaut pour un nœud PTZ spécifique;
- *PTZ Control Operation (Opération de contrôle PTZ)* – opérations de mouvement, préréglage et auxiliaire.

Le contrôle PTZ est associé à un profil multimédia en incluant la configuration PTZ dans le profil (voir 4.8.2), et *toutes* les opérations de contrôle PTZ sont exécutées en référence à un profil multimédia particulier.

Le contrôle PTZ ne comporte pas d'opérations pour créer ou manipuler des nœuds PTZ. Pour chaque nœud PTZ disponible, le dispositif doit fournir au moins une configuration PTZ assignée à ce nœud PTZ. Cette configuration PTZ peut ensuite être ajoutée à des profils multimédia qui sont utilisés pour contrôler le dome. Chaque profil multimédia ne contient pas plus d'une configuration PTZ. La configuration PTZ et VideoSourceConfiguration sont associés dans le profil multimédia car VideoSourceConfiguration fait référence à la caméra contrôlée par la configuration PTZ.

Un dispositif doté de la fonctionnalité PTZ doit fournir au moins un *profil prêt à l'usage* comprenant une PTZConfiguration qui couvre les réglages les plus basiques et une VideoSourceConfiguration correspondante dès que le dispositif PTZ sous-jacent est prêt à fonctionner.

16.1 Modèle PTZ

Le modèle PTZ regroupe les mouvements possibles de l'unité PTZ dans une composante Pan/Tilt (basculement horizontal/basculement vertical) et une composante Zoom. Pour diriger l'unité PTZ, le service permet d'effectuer des opérations de mouvement absolu, mouvement relatif et mouvement continu. Différents systèmes de coordonnées et unités sont utilisés pour effectuer ces opérations.

Le service PTZ permet d'effectuer une opération AbsoluteMove pour déplacer le dispositif PTZ vers une position absolue. Le service attend la position absolue sous la forme d'un argument en référence à un système de coordonnées absolu. La vitesse du mouvement de basculement horizontal/vertical et du mouvement zoom peut être éventuellement spécifiée. Les valeurs de vitesse sont des scalaires positifs et ne contiennent pas d'information directionnelle. Les vitesses de basculement horizontal et vertical ne peuvent pas être spécifiées séparément sans connaître la position actuelle. Cette approche de spécification d'une position souhaitée produit généralement une action qui n'est ni fluide, ni intuitive.

Une opération RelativeMove est introduite par le service PTZ afin de diriger le dome par rapport à la position actuelle, mais sans qu'il soit nécessaire de connaître la position actuelle. L'opération attend une translation de position en tant qu'argument faisant référence à un système de coordonnées relatif. La présente norme différencie les systèmes de coordonnées relatif et absolu, étant donné que dans certains cas, il n'existe pas de système de

coordonnées absolu pour un système de coordonnées relatif bien défini. Un argument de vitesse optionnel peut être ajouté à l'opération `RelativeMove` avec la même définition que pour l'opération `AbsoluteMove`.

Enfin, le dispositif PTZ peut être déplacé en continu à l'aide de la commande `ContinuousMove` dans une certaine direction et selon une certaine vitesse. Ainsi, un vecteur de vélocité représente à la fois les informations de direction et de vitesse. Cette dernière est exprimée par la longueur du vecteur.

Les coordonnées de basculement horizontal/vertical et de zoom peuvent être spécifiées de manière unique en ajoutant aux coordonnées les URI spatiaux appropriés. Un URI spatial représente de façon unique le système de coordonnées sous-jacent. Le paragraphe 0 définit un ensemble normalisé de systèmes de coordonnées. Un nœud PTZ doit mettre en œuvre ces systèmes de coordonnées si le type de mouvement correspondant est pris en charge par le nœud PTZ. Dans la plupart des cas, la position de basculement horizontal/vertical est représentée par des angles de basculement horizontal et de basculement vertical dans un système de coordonnées sphérique. Un PTZ numérique, fonctionnant sur une caméra mégapixel fixe, peut représenter la direction de visée de la caméra par une position de pixel sur un plan de projection statique. Par conséquent, différents systèmes de coordonnées sont nécessaires dans ce cas pour capturer les mouvements physiques ou virtuels du dispositif PTZ. Ces systèmes de coordonnées, et d'autres, sont définis dans un document séparé intitulé [ONVIF PTZ]. Le nœud PTZ peut éventuellement définir ses propres systèmes de coordonnées spécifiques au dispositif afin de permettre aux NVC de tirer profit des propriétés spécifiques du nœud PTZ en question.

La description du nœud PTZ extraite via l'opération `GetNode` ou `GetNodes` contient tous les systèmes de coordonnées pris en charge par un nœud PTZ spécifique. Chaque système de coordonnées appartient à l'un des groupes suivants:

- `AbsolutePanTiltPositionSpace`;
- `RelativePanTiltTranslationSpace`;
- `ContinuousPanTiltVelocitySpace`;
- `PanTiltSpeedSpace`;
- `AbsoluteZoomPositionSpace`;
- `RelativeZoomTranslationSpace`;
- `ContinuousZoomVelocitySpace`;
- `ZoomSpeedSpace`.

Si le nœud PTZ ne prend pas en charge les systèmes de coordonnées d'un certain groupe, l'opération de mouvement correspondante n'est pas disponible pour ce nœud PTZ. Par exemple, si la liste ne contient pas un groupe `AbsolutePanTiltPositionSpace`, l'opération `AbsoluteMove` doit échouer lorsqu'une position de basculement horizontal/vertical absolu est spécifiée. La section de commande correspondante décrit les espaces requis pour une commande de mouvement spécifique.

16.2 Nœud PTZ

Un dispositif doté de la fonctionnalité PTZ peut avoir plusieurs nœuds PTZ. Les nœuds PTZ peuvent représenter des pilotes PTZ mécaniques, des pilotes PTZ téléchargés ou des pilotes PTZ numériques. Les nœuds PTZ sont les entités de plus bas niveau dans l'API de contrôle PTZ et reflètent les fonctionnalités PTZ prises en charge. Le nœud PTZ est référencé par son nom ou par son jeton de référence. Le service PTZ ne comporte pas d'opérations de création ou de manipulation des nœuds PTZ.

Les propriétés suivantes doivent être disponibles pour tous les nœuds PTZ:

- `Token` – identifiant unique utilisé pour référencer des nœuds PTZ;

- Name – nom donné par l'installateur;
- SupportedPTZSpaces – liste des systèmes de coordonnées disponibles pour le nœud PTZ. Pour chaque système de coordonnées, le nœud PTZ doit spécifier sa propre sélection;
- MaximumNumberOfPresets – toutes les opérations prédéfinies doivent être disponibles pour ce nœud PTZ si un préréglage est pris en charge;
- HomeSupported – opérateur booléen spécifiant la disponibilité d'une position de départ. Si sa valeur est "true", les opérations relatives à la position de départ doivent être disponibles pour le nœud PTZ en question;
- AuxiliaryCommands – liste des commandes auxiliaires prises en charge. Si la liste n'est pas vide, les opérations auxiliaires doivent être disponibles pour ce nœud PTZ.

16.2.1 GetNodes

Un dispositif doté de la fonctionnalité PTZ doit mettre en œuvre cette opération et retourner tous les nœuds PTZ disponibles sur le dispositif (voir Tableau 214).

Tableau 214 – Commande GetNodes

GetNodes		Demande-Réponse	
Nom du message		Description	
GetNodesRequest		<i>Ceci est un message vide.</i>	
GetNodesResponse		<i>Le message de réponse contient une liste des nœuds PTZ existants sur le dispositif.</i> tt:PTZNode PTZNode[0][non limité]	
Codes de défaut		Description	
env:Receiver		<i>PTZ n'est pas pris en charge par le dispositif.</i>	
ter:ActionNotSupported			
ter:PTZNotSupported			

16.2.2 GetNode

Un dispositif doté de la fonctionnalité PTZ doit mettre en œuvre l'opération GetNode et retourner les propriétés du nœud PTZ demandé, s'il existe (voir Tableau 215). Sinon, le dispositif doit répondre par un message de défaut approprié.

Tableau 215 – Commande GetNode

GetNode		Demande-Réponse	
Nom du message		Description	
GetNodeRequest		Ce message contient une référence au PTZNode demandé.	
		tt:ReferenceToken NodeToken[1][1]	
GetNodeResponse		Le message de réponse PTZNode contient le PTZNode demandé.	
		tt:PTZNode PTZNode[1][1]	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoEntity		Pas de PTZNode de ce type sur le dispositif.	
env:Receiver ter:ActionNotSupported ter:PTZNotSupported		PTZ n'est pas pris en charge.	

16.3 Configuration PTZ

PTZConfiguration contient une référence au nœud PTZ auquel il appartient. Cette référence ne peut pas être modifiée par un NVC.

Les éléments suivants font partie de la configuration PTZ:

PTZNodeToken – Référence obligatoire au nœud PTZ auquel la configuration PTZ appartient.

- DefaultAbsolutePanTiltPositionSpace – si le nœud PTZ prend en charge les mouvements de basculement horizontal et vertical absolus, il doit spécifier un espace de position de basculement horizontal/vertical absolu par défaut;
- DefaultRelativePanTiltTranslationSpace – si le nœud PTZ prend en charge les mouvements de basculement horizontal/vertical relatif, il doit spécifier un espace de translation de basculement horizontal/vertical relatif par défaut;
- DefaultContinuousPanTiltVelocitySpace – si le nœud PTZ prend en charge les mouvements continus de basculement horizontal/vertical, il doit spécifier un espace de vitesse de basculement horizontal/vertical continu par défaut;
- DefaultPanTiltSpeedSpace – si le nœud PTZ prend en charge les mouvements absolus ou relatifs, il doit spécifier un espace de vitesse de basculement horizontal/vertical par défaut;
- DefaultAbsoluteZoomPositionSpace – si le nœud PTZ prend en charge les mouvements de zoom absolus, il doit spécifier un espace de position de zoom absolu par défaut;
- DefaultRelativeZoomTranslationSpace – si le nœud PTZ prend en charge les mouvements de zoom relatifs, il doit spécifier un espace de translation de zoom relatif par défaut;

- `DefaultContinuousZoomVelocitySpace` – si le nœud PTZ prend en charge les mouvements de zoom continu, il doit spécifier un espace de vitesse de zoom continu par défaut;
- `DefaultZoomSpeedSpace` – si le nœud PTZ prend en charge les mouvements absolus ou relatifs, il doit spécifier un espace de vitesse de zoom par défaut;
- `DefaultPTZSpeed` – si le nœud PTZ prend en charge les mouvements PTZ absolus ou relatifs, il doit spécifier des vitesses de basculement horizontal/vertical et de zoom par défaut correspondantes;
- `DefaultPTZTimeout` – si le nœud PTZ prend en charge des mouvements continus, il doit spécifier un délai d'expiration par défaut, au-delà duquel le mouvement s'arrête;
- `PanTiltLimits` – il convient que l'élément `PanTiltLimits` soit présent pour un nœud PTZ qui prend en charge un basculement horizontal/vertical absolu. Si cet élément est présent, il signale la prise en charge de limites de basculement horizontal/vertical configurables. Si les limites sont activées, les mouvements de basculement horizontal/vertical doivent toujours rester dans la plage spécifiée. Les limites de basculement horizontal/vertical sont désactivées en définissant les limites à $-\text{INF}$ ou $+\text{INF}$;
- `ZoomLimits` – il convient que l'élément de limites de zoom soit présent pour un nœud PTZ qui prend en charge un zoom absolu. Si l'élément est présent, il signale la prise en charge de limites de zoom configurables. Si les limites sont activées, les mouvements de zoom doivent toujours rester dans la plage spécifiée. Les limites de zoom sont désactivées en définissant les limites à $-\text{INF}$ et $+\text{INF}$.

Les espaces de position/translation/vitesse par défaut sont introduits afin de permettre aux NVC d'envoyer des demandes de mouvement sans avoir à spécifier un certain système de coordonnées. Les vitesses par défaut sont introduites pour contrôler la vitesse des demandes de mouvement (absolu, relatif, prédéfini) si aucune vitesse explicite n'a été définie.

L'intervalle de basculement horizontal et de basculement vertical admis pour les limites de basculement horizontal/vertical est défini par une plage d'espace bidimensionnel qui correspond à un espace de position de basculement horizontal/vertical absolu spécifique. Au moins un espace de position de basculement horizontal/vertical est requis par le nœud PTZ pour prendre en charge les limites de basculement horizontal/vertical. Les limites s'appliquent à tous les mouvements de basculement horizontal/vertical absolus, relatifs et continus pris en charge. Les limites doivent être vérifiées dans le système de coordonnées pour lequel elles ont été spécifiées. Cela signifie que même si les mouvements sont spécifiés dans un système de coordonnées différent, les mouvements demandés doivent être transformés dans le système de coordonnées des limites dans lequel les limites peuvent être contrôlées. Lorsqu'un mouvement relatif ou continu est spécifié, qui est susceptible de sortir des limites spécifiées, il faut que l'unité PTZ se déplace conformément aux limites spécifiées. Il faut que les limites de zoom soient interprétées en conséquence.

16.3.1 `GetConfigurations`

Un dispositif doté de la fonctionnalité PTZ doit retourner toutes les `PTZConfiguration` disponibles grâce à l'opération `GetConfigurations` (voir Tableau 216).

Tableau 216 – Commande GetConfigurations

GetConfigurations		Demande-Réponse	
Nom du message		Description	
GetConfigurations		<i>Ceci est un message vide.</i>	
GetConfigurationsResponse		<i>La réponse contient toutes les PTZConfiguration existantes sur le dispositif.</i> tt:PTZConfiguration PTZConfiguration [0][non limité]	
Codes de défaut		Description	
env: Receiver ter: ActionNotSupported ter: PTZNotSupported		<i>PTZ n'est pas pris en charge.</i>	

16.3.2 GetConfiguration

Un dispositif doté de la fonctionnalité PTZ doit retourner la configuration PTZ demandée, si elle existe, par l'intermédiaire de l'opération GetConfiguration (voir Tableau 217).

Tableau 217 – Commande GetConfiguration

GetConfiguration		Demande-Réponse	
Nom du message		Description	
GetConfigurationRequest		<i>Ce message contient une référence à la PTZConfiguration demandée.</i> tt:ReferenceToken ConfigurationToken[1][1]	
GetConfigurationResponse		<i>La réponse contient la PTZConfiguration demandée.</i> tt:PTZConfiguration PTZConfiguration [1][1]	
Codes de défaut		Description	
env: Sender ter: InvalidArgVal ter: NoConfig		<i>La configuration demandée n'existe pas.</i>	
env: Receiver ter: ActionNotSupported ter: PTZNotSupported		<i>PTZ n'est pas pris en charge par le dispositif.</i>	

16.3.3 GetConfigurationOptions

Un dispositif doté de la fonctionnalité PTZ doit mettre en œuvre l'opération GetConfigurationOptions. Celle-ci retourne la liste des systèmes de coordonnées pris en charge comprenant leurs limitations d'intervalle. Par conséquent, les options PEUVENT différer suivant que la configuration PTZ est assignée à un profil contenant une configuration de source vidéo ou non. Dans ce cas, les options peuvent contenir en outre des systèmes de coordonnées faisant référence au système de coordonnées d'image décrit par la configuration de source vidéo. Chaque système de coordonnées listé appartient à l'un des groupes mentionnés en 16.1. Si le nœud PTZ prend en charge les mouvements continus, il doit retourner un intervalle de délai d'attente dans lequel les délais d'attente sont acceptés par le nœud PTZ (voir Tableau 218).

Tableau 218 – Commande GetConfigurationOptions

GetConfigurationOptions		Demande-Réponse
Nom du message	Description	
GetConfigurationOptions-Request	Ce message contient un jeton pour une configuration PTZ. ConfigurationToken spécifie une configuration existante à laquelle les options sont destinées. tt:ReferenceToken ConfigurationToken [1][1]	
GetConfigurationOptions-Response	<i>Ce message contient les options de configuration PTZ.</i> tt:PTZConfigurationOptions PTZConfigurationOptions[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée n'existe pas.	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	PTZ n'est pas pris en charge.	

16.3.4 SetConfiguration

Un dispositif doté de la fonctionnalité PTZ doit mettre en œuvre l'opération SetConfiguration (voir Tableau 219). Le drapeau ForcePersistence indique si les modifications persistent après le redémarrage du dispositif.

Tableau 219 – Commande SetConfiguration

SetConfiguration		Demande-Réponse	
Nom du message		Description	
SetConfigurationRequest		<i>L'élément PTZConfiguration contient la configuration PTZ modifiée. La configuration doit exister dans le dispositif.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:PTZConfiguration PTZConfiguration[1][1] xs:boolean ForcePersistence[1][1]	
SetConfigurationResponse		Ceci est un message vide	
Codes de défaut		Description	
env:Sender ter:InvalidArgVal ter:NoConfig		La configuration n'existe pas.	
env:Sender ter:InvalidArgVal ter:ConfigModify		Les paramètres de configuration ne peuvent pas être définis.	
env:Receiver ter:Action ter:ConfigurationConflict		Les nouveaux réglages sont en conflit avec d'autres utilisations de la configuration.	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported		PTZ n'est pas pris en charge.	

16.4 Opérations de mouvement

Le présent paragraphe décrit les trois opérations de mouvement de l'unité PTZ: absolu, relatif ou continu. Toutes les opérations requièrent un *ProfileToken* référençant un profil multimédia comprenant une PTZConfiguration.

Toutes les commandes de mouvement sont non bloquantes, ce qui signifie qu'elles n'attendent pas que l'opération de mouvement demandée soit terminée. La dernière opération de mouvement peut être supplantée en envoyant une autre demande de mouvement.

16.4.1 AbsoluteMove

Si un nœud PTZ prend en charge des mouvements de basculement horizontal/vertical absolus ou de zoom absolus, il doit prendre en charge l'opération AbsoluteMove (voir Tableau 220). L'argument Position de cette commande spécifie la position absolue vers laquelle l'unité PTZ se déplace. Il comporte un élément de basculement horizontal/vertical optionnel et un élément de zoom optionnel. Si la position de basculement horizontal/vertical est omise, le mouvement de basculement horizontal/vertical en cours ne doit PAS être affecté par cette commande. Cela est également vrai pour la position de zoom.

Les espaces référencés dans le paramètre Position doivent être des espaces de position absolue pris en charge par le nœud PTZ. Si l'information d'espace est omise, l'espace par défaut correspondant de la configuration PTZ, qui fait partie du profil multimédia spécifié, est utilisé. Un dispositif peut prendre en charge des mouvements de basculement horizontal/vertical absolus, des mouvements de zoom absolus ou des mouvements non absolus en fournissant uniquement des espaces de position absolue pour les cas pris en charge.

Un argument Speed existant supprime la DefaultSpeed (vitesse par défaut) de la configuration PTZ correspondante pendant le mouvement vers la position demandée. Si des espaces sont référencés dans l'argument Speed, il doit s'agir d'espaces de vitesse pris en charge par le nœud PTZ.

L'opération doit échouer si la position absolue demandée ne peut pas être atteinte.

Tableau 220 – Commande AbsoluteMove

AbsoluteMove		Demande-Réponse
Nom du message	Description	
AbsoluteMoveRequest	<i>Ce message contient une référence au profil multimédia, un vecteur Position spécifiant la position cible absolue et un élément Speed facultatif.</i> tt:ReferenceToken ProfileToken [1][1] tt:PTZVector Position [1][1] tt:PTZSpeed Speed [0][1]	
AbsoluteMoveResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	<i>Un espace est référencé dans un argument qui n'est pas pris en charge par le nœud PTZ.</i>	
env:Sender ter:InvalidArgVal ter:InvalidPosition	<i>La position demandée est hors limites.</i>	
env:Sender ter:InvalidArgVal ter:InvalidSpeed	<i>La vitesse demandée est hors limites.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.4.2 RelativeMove

Si un nœud PTZ prend en charge les mouvements de basculement horizontal/vertical relatifs ou de zoom relatifs, il doit prendre en charge l'opération RelativeMove (voir Tableau 221). L'argument Translation de cette opération spécifie la différence entre la position actuelle et la position à laquelle le dispositif PTZ a reçu l'instruction de se déplacer. L'opération est divisée en un élément de basculement horizontal/vertical facultatif et un élément de zoom facultatif. Si l'élément de basculement horizontal/vertical est omis, le mouvement de basculement horizontal/vertical en cours ne doit PAS être affecté par cette commande. Cela est également vrai pour l'élément de zoom.

Les espaces référencés dans l'élément Translation doivent être des espaces de translation pris en charge par le nœud PTZ. Si l'information d'espace est omise pour l'argument Translation, l'espace par défaut correspondant de la configuration PTZ, qui fait partie du profil multimédia spécifié, est utilisé. Un dispositif peut prendre en charge des mouvements de basculement horizontal/vertical relatifs, des mouvements de zoom relatifs ou des mouvements non relatifs en fournissant uniquement des espaces de translation pour les cas pris en charge.

Un argument Speed existant supplante DefaultSpeed de la configuration PTZ correspondante pendant le mouvement de translation demandé. Si des espaces sont référencés dans l'argument Speed, il doit s'agir d'espaces de vitesse pris en charge par le nœud PTZ.

La commande peut être utilisée pour arrêter l'unité PTZ à sa position actuelle en envoyant des valeurs nulles pour le basculement horizontal/vertical et le zoom. L'arrêt doit avoir exactement le même effet, quel que soit l'espace relatif référencé.

Si la translation demandée conduit à une position absolue qui ne peut pas être atteinte, le nœud PTZ doit se déplacer vers une position pouvant être atteinte à la limite des positions valides.

Tableau 221 – Commande RelativeMove

RelativeMove		Demande-Réponse
Nom du message	Description	
RelativeMoveRequest	<i>Ce message contient une référence au profil multimédia, un vecteur de Translation de position par rapport à la position actuelle et un paramètre Speed facultatif.</i> tt:ReferenceToken ProfileToken [1][1] tt:PTZVector Translation [1][1] tt:PTZSpeed Speed [0][1]	
RelativeMoveResponse	<i>Ceci est un message vide.</i>	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoProfile		Le ProfileToken (jeton de profil) demandé n'existe pas.
env:Sender ter:InvalidArgVal ter:NoPTZProfile		Le jeton de profil demandé ne référence pas une configuration PTZ.
env:Sender ter:InvalidArgVal ter:SpaceNotSupported		Un espace est référencé dans un argument qui n'est pas pris en charge par le nœud PTZ.
env:Sender ter:InvalidArgVal ter:InvalidTranslation		La translation demandée est hors limites.
env:Sender ter:InvalidArgVal ter:InvalidSpeed		La vitesse demandée est hors limites.
env: Receiver ter:ActionNotSupported ter:PTZNotSupported		PTZ n'est pas pris en charge.

16.4.3 ContinuousMove

Un dispositif doté de la fonctionnalité PTZ doit prendre en charge les mouvements continus (voir Tableau 222). L'argument Velocity de cette commande spécifie une valeur de vitesse signée pour le basculement horizontal, le basculement vertical et le zoom. L'élément Pan/Tilt combiné est optionnel et l'élément Zoom lui-même est optionnel. Si l'élément de basculement horizontal/vertical est omis, le mouvement de basculement horizontal/vertical en cours ne doit PAS être affecté par cette commande. Cela est également vrai pour l'élément de zoom. Les espaces référencés dans l'élément Velocity doivent être des espaces de vitesse pris en charge par le nœud PTZ. Si l'information d'espace est omise pour l'argument Velocity, l'espace par défaut correspondant de la configuration PTZ, qui fait partie du profil multimédia spécifié, est utilisé. Un dispositif PEUT prendre en charge des mouvements de basculement horizontal/vertical continus et/ou des mouvements de zoom continus en fournissant uniquement des espaces de vitesse pour les cas pris en charge.

Un argument Timeout existant supprime le paramètre DefaultPTZTimeout de la configuration PTZ correspondante pendant cette opération de mouvement. Le paramètre Timeout spécifie la durée de déplacement du nœud PTZ.

La commande peut être utilisée pour arrêter le dispositif PTZ à sa position actuelle en envoyant des valeurs nulles pour les paramètres de basculement horizontal/vertical et de zoom. L'arrêt doit avoir exactement le même effet, quel que soit l'espace de vitesse référencé. Cette commande a le même effet sur un mouvement continu que la commande d'arrêt décrite en 16.4.4.

Si la vitesse demandée conduit à des positions absolues qui ne peuvent pas être atteintes, le nœud PTZ doit se déplacer vers une position pouvant être atteinte à la limite de l'intervalle autorisé. Une application classique de l'opération de mouvement continu est le contrôle PTZ à l'aide d'une manette.

Tableau 222 – Commande ContinuousMove

ContinuousMove		Demande-Réponse
Nom du message	Description	
ContinuousMoveRequest	<i>Ce message contient une référence au profil multimédia, un vecteur Velocity spécifiant la vitesse de basculement horizontal/vertical et de zoom et un paramètre Timeout.</i> tt:ReferenceToken ProfileToken [1][1] tt:PTZSpeed Velocity [1][1] xs:duration Timeout [0][1]	
ContinuousMoveResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	<i>Un espace est référencé dans un argument qui n'est pas pris en charge par le nœud PTZ.</i>	
env:Sender ter:InvalidArgVal ter:TimeoutNotSupported	<i>L'argument de délai d'attente spécifié n'est pas dans l'intervalle de délai d'attente pris en charge.</i>	
env:Sender ter:InvalidArgVal ter:InvalidVelocity	<i>La vitesse demandée est hors limites.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.4.4 Stop

Un dispositif doté de la fonctionnalité PTZ doit prendre en charge l'opération Stop (voir Tableau 223). Si aucun argument de filtre Stop n'est présent, cette commande arrête tous les

mouvements de basculement horizontal, de basculement vertical et de zoom en cours. L'opération Stop peut être filtrée pour arrêter un mouvement spécifique en définissant l'argument d'arrêt correspondant.

Tableau 223 – Commande Stop (PTZ)

Stop	Demande-Réponse
Nom du message	Description
StopRequest	<i>Ce message contient une référence au MediaProfile et des paramètres qui indiquent quelle opération il convient d'arrêter.</i> tt:ReferenceToken ProfileToken[1][1] xs:boolean PanTilt [0][1] xs:boolean Zoom0[1]
StopResponse	<i>Ceci est un message vide.</i>
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>

16.4.5 GetStatus

Un dispositif doté de la fonctionnalité PTZ doit être capable de rapporter son statut PTZ grâce à la commande GetStatus (voir Tableau 224). Le statut PTZ contient les informations suivantes:

- Position (facultatif) – spécifie la position absolue de l'unité PTZ avec les références d'espace. Les espaces absolus par défaut de la configuration PTZ correspondante doivent être référencés dans l'élément Position;
- MoveStatus (facultatif) – indique si l'unité de dispositif de basculement horizontal/basculement vertical/zoom est actuellement en mouvement, au repos ou dans un état inconnu;
- Error (facultatif) – déclare une erreur PTZ en cours;
- UTC Time – spécifie l'heure TUC à laquelle ce statut a été généré.

Tableau 224 – Commande GetStatus (PTZ)

GetStatus		Demande-Réponse
Nom du message	Description	
GetStatusRequest	Ce message contient une référence au profil multimédia auquel il convient de demander le PTZStatus.	
	tt:ReferenceToken ProfileToken[1][1]	
GetStatusResponse	Ce message contient le PTZStatus pour le MediaProfile demandé.	
	tt:PTZStatus PTZStatus[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	Le profil demandé n'existe pas.	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	Le jeton de profil demandé ne référence pas une configuration PTZ.	
env:Receiver ter:Action ter:NoStatus	Aucun statut PTZ n'est disponible dans le profil multimédia demandé.	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	PTZ n'est pas pris en charge.	

16.5 Opérations de préréglage

Le présent paragraphe décrit les opérations de gestion des préréglages d'un nœud PTZ. Ces opérations doivent être mises en œuvre pour les nœuds PTZ prenant en charge les préréglages. Toutes les opérations requièrent un *ProfileToken* référençant un profil multimédia comprenant une PTZConfiguration.

16.5.1 SetPreset

La commande SetPreset permet de sauvegarder les paramètres de position de dispositif *en cours*, de sorte que le dispositif puisse se déplacer vers la position de préréglage sauvegardée grâce à l'opération GotoPreset (voir Tableau 225).

Afin de créer un nouveau préréglage, la SetPresetRequest (demande de définition de préréglage) ne contient pas de PresetToken (jeton de préréglage). Si la création aboutit avec succès, la réponse contient le PresetToken qui identifie de façon unique le préréglage. Un préréglage existant peut être remplacé en spécifiant le PresetToken (jeton de préréglage) du préréglage correspondant. Dans les deux cas (remplacement ou création), un PresetName (nom de préréglage) optionnel peut être spécifié. L'opération échoue si le dispositif PTZ est en mouvement pendant l'opération SetPreset.

Le dispositif PEUT sauvegarder en interne les états supplémentaires tels que les propriétés d'imagerie dans le préréglage PTZ, qu'il convient de rappeler ensuite dans l'opération GotoPreset (aller au préréglage).

Tableau 225 – Commande SetPreset

SetPreset		Demande-Réponse
Nom du message	Description	
SetPresetRequest	<i>Ce message contient une référence au MediaProfile et le nom ou jeton demandé pour le préréglage.</i> tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[0][1] xs:string PresetName[0][1]	
SetPresetResponse	<i>Ce message contient une référence au préréglage qui a été défini.</i> tt:ReferenceToken PresetToken[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:PresetExist	<i>Le nom demandé existe déjà pour un autre préréglage.</i>	
env:Sender ter:InvalidArgVal ter:InvalidPresetName	<i>Le PresetName (nom de préréglage) est trop long ou contient des caractères non valides.</i>	
env:Receiver ter:Action ter:MovingPTZ	<i>Le préréglage ne peut pas être défini pendant que l'unité PTZ est en mouvement.</i>	
env:Receiver ter:Action ter:TooManyPresets	<i>Le nombre maximal de préréglages est atteint.</i>	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoToken	<i>Le jeton de préréglage demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.5.2 GetPresets

L'opération GetPresets (voir Tableau 226) retourne les préréglages enregistrés constitués des éléments suivants:

- Token (jeton) – identifiant unique utilisé pour référencer le préréglage;
- Name (nom) – nom mnémonique optionnel;
- PTZ Position (position PTZ) – position absolue facultative. Si le nœud PTZ prend en charge les espaces de position de basculement horizontal/vertical absolu, la position de basculement horizontal/vertical doit être spécifiée. Si le nœud PTZ prend en charge les espaces de position de zoom absolu, la position de zoom doit être spécifiée.

Tableau 226 – Commande GetPresets

GetPresets		Demande-Réponse
Nom du message	Description	
GetPresetsRequest	<i>Ce message contient une référence au MediaProfile dans lequel il convient que l'opération soit effectuée.</i> tt:ReferenceToken ProfileToken[1][1]	
GetPresetsResponse	<i>Ce message contient une liste de préréglages qui sont disponibles pour le MediaProfile demandé.</i> tt:PTZPreset Preset[0][non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.5.3 GotoPreset

L'opération GotoPreset rappelle un préréglage déjà défini (voir Tableau 227). Si le paramètre de vitesse est omis, la vitesse par défaut de la configuration PTZ correspondante doit être utilisée. Le paramètre de vitesse peut être spécifié uniquement si des espaces de vitesse sont disponibles pour le nœud PTZ. La commande GotoPreset est une opération non bloquante, qui peut être interrompue par d'autres commandes de mouvement.

Tableau 227 – Commande GotoPreset

GotoPreset		Demande-Réponse
Nom du message	Description	
GotoPresetRequest	<i>Ce message contient une référence au MediaProfile dans lequel il convient que le déplacement au préréglage identifié par son jeton se produise.</i> tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[1][1] tt:PTZSpeed Speed[0][1]	
GotoPresetResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoToken	<i>Le jeton de préréglage demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:SpaceNotSupported	<i>Un espace est référencé dans un argument qui n'est pas pris en charge par le nœud PTZ.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env:Sender ter:InvalidArgs ter:InvalidSpeed	<i>La vitesse demandée est hors limites.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.5.4 RemovePreset

L'opération RemovePreset supprime un préréglage précédemment défini (voir Tableau 228).

Tableau 228 – Commande RemovePreset

RemovePreset		Demande-Réponse
Nom du message	Description	
RemovePresetRequest	<i>Ce message contient une référence au MediaProfile dans lequel il convient que le préréglage identifié par le jeton soit supprimé.</i> tt:ReferenceToken ProfileToken[1][1] tt:ReferenceToken PresetToken[1][1]	
RemovePresetResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoToken	<i>Le jeton de préréglage demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.6 Opérations de position de départ

Le présent paragraphe décrit les opérations utilisées pour gérer la position de départ d'un nœud PTZ. Ces opérations doivent être mises en œuvre pour des nœuds PTZ prenant en charge les positions de départ. Toutes les opérations requièrent un *ProfileToken* référençant un profil multimédia comprenant une PTZConfiguration.

La position de "départ" PEUT être définie par l'opération SetHome ou est une position fixe de l'unité PTZ.

16.6.1 GotoHomePosition

Cette opération déplace le dome à sa position de départ (voir Tableau 229). Si le paramètre de vitesse est omis, la vitesse par défaut de la configuration PTZ correspondante doit être utilisée. Le paramètre de vitesse peut uniquement être spécifié si les espaces de vitesse sont disponibles pour le nœud PTZ. La commande est non bloquante et peut être interrompue par d'autres commandes de mouvement.

Tableau 229 – Commande GotoHomePosition

GotoHomePosition		Demande-Réponse
Nom du message	Description	
GotoHomePositionRequest	<i>Ce message contient une référence au MediaProfile dans lequel il convient que l'opération soit effectuée.</i> tt:ReferenceToken ProfileToken[1][1] tt:PTZSpeed Speed[0][1]	
GotoHomePositionResponse	<i>Ceci est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Receiver ter:Action ter:NoHomePosition	<i>Aucune position de départ n'a été définie pour ce profil.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.6.2 SetHomePosition

L'opération SetHome sauvegarde les paramètres de position *en cours* en tant que position de départ, de sorte que l'opération GotoHome puisse demander que le dispositif se déplace vers la position de départ (voir Tableau 230).

La commande SetHomePosition doit retourner un échec si la position de "départ" est fixe et ne peut pas être remplacée. Si la commande SetHomePosition aboutit, il doit être possible de rappeler la position de départ à l'aide de la commande GotoHomePosition.

Tableau 230 – Commande SetHomePosition

SetHomePosition		Demande-Réponse
Nom du message	Description	
SetHomePositionRequest	<i>Ce message contient une référence au MediaProfile dans lequel il convient que la position de départ soit définie.</i> tt:ReferenceToken ProfileToken[1][1]	
SetHomePositionResponse	<i>Ce message est vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoProfile	<i>Le ProfileToken (jeton de profil) demandé n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:NoPTZProfile	<i>Le jeton de profil demandé ne référence pas une configuration PTZ.</i>	
env:Receiver ter:Action ter:CannotOverwriteHome	<i>La position de départ est fixe et ne peut pas être remplacée.</i>	
env: Receiver ter:ActionNotSupported ter:PTZNotSupported	<i>PTZ n'est pas pris en charge.</i>	

16.7 Opérations auxiliaires

16.7.1 Généralités

Le présent paragraphe décrit les opérations de gestion des commandes auxiliaires d'un nœud PTZ (une lampe infrarouge (IR), un chauffage ou un dispositif d'essuyage, par exemple).

Ces opérations doivent être mises en œuvre pour des nœuds PTZ indiquant des commandes auxiliaires dans les propriétés de nœud. Toutes les opérations requièrent un *ProfileToken* référant un profil multimédia comprenant une PTZConfiguration.

16.7.2 SendAuxiliaryCommand

Cette opération est utilisée pour appeler une opération auxiliaire sur le dispositif (voir Tableau 231). Les commandes prises en charge peuvent être extraites via les propriétés de nœud PTZ. Il convient que la commande auxiliaire corresponde à la commande prise en charge figurant dans le nœud PTZ. Aucune autre syntaxe n'est prise en charge. Si le nœud PTZ permet de répertorier la commande *irlampon*, l'argument *AuxiliaryCommand* est *irlampon*.

L'opération `SendAuxiliaryCommand` doit être mise en œuvre si le nœud PTZ prend en charge les commandes auxiliaires.

Tableau 231 – Commande `SendAuxiliary`

SendAuxiliaryCommand		Demande-Réponse
Nom du message	Description	
SendAuxiliaryCommandRequest	<i>Ce message contient une référence au <code>MediaProfile</code> dans lequel il convient d'effectuer la demande auxiliaire, et les données de demande auxiliaire.</i> tt:ReferenceToken ProfileToken[1][1] tt:AuxiliaryData AuxiliaryData[1][1]	
SendAuxiliaryCommandResponse	<i>La réponse contient la réponse auxiliaire.</i> tt:AuxiliaryData AuxiliaryResponse[1][1]	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoProfile		Le <code>ProfileToken</code> (jeton de profil) demandé n'existe pas.
env:Sender ter:InvalidArgVal ter:NoPTZProfile		Le jeton de profil demandé ne référence pas une configuration PTZ.
env: Receiver ter:ActionNotSupported ter:PTZNotSupported		PTZ n'est pas pris en charge.

16.8 Espaces PTZ prédéfinis

Des espaces sont utilisés pour spécifier des mouvements absolus, relatifs et continus. Les mouvements absolus requièrent une position absolue, les mouvements relatifs étant spécifiés par une translation de position. Les mouvements continus requièrent la spécification d'une vitesse (mouvement relatif au cours du temps). Pour ces trois cas, différents systèmes de coordonnées sont utilisés décrivant le mouvement souhaité. Les espaces génériques ne spécifient pas de façon absolue le modèle PTZ sous-jacent, de sorte qu'il peut être appliqué à tout matériel PTZ. Les espaces supplémentaires sont définis dans le document [ONVIF PTZ].

16.8.1 Espaces de position absolue

16.8.1.1 Espace de position de basculement horizontal/vertical générique

L'espace de position de basculement horizontal/vertical générique doit être fourni par chaque nœud PTZ qui prend en charge un basculement horizontal/vertical absolu, étant donné qu'il n'est pas lié à un intervalle physique spécifique. Il convient plutôt que l'intervalle soit défini

comme étant l'intervalle total de l'unité PTZ normalisé à la plage comprise entre -1 et 1, conduisant à la description d'espace suivante:

```
<tt:AbsolutePanTiltPositionSpace>

  <tt:SpaceURI>

    http://www.onvif.org/ver10/tptz/PanTiltSpaces/PositionGenericSpace

  </tt:SpaceURI>

  <tt:XRange>

    <tt:Min>-1.0</tt:Min>

    <tt:Max>1.0</tt:Max>

  </tt:XRange>

  <tt:YRange>

    <tt:Min>-1.0</tt:Min>

    <tt:Max>1.0</tt:Max>

  </tt:YRange>

</tt:AbsolutePanTiltPositionSpace>
```

16.8.1.2 Espace de position de zoom générique

L'espace de position de zoom générique doit être fourni par chaque nœud PTZ qui prend en charge un zoom absolu, dans la mesure où il n'est pas associé à un intervalle physique spécifique. Il convient plutôt que l'intervalle soit défini comme étant l'intervalle total du zoom normalisé à l'intervalle compris entre 0 (large) et 1 (zoomé). Aucune hypothèse n'est faite quant à la façon dont l'intervalle de zoom générique est mis en correspondance avec le grossissement, FOV ou une autre dimension de zoom physique. Cela donne lieu à la description d'espace suivante:

```
<tt:AbsoluteZoomPositionSpace>

  <tt:SpaceURI>

    http://www.onvif.org/ver10/tptz/ZoomSpaces/PositionGenericSpace

  </tt:SpaceURI>

  <tt:XRange>

    <tt:Min>0.0</tt:Min>

    <tt:Max>1.0</tt:Max>

  </tt:XRange>

</tt:AbsoluteZoomPositionSpace>
```

16.8.2 Espaces de translation relative

Un espace de translation de basculement horizontal/vertical relative déplace l'unité PTZ d'une certaine translation dans une certaine direction sans connaître la position de basculement horizontal/vertical en cours de la caméra.

16.8.2.1 Espace de translation de basculement horizontal/vertical générique

L'espace de translation de basculement horizontal/vertical générique doit être fourni par chaque nœud PTZ qui prend en charge le basculement horizontal/vertical relatif, dans la mesure où il n'est pas associé à un intervalle physique spécifique. Il convient plutôt que l'intervalle soit défini comme étant l'intervalle de translation positive et négative total de l'unité PTZ normalisé à la plage comprise entre -1 et 1, où une translation positive désigne une rotation horaire ou un mouvement vers la droite/le haut conduisant à la description d'espace suivante:

```
<tt:RelativePanTiltTranslationSpace>
```

```
  <tt:SpaceURI>
```

```
http://www.onvif.org/ver10/tptz/PanTiltSpaces/TranslationGenericSpace
```

```
  </tt:SpaceURI>
```

```
  <tt:XRange>
```

```
    <tt:Min>-1.0</tt:Min>
```

```
    <tt:Max>1.0</tt:Max>
```

```
  </tt:XRange>
```

```
  <tt:YRange>
```

```
    <tt:Min>-1.0</tt:Min>
```

```
    <tt:Max>1.0</tt:Max>
```

```
  </tt:YRange>
```

```
</tt:RelativePanTiltTranslationSpace>
```

16.8.2.2 Espace de translation de zoom générique

L'espace de translation de zoom générique doit être fourni par chaque nœud PTZ qui prend en charge le zoom relatif, dans la mesure où il n'est pas associé à un intervalle physique spécifique. Il convient plutôt que l'intervalle absolu correspondant soit défini comme étant l'intervalle de translation positive et négative total de la fonction Zoom normalisé à la plage comprise entre -1 et 1, où une translation positive correspond à un mouvement dans la direction d'un zoom positif. La translation est signée de manière à indiquer la direction (négatif pour large, positif pour zoom positif). Aucune hypothèse n'est faite quant à la façon dont l'intervalle de zoom générique est mis en correspondance avec le grossissement, FOV ou une autre dimension de zoom physique. Cela donne lieu à la description d'espace suivante:

```
<tt:RelativeZoomTranslationSpace>
```

```
  <tt:SpaceURI>
```

<http://www.onvif.org/ver10/tptz/ZoomSpaces/TranslationGenericSpace>

</tt:SpaceURI>

<tt:XRange>

<tt:Min>-1.0</tt:Min>

<tt:Max>1.0</tt:Max>

</tt:XRange>

</tt:RelativeZoomTranslationSpace>

16.8.3 Espaces de vitesse continue

Les espaces de vitesse continue sont utilisés pour déplacer en continu l'unité PTZ dans une certaine direction.

16.8.3.1 Espace de vitesse de basculement horizontal/vertical générique

L'espace de vitesse de basculement horizontal/vertical générique doit être fourni par chaque nœud PTZ, dans la mesure où il n'est pas associé à un intervalle physique spécifique. Il convient plutôt que l'intervalle soit défini comme étant un intervalle de la vitesse de l'unité PTZ normalisé à l'intervalle compris entre -1 et 1, une vitesse positive correspondant à une rotation horaire ou un mouvement vers la droite/le haut. Une vitesse signée peut être indépendamment spécifiée pour la composante de basculement horizontal et de basculement vertical, conduisant à la description d'espace suivante:

<tt:ContinuousPanTiltVelocitySpace>

<tt:SpaceURI>

<http://www.onvif.org/ver10/tptz/PanTiltSpaces/VelocityGenericSpace>

</tt:SpaceURI>

<tt:XRange>

<tt:Min>-1.0</tt:Min>

<tt:Max>1.0</tt:Max>

</tt:XRange>

<tt:YRange>

<tt:Min>-1.0</tt:Min>

<tt:Max>1.0</tt:Max>

</tt:YRange>

</tt:ContinuousPanTiltVelocitySpace>

16.8.3.2 Espace de vitesse de zoom générique

L'espace de vitesse de zoom générique spécifie une vitesse de facteur de zoom sans connaître le modèle physique sous-jacent. Il convient que l'intervalle soit normalisé entre -1 et 1, une vitesse positive correspondant à une direction de zoom positif. Une description d'espace de vitesse de zoom générique se présente comme suit:

```
<tt:ContinuousZoomVelocitySpace>
```

```
  <tt:SpaceURI>
```

```
http://www.onvif.org/ver10/tptz/ZoomSpaces/VelocityGenericSpace
```

```
  </tt:SpaceURI>
```

```
  <tt:XRange>
```

```
    <tt:Min>-1.0</tt:Min>
```

```
    <tt:Max>1.0</tt:Max>
```

```
  </tt:XRange>
```

```
</tt:ContinuousZoomVelocitySpace>
```

16.8.4 Espaces de vitesse

Les espaces de vitesse spécifient la vitesse d'un mouvement de basculement horizontal/vertical et de zoom lors d'un mouvement vers une position absolue ou une translation relative. Contrairement aux espaces de vitesse, les espaces de vitesse ne contiennent pas d'information de direction. La vitesse d'un mouvement de basculement horizontal/vertical combiné est représentée par une seule valeur scalaire non négative.

16.8.4.1 Espace de vitesse de basculement horizontal/vertical générique

L'espace de vitesse de basculement horizontal/vertical générique doit être fourni par chaque nœud PTZ qui prend en charge la vitesse configurable du basculement horizontal/vertical, dans la mesure où il n'est pas associé à un intervalle physique spécifique. Il convient plutôt que l'intervalle soit défini comme étant l'intervalle total de l'intervalle de vitesse normalisé à l'intervalle compris entre 0 (arrêt) et 1 (vitesse maximale). Cela donne lieu à la description d'espace suivante:

```
<tt:PantiltSpeedSpace>
```

```
  <tt:SpaceURI>
```

```
http://www.onvif.org/ver10/tptz/PanTiltSpaces/GenericSpeedSpace
```

```
  </tt:SpaceURI>
```

```
  <tt:XRange>
```

```
    <tt:Min>0.0</tt:Min>
```

```
    <tt:Max>1.0</tt:Max>
```

```
  </tt:XRange>
```

```
</tt:PantiltSpeedSpace>
```

16.8.4.2 Espace de vitesse de zoom générique

L'espace de vitesse de zoom générique doit être fourni par chaque nœud PTZ qui prend en charge la vitesse configurable du zoom, dans la mesure où il n'est pas associé à un intervalle physique spécifique. Il convient plutôt que l'intervalle soit défini comme étant l'intervalle total de l'intervalle de vitesse normalisé à l'intervalle compris entre 0 (arrêt) et 1 (vitesse maximale). Cela donne lieu à la description d'espace suivante:

```
<tt:ZoomSpeedSpace>
```

```
  <tt:SpaceURI>
```

```
  http://www.onvif.org/ver10/tptz/ZoomSpaces/ZoomGenericSpeedSpace
```

```
  </tt:SpaceURI>
```

```
  <tt:XRange>
```

```
    <tt:Min>0.0</tt:Min>
```

```
  </tt:XRange>
```

```
</tt:ZoomSpeedSpace>
```

16.9 Codes de défaut spécifiques au service

Le Tableau 232 présente les codes de défaut spécifiques au service PTZ. Chaque commande peut générer un défaut générique (voir Tableau 6).

Les défauts spécifiques sont définis en tant que sous-code d'un défaut générique (voir 5.11.2.1). Le sous-code générique parent est le *sous-code* en haut de chaque ligne ci-dessous, et le *sous-code* de défaut spécifique se trouve en bas de la cellule.

Tableau 232 – Codes de défaut spécifiques au service PTZ

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Receiver	ter:Action	Le préréglage ne peut pas être défini	Le préréglage ne peut pas être défini lorsque l'unité PTZ est en déplacement.
	ter:MovingPTZ		
env:Receiver	ter:Action	Nombre maximal de préréglages atteint	Le nombre maximal de préréglages est atteint.
	ter:TooManyPresets		
env:Receiver	ter:ActionNotSupported	PTZ non pris en charge	PTZ n'est pas pris en charge par le dispositif.
	ter:PTZNotSupported		
env:Sender	ter:InvalidArgVal	Le jeton existe déjà	Le nom ou jeton demandé existe déjà pour un autre préréglage.
	ter:PresetExist		
env:Receiver	ter:Action	Aucun statut PTZ disponible	Aucun statut PTZ n'est disponible dans le profil multimédia demandé.
	ter:NoStatus		
env:Receiver	ter:Action	Conflit d'utilisation des nouveaux paramètres	Les nouveaux paramètres conduisent à une configuration incohérente.
	ter:ConfigurationConflict		
env:Receiver	ter:Action	La position de départ ne peut pas être remplacée	La position de départ est fixe et ne peut pas être remplacée.
	ter:CannotOverwriteHome		
env:Sender	ter:InvalidArgVal	Nœud PTZ introuvable	Le nœud PTZ n'est pas présent sur le dispositif
	ter:NoEntity		
env:Sender	ter:InvalidArgVal	Configuration introuvable	La configuration en question n'existe pas.
	ter:NoConfig		
env:Sender	ter:InvalidArgVal	Les paramètres n'ont pas pu être définis.	Les paramètres de configuration ne peuvent pas être définis.
	ter:ConfigModify		
env:Sender	ter:InvalidArgVal	Destination hors limites	La destination demandée est hors limites.
	ter:InvalidPosition		
env:Sender	ter:InvalidArgVal	Translation hors limites	La translation demandée est hors limites.
	ter:InvalidTranslation		

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Sender	ter:InvalidArgVal	Vitesse demandée hors limites	La vitesse demandée est hors limites.
	ter:InvalidSpeed		
env:Sender	ter:InvalidArgVal	Vélocité hors limites	La vélocité demandée est hors limites.
	ter:InvalidVelocity		
env:Sender	ter:InvalidArgVal	PresetName trop long	Le PresetName (nom de préréglage) est trop long ou contient des caractères non valides.
	ter:InvalidPresetName		
env:Sender	ter:InvalidArgVal	Profil sans configuration PTZ	Le jeton de profil demandé ne référence pas une configuration PTZ.
	ter:NoPTZProfile		
env:Sender	ter:InvalidArgVal	Le jeton de profil n'existe pas.	Le ProfileToken (jeton de profil) demandé n'existe pas.
	ter:NoProfile		
env:Sender	ter:InvalidArgVal	Délai d'attente non pris en charge	L'argument de délai d'attente spécifié n'est pas dans l'intervalle de délai d'attente pris en charge.
	ter:TimeoutNotSupported		
env:Sender	ter:InvalidArgVal	Le jeton n'existe pas.	Le jeton de préréglage demandé n'existe pas.
	ter:NoToken		
env:Receiver	ter:Action	Pas de HomePosition	Aucune position de départ n'a été définie pour ce profil.
	ter:NoHomePosition		
env:Sender	ter:InvalidArgVal	Espace introuvable	Un espace est référencé dans un argument qui n'est pas pris en charge par le nœud PTZ.
	ter:SpaceNotSupported		

17 Analyse vidéo

Le paragraphe 4.12 présente une vue d'ensemble générale de l'architecture d'analyse vidéo ONVIF. Le présent paragraphe couvre les principaux aspects suivants de cette architecture:

- interface de module d'analyse;
- description de scène;
- interface de règles;

- interface d'événement.

L'interface d'événement est gérée par l'intermédiaire du service d'événement décrit à l'Article 15. Le paragraphe 17.1 présente la description de scène basée sur XML, qui peut être transmise en continu sous forme de métadonnées à des clients via RTP (voir 12.1.2.1 pour plus de détails). Le service multimédia comporte des opérations de gestion des configurations d'analyse complètes composées de la configuration de moteur de règles et de moteur d'analyse (voir Article 11). Le service d'analyse permet une configuration plus fine des règles individuelles et des modules d'analyse individuels (voir 17.2 et 0).

Un dispositif qui prend en charge l'analyse doit mettre en œuvre la description de scène et l'interface d'événement, ainsi que la configuration d'analyse par le service multimédia. Si le dispositif prend également en charge un moteur de règles, chargé du moteur d'analyse comme indiqué dans la présente Norme, il doit mettre en œuvre l'interface des modules d'analyse de règles.

Une configuration d'analyse vidéo complète peut être associée à un profil par l'intermédiaire du service multimédia. Une configuration d'analyse vidéo devient connectée à une source vidéo spécifique (voir 0). Le dispositif doit s'assurer qu'un moteur d'analyse correspondant démarre lorsqu'un client s'abonne directement ou indirectement pour des événements générés par le moteur d'analyse ou de règles ou lorsqu'un client demande le flux de description de scène correspondant.

17.1 Interface de description de scène

17.1.1 Présentation

La présente norme définit le schéma XML qui doit être utilisé pour coder des descriptions de scène par un dispositif. Le domaine d'application de la description de scène couvre des éléments de scène de base qui peuvent être affichés dans une disposition vidéo à l'utilisateur final, ainsi qu'un cadre pour des extensions spécifiques au fournisseur. L'Annexe B présente des éléments de scène supplémentaires qui peuvent être utilisés pour traiter des règles spécifiques au fournisseur.

Le moteur d'analyse vidéo est configuré grâce aux profils de la section MediaControl. Si des analyses vidéo sont disponibles dans un profil, VideoSourceConfiguration et VideoAnalyticsConfiguration doivent être référencés dans le profil. Le moteur d'analyse vidéo traite ensuite des trames en fonction du VideoSourceConfiguration référencé.

17.1.2 Contenu associé aux trames

L'entrée du moteur d'analyse vidéo est composée d'images provenant d'une source vidéo. Les éléments de scène extraits sont associés à l'image depuis laquelle ils ont été extraits. Une scène extraite se distingue de la description générale de la source vidéo traitée par le moteur d'analyse vidéo (informations telles que la ligne d'entrée vidéo, la résolution vidéo, le rognage de trame, la fréquence de trames, etc.), la liaison temporelle de trames dans le flux d'entrée, et le positionnement spatial des éléments dans une trame.

La liaison entre une source vidéo et un composant d'analyse vidéo fait partie du contrôle multimédia, qui permet aux analyses vidéo de fonctionner sur une source vidéo rognée avec une fréquence de trame réduite.

La relation temporelle et spatiale d'éléments de scène par rapport à la source vidéo sélectionnée est décrite en 17.1.2.1 et en 17.1.2.2. L'aspect et le comportement des objets suivis sont décrits en 17.1.3.1. Les interactions entre les objets (séparations et fusions, par exemple) sont décrites en 17.1.3.2.

Un dispositif PTZ peut placer des informations relatives au basculement horizontal, au basculement vertical et au zoom au début d'une trame, ce qui permet à un client d'estimer les coordonnées 3D des éléments de scène. Ensuite, le système de coordonnées d'image peut

être adapté avec un nœud de transformation facultatif, décrit dans la sous-section suivante. Enfin, plusieurs descriptions d'objet peuvent être placées, et leur liaison peut être spécifiée dans un nœud ObjectTree. Ci-dessous, les définitions sont incluses pour des raisons de commodité⁶:

```
<xs:complexType name="Frame">
  <xs:sequence>
    <xs:element name="PTZStatus" type="tt:PTZStatus"
      minOccurs="0"/>
    <xs:element name="Transformation" type="tt:Transformation"
      minOccurs="0"/>
    <xs:element name="Object" type="tt:Object" minOccurs="0"
      maxOccurs="unbounded"/>
    <xs:element name="ObjectTree" type="tt:ObjectTree" minOccurs="0"/>
    ...
  </xs:sequence>
  <xs:attribute name="UtcTime" type="xs:dateTime" use="required"/>
  ...
</xs:complexType>

<xs:element name="Frame" type="tt:Frame">
```

Le paragraphe 17.1.2.1 explique comment les trames traitées par l'algorithme d'analyse vidéo sont référencées dans le flux d'analyse vidéo.

17.1.2.1 Relation temporelle

Étant donné que plusieurs éléments de scène peuvent être extraits de la même image, les éléments de scène figurent sous un nœud de trame qui établit le lien avec une image spécifique provenant de l'entrée vidéo. Le nœud de trame contient un attribut UtcTime OBLIGATOIRE. Cet horodatage UtcTime doit permettre à un client de mettre en correspondance le nœud de trame avec exactement une trame vidéo. Par exemple, l'horodatage RTP de la trame vidéo codée correspondante doit donner lieu au même horodatage TUC après conversion. La synchronisation entre les flux vidéo et les flux de métadonnées est décrite plus avant dans la section Visualisation en temps réel en 12.1.2.2.

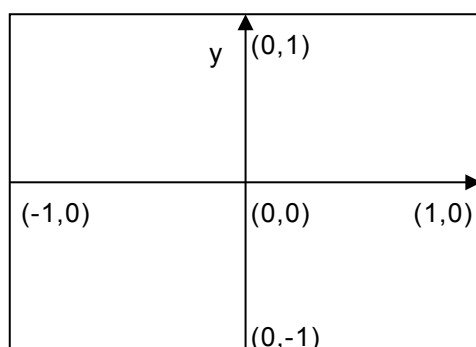
Exemple:

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  ...
</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.521">
  ...
</tt:Frame>
```

17.1.2.2 Relation spatiale

La plupart des éléments de scène font référence à une partie d'une image à partir de laquelle des informations ont été extraites. Par exemple, si des objets sont suivis dans le temps, leur position dans chaque trame doit être spécifiée. Ces positions doivent être associées à un système de coordonnées. Le système de coordonnées par défaut est représenté à la Figure 23. Il est mis en correspondance avec le rectangle sélectionné dans la VideoSourceConfiguration du profil correspondant.

⁶ À noter que le schéma est inclus ici à titre informatif uniquement. [ONVIF Schema] contient la définition de schéma normative.



IEC 2763/13

Figure 23 – Système de coordonnées de trame par défaut

La présente spécification permet la modification du système de coordonnées pour des nœuds individuels de l'arbre XML. En conséquence, chaque nœud de trame démarre avec le système de coordonnées par défaut. Chaque nœud enfant hérite du système de coordonnées le plus récent de son parent. Un nœud de transformation modifie le système de coordonnées le plus récent de son parent. Les spécifications de coordonnées sont toujours associées au système de coordonnées le plus récent du nœud parent.

La spécification définit des nœuds de transformation pour la mise à l'échelle et la translation. La description de scène contient des paramètres fictifs dans lesquels ces nœuds de transformation sont placés⁷.

```
<xs:complexType name="Transformation">
  <xs:sequence>
    <xs:element name="Translate" type="Vector" minOccurs="0"/>
    <xs:element name="Scale" type="Vector" minOccurs="0"/>
    ...
  </xs:sequence>
</xs:complexType>
```

Une description mathématique des systèmes de coordonnées et des transformations est présentée ci-après. Un système de coordonnées est composé d'un vecteur translationnel

$t = \begin{pmatrix} t_x \\ t_y \end{pmatrix}$ et d'une mise à l'échelle $s = \begin{pmatrix} s_x \\ s_y \end{pmatrix}$. Un point $p = \begin{pmatrix} p_x \\ p_y \end{pmatrix}$ défini par rapport à ce

système de coordonnées est transformé en un point correspondant $q = \begin{pmatrix} q_x \\ q_y \end{pmatrix}$ du système de

coordonnées par défaut par la formule suivante: $\begin{pmatrix} q_x \\ q_y \end{pmatrix} = \begin{pmatrix} p_x \cdot s_x + t_x \\ p_y \cdot s_y + t_y \end{pmatrix}$. De même, un vecteur

v défini par rapport au système de coordonnées est transformé en vecteur correspondant w du système de coordonnées par défaut grâce à la formule suivante: $\begin{pmatrix} w_x \\ w_y \end{pmatrix} = \begin{pmatrix} v_x \cdot s_x \\ v_y \cdot s_y \end{pmatrix}$.

⁷ À noter que le schéma est inclus ici à titre informatif uniquement. [ONVIF Schema] contient la définition de schéma normative.

Un nœud de transformation contient un vecteur de mise à l'échelle $u = \begin{pmatrix} u_x \\ u_y \end{pmatrix}$ et un vecteur

translationnel $v = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$ facultatifs. Si la mise à l'échelle n'est pas spécifiée, sa valeur par

défaut $u = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ est supposée. De même, la valeur par défaut pour la translation est $v = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

Le nœud de transformation modifie le système de coordonnées supérieur de la façon suivante:

$\begin{pmatrix} t'_x \\ t'_y \end{pmatrix} = \begin{pmatrix} v_x \times s_x + t_x \\ v_y \times s_y + t_y \end{pmatrix}$, $\begin{pmatrix} s'_x \\ s'_y \end{pmatrix} = \begin{pmatrix} u_x \times s_x \\ u_y \times s_y \end{pmatrix}$, où $\begin{pmatrix} t'_x \\ t'_y \end{pmatrix}$ et $\begin{pmatrix} s'_x \\ s'_y \end{pmatrix}$ remplacent le système de coordonnées supérieur.

Par exemple, les coordonnées de la description de scène sont présentées dans un système de coordonnées de trame, où les coordonnées du coin inférieur gauche sont (0,0) et celles du coin supérieur droit (320,240). Le nœud de trame s'apparente au code suivant, où la mise à l'échelle est définie au double de l'inverse de la largeur de trame et de la hauteur de trame:

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-1.0"/>
    <tt:Scale x="0.00625" y="0.00834"/>
  </tt:Transformation>
  ...
</tt:Frame>
```

17.1.3 Éléments de scène

Le présent paragraphe concerne les éléments de scène générés par des algorithmes de poursuite d'objet et définit la manipulation d'objet et les formes d'objet pour ceux-ci.

Les trames dans lesquelles aucun objet n'a été détecté peuvent être ignorées dans la description de scène afin d'économiser la bande passante, tant que la dernière trame dans la description de scène est également vide. Il est RECOMMANDÉ que le dispositif envoie périodiquement l'événement de description de scène même s'il est vide, afin d'indiquer que le moteur d'analyse est opérationnel. Le dispositif doit envoyer une description de scène si un point de synchronisation est demandé pour le flux correspondant.

Si le récepteur d'une description de scène reçoit une trame vide, il convient qu'il suppose que toutes les trames suivantes sont également vides tant qu'il n'a pas reçu la trame non vide suivante. Si la dernière trame reçue n'est pas vide, il convient que le récepteur suppose qu'une description de la trame traitée suivante va être transmise.

17.1.3.1 Objets

Les objets sont identifiés par leur ID d'objet. Les caractéristiques relatives à un objet particulier sont collectées dans un nœud d'objet avec l'ID d'objet correspondant en tant qu'attribut. Les associations d'objets (les objets renommés, les divisions d'objets, les fusions d'objets et les suppressions d'objet, par exemple) sont représentées dans un nœud ObjectTree séparé. Un ID d'objet est créé de manière implicite lors de la première apparition de l'ID d'objet dans un nœud d'objet⁸.

```
<xs:complexType name="ObjectId">
```

⁸ À noter que le schéma est inclus ici à titre informatif uniquement. [ONVIF Schema] contient la définition de schéma normative.

```

    <xs:attribute name="ObjectId" type="xs:int"/>
  </xs:complexType>

  <xs:complexType name="Object">
    <xs:complexContent>
      <xs:extension base="ObjectId">
        <xs:sequence>
          <xs:element name="Appearance" type="Appearance" minOccurs="0"/>
          <xs:element name="Behaviour" type="Behaviour" minOccurs="0"/>
          ...
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

```

Le nœud d'objet comporte deux paramètres fictifs pour les informations d'aspect et de comportement. Le nœud d'aspect commence par un nœud de transformation facultatif qui peut être utilisé pour passer d'un système de coordonnées orienté trame à un système de coordonnées orienté objet. Ensuite, la forme d'un objet peut être spécifiée. Si un objet est détecté dans une trame, il convient que les informations de forme soient présentes dans la description d'aspect. L'algorithme d'analyse vidéo PEUT ajouter des nœuds d'objet pour des objets non visibles actuellement, s'il est capable de déduire les informations concernant cet objet. Dans de tels cas, la description de forme PEUT être omise.

D'autres caractéristiques d'objet (la couleur et la classe d'objet, par exemple) peuvent être ajoutées au nœud Appearance. La présente norme concerne particulièrement les descripteurs de forme (voir 17.1.3.3). La définition de la couleur et de la classe d'objet est présentée en B.1.

La présente norme définit deux comportements normalisés pour les objets. Lorsqu'un objet arrête de se déplacer, il peut être marqué comme Removed (retiré) ou Idle (stationnaire). Ces comportements doivent être énumérés en tant que nœuds enfants du nœud de comportement d'un objet. La présence d'un nœud Removed ou Idle ne supprime pas automatiquement l'ID d'objet correspondant, ce qui permet de réutiliser le même ID d'objet lorsque l'objet commence à se déplacer.

Un objet marqué avec le comportement Removed spécifie l'emplacement duquel l'objet réel a été supprimé. Il convient de ne pas utiliser le marqueur en tant que comportement de l'objet retiré. La suppression d'un objet peut être détectée même si l'action d'enlèvement de l'objet n'a pas été détectée.

Les objets précédemment en mouvement peuvent être marqués comme étant Idle (stationnaire) afin d'indiquer que l'objet a arrêté de se déplacer. Tant que ces objets ne changent pas, ils ne figurent plus dans la description de scène. Lorsqu'un objet stationnaire apparaît de nouveau dans la description de scène, le drapeau Idle est automatiquement enlevé.

Exemple:

```

...
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-1.0"/>
    <tt:Scale x="0.003125" y="0.00416667"/>
  </tt:Transformation>
  <tt:Object ObjectId="12">
    <tt:Appearance>
      <tt:Shape>
        <tt:BoundingBox left="20.0" top="30.0" right="100.0"
bottom="80.0"/>
        <tt:CenterOfGravity x="60.0" y="50.0"/>
      </tt:Shape>
    </tt:Appearance>
  </tt:Object>

```

```

</tt:Frame>
...
<tt:Frame UtcTime="2008-10-10T12:24:57.421">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-1.0"/>
    <tt:Scale x="0.003125" y="0.00416667"/>
  </tt:Transformation>
  <tt:Object ObjectId="12">
    <tt:Appearance>
      <tt:Shape>
        <tt:BoundingBox left="20.0" top="30.0" right="100.0"
bottom="80.0"/>bottom="80.0"/>bottom="80.0"/>

```

17.1.3.2 Arbre d'objets

Lorsque deux objets sont trop proches l'un de l'autre, l'analyse vidéo ne pouvant alors plus les suivre individuellement, il convient qu'une fusion d'objets soit signalée en ajoutant un nœud Merge (fusion) au nœud ObjectTree (arbre d'objets) du nœud Frame (trame). Le nœud Merge contient un nœud From (de) énumérant les ID d'objet de fusion et un nœud To (vers)

contenant l'ObjectId (ID d'objet). L'objet fusionné est utilisé dans les trames ultérieures en tant qu'ID de suivi. Si l'algorithme d'analyse vidéo détecte qu'un objet masque les autres, et qu'il est capable de suivre plus avant cet objet, il convient de placer l'objet masquant dans le nœud To.

La séparation d'objets est indiquée par un nœud Split (séparation). Dans ce cas, le nœud From contient un seul ObjectId représentant l'objet qui est divisé dans la trame en cours. Les objets se séparant de cet objet divisé figurent dans le nœud To. L'ObjectId du nœud From peut réapparaître dans le nœud To, si cet objet masquait les autres et que l'algorithme d'analyse vidéo était capable de suivre cet objet pendant le masquage.

Il n'est pas nécessaire qu'un objet soit impliqué dans une opération de fusion pour faire partie d'une opération de séparation. Par exemple, si un objet se déplace conjointement avec une personne et que la personne laisse l'objet quelque part, l'objet peut être détecté la première fois par l'analyse vidéo lorsque la personne s'éloigne de l'objet. Dans ce cas, le premier aspect de l'objet peut être combiné avec une opération de séparation.

Lorsqu'un objet de fusion réapparaît sous la forme d'un nœud d'objet dans une trame ultérieure sans indication de séparation, cet objet est implicitement séparé. Toutefois, l'algorithme d'analyse vidéo ne peut pas déterminer la provenance de l'objet séparé.

Un algorithme d'analyse vidéo peut suivre et mémoriser un nombre limité d'objets. Afin d'indiquer qu'un certain objet a été retiré de la mémoire de l'algorithme et qu'il ne réapparaîtra donc jamais, SceneDescription peut contenir un nœud Delete dans le nœud ObjectTree.

Si l'algorithme d'analyse vidéo ne peut pas déterminer l'identité d'un objet lors d'une opération de séparation, il convient d'utiliser un nouvel ObjectId. Si l'algorithme a rassemblé des preuves suffisantes quant à l'identité de cet objet, il peut modifier l'ObjectId grâce à l'opération Rename (renommer). L'opération Rename peut également être utilisée lorsqu'un objet entre de nouveau dans la scène et que sa véritable identité est découverte au bout d'un certain temps.

Un ObjectId ne doit PAS être réutilisé dans la description de scène tant que son conteneur n'est pas de nouveau utilisé.

Exemple:

```
<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Object ObjectId="12">
    ...
  </tt:Object>
  <tt:Object ObjectId="17">
    ...
  </tt:Object>
</tt:Frame>

<tt:Frame UtcTime="2008-10-10T12:24:57.421">
  <tt:Object ObjectId="12">
    ...
  </tt:Object>
  <tt:ObjectTree>
    <tt:Merge>
      <tt:From ObjectId="12"/>
      <tt:From ObjectId="17"/>
      <tt:To ObjectId="12"/>
    </tt:Merge>
  </tt:ObjectTree>
</tt:Frame>

<tt:Frame UtcTime="2008-10-10T12:24:57.521">
  <tt:Object ObjectId="12">
```

```

...
</tt:Object>
</tt:Frame>

<tt:Frame UtcTime="2008-10-10T12:24:57.621">
  <tt:Object ObjectId="12">
    ...
  </tt:Object>
  <tt:Object ObjectId="17">
    ...
  </tt:Object>
  <tt:ObjectTree>
    <tt:Split>
      <tt:From ObjectId="12"/>
      <tt:To ObjectId="17"/>
      <tt:To ObjectId="12"/>
    </tt:Split>
  </tt:ObjectTree>
</tt:Frame>

```

17.1.3.3 Descripteur de forme

Des informations de forme doivent être placées au-dessous du nœud Shape (forme) facultatif dans un nœud Object Appearance (aspect d'objet). S'il est présent, le nœud Shape contient des informations sur la détection de l'objet en question dans la trame spécifiée. Un nœud Shape doit contenir au moins deux nœuds représentant le cadre délimitant et le centre de gravité de l'objet détecté.

Le cadre délimitant épais est affiné avec des nœuds enfants supplémentaires, représentant chacun une primitive de forme. En présence de plusieurs primitives de forme, leur union définit la forme de l'objet. Dans la présente norme, un descripteur de polygone générique est fourni.

Les polygones qui décrivent la forme d'un objet doivent être de simples polygones définis par une liste de points.

Deux points consécutifs (le dernier point étant relié au premier) dans la liste définissent un segment de droite. L'ordre des points doit être choisi de sorte que la zone délimitée de l'objet puisse être présente sur le côté gauche de tous les segments de droite. La ligne brisée définie par la liste de points ne doit PAS présenter d'auto-intersection.

Exemple:

```

<tt:Frame UtcTime="2008-10-10T12:24:57.321">
  <tt:Transformation>
    <tt:Translate x="-1.0" y="-1".0/>
    <tt:Scale x="0.003125" y="0.00416667"/>
  </tt:Transformation>
  <tt:Object ObjectId="12">
    <tt:Appearance>
      <tt:Shape>
        <tt:BoundingBox left="20.0" top="30.0" right="100.0"
bottom="80.0"/>
        <tt:CenterOfGravity x="60.0" y="50.0"/>
        <tt:Polygon>
          <tt:Point x="20.0" y="30.0"/>
          <tt:Point x="100.0" y="30.0"/>
          <tt:Point x="100.0" y="80.0"/>
          <tt:Point x="20.0" y="80.0"/>
        </tt:Polygon>
      </tt:Shape>
    </tt:Appearance>
  </tt:Object>
</tt:Frame>

```

17.2 Interface de règles

La configuration d'analyse vidéo est composée de deux parties (voir 0). La première partie configure le moteur d'analyse vidéo créant la SceneDescription (description de scène). La seconde partie configure le moteur de règles. Dans la seconde partie, une structure XML est introduite en 17.2.1 pour communiquer la configuration des règles. Le paragraphe 17.2.2 spécifie un langage de description de la configuration d'un type de règle spécifique. Le paragraphe 17.2.3 définit deux règles normalisées qu'il convient qu'un dispositif mettant en œuvre un moteur de règles prenne en charge. Le paragraphe 17.2.4 introduit des opérations de gestion des règles. Si le dispositif prend en charge un moteur de règles, il doit mettre en œuvre l'interface de règles complète.

17.2.1 Représentation des règles

La configuration d'une règle implique l'utilisation de deux attributs, l'un spécifiant le nom et l'autre le type de la règle. Les différents paramètres de configuration figurent sous l'élément Parameters de l'élément Rule. Chaque paramètre est un SimpleItem ou un ElementItem (comparer à la charge utile de message à l'Article 15). L'attribut Name de chaque élément doit être unique dans la liste des paramètres. SimpleItems comporte un attribut Value supplémentaire contenant la valeur du paramètre. La valeur des éléments ElementItem est donnée par l'élément enfant d'ElementItem. Il est RECOMMANDÉ de représenter autant de paramètres que possible par des éléments SimpleItem.

L'exemple suivant présente une configuration d'analyse vidéo complète contenant deux règles:

```
<tt:VideoAnalyticsConfiguration>
  <tt:AnalyticsEngineConfiguration>
    ...
  </tt:AnalyticsEngineConfiguration>
  <tt:RuleEngineConfiguration>
    <tt:Rule Name="MyLineDetector" Type="tt:LineDetector">
      <tt:Parameters>
        <tt:SimpleItem Name="Direction" Value="Any"/>
        <tt:ElementItem Name="Segments">
          <tt:Polyline>
            <tt:Point x="10.0" y="50.0"/>
            <tt:Point x="100.0" y="50.0"/>
          </tt:Polyline>
        </tt:ElementItem>
      </tt:Parameters>
    </tt:Rule>
    <tt:Rule Name="MyFieldDetector" Type="tt:FieldDetector">
      <tt:Parameters>
        <tt:ElementItem Name="Field">
          <tt:Polygon>
            <tt:Point x="10.0" y="50.0"/>
            <tt:Point x="100.0" y="50.0"/>
            <tt:Point x="100.0" y="150.0"/>
          </tt:Polygon>
        </tt:ElementItem>
      </tt:Parameters>
    </tt:Rule>
  </tt:RuleEngineConfiguration>
</tt:VideoAnalyticsConfiguration>
```

17.2.2 Langage de description de règles

La description d'une règle contient les informations de type de tous les paramètres appartenant à un certain type de règle, ainsi que la description de la sortie générée par une telle règle. La sortie du moteur de règles est composée d'événements, qui peuvent être utilisés dans un moteur d'événements ou auxquels un client peut s'abonner.

Les paramètres d'un certain type de règle figurent sous l'élément ParameterDescription. Tous les paramètres sont des éléments Simple ou ElementItem, et peuvent être décrits par une

SimpleItemDescription ou une ElementItemDescription. Les deux ItemDescription contiennent un attribut Name pour identifier le paramètre et un attribut Type pour référencer un type de schéma XML spécifique. Dans le cas de SimpleItemDescription, l'attribut Type doit référencer une définition de schéma SimpleType. Dans le cas d'ElementItemDescription, l'attribut Type doit référencer une déclaration globale d'élément d'un schéma XML.

La sortie générée par ce type de règle est décrite dans plusieurs éléments MessageDescription. Chaque élément MessageDescription contient une description de la charge utile de message conformément au langage de description de message décrit à l'Article 15. De plus, MessageDescription doit contenir un élément ParentTopic indiquant la rubrique à laquelle un client doit s'abonner pour recevoir cette sortie spécifique. La rubrique doit être spécifiée sous la forme d'une expression de rubrique concrète.

Le paragraphe 17.2.3 décrit l'utilisation du langage de description de règle de deux règles normalisées. Par souci de commodité, les définitions sont présentées ci-dessous⁹:

```
<xs:element name="RuleDescription" type="tt:ConfigDescription"/>

<xs:complexType name="ConfigDescription">
  <xs:sequence>
    <xs:element name="ParameterDescription"
      type="tt:ItemListDescription"/>
    <xs:element name="Messages" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="tt:MessageDescription">
            <xs:sequence>
              <xs:element name="ParentTopic" type="xs:string"/>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
    ...
  </xs:sequence>
  <xs:attribute name="Name" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="ItemListDescription">
  <xs:sequence>
    <xs:element name="SimpleItemDescription" minOccurs="0"
      maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="ElementItemDescription" minOccurs="0"
      maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="Name" type="xs:string" use="required"/>
        <xs:attribute name="Type" type="xs:QName" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

17.2.3 Règles normalisées

Les règles normalisées suivantes s'appliquent aux caméras statiques. Dans le cas d'un dispositif PTZ, il convient que les règles à base d'image contiennent un ElementItem

⁹ À noter que le schéma est inclus ici à titre informatif uniquement. [ONVIF Schema] contient la définition de schéma normative.

supplémentaire. L'élément `ElementItem` identifie la position du dispositif pour laquelle la règle a été définie. Un élément `ElementItemDescription` est décrit ci-dessous:

```
<tt:ElementItemDescription Name="PTZStatus" Type="tt:PTZStatusType">
```

17.2.3.1 LineDetector

L'élément `LineDetector` est défini par une ligne brisée simple sans intersection. Si un objet coupe la ligne brisée dans la direction spécifiée, le moteur de règles envoie un événement `Crossed` contenant le nom de l'élément `LineDetector` et une référence à l'objet qui a coupé la ligne. On peut sélectionner les directions `Left` (gauche), `Right` (droite) et `Any` (quelconque), les directions gauche et droite faisant référence à la direction de traversée de la ligne du premier point au deuxième point et étant les directions interdites.

L'élément `LineDetector` s'apparente au code suivant utilisant le langage de description de règle, détaillé dans la section précédente:

```
<tt:RuleDescription Name="tt:LineDetector">
  <tt:Parameters>
    <tt:SimpleItemDescription Name="Direction" Type="tt:Direction"/>
    <tt:ElementItemDescription Name="Segments" Type="tt:Polyline"/>
  </tt:Parameters>
  <tt:MessageDescription>
    <tt:Source>
      <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
    </tt:Source>
    <tt>Data>
      <tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectId"/>
    </tt>Data>
    <tt:ParentTopic>tns1:RuleEngine/LineDetector/Crossed</tt:ParentTopic>
  </tt:MessageDescription>
</tt:RuleDescription>
```

Le code ci-dessus définit deux paramètres, `Segments` et `Direction`, et produit un événement lié à la rubrique `tns1:RuleEngine/LineDetector/Crossed`.

17.2.3.2 FieldDetector

Un élément `FieldDetector` est défini par un polygone simple sans intersection. L'élément `FieldDetector` détermine si chaque objet de la scène est à l'intérieur ou à l'extérieur du polygone. Ces informations sont placées dans une propriété.

L'élément `FieldDetector` s'apparente au code suivant, utilisant le langage de description de règle détaillé dans la section précédente:

```
<tt:RuleDescription Name="tt:FieldDetector">
  <tt:Parameters>
    <tt:ElementItemDescription Name="Field" Type="tt:Polygon"/>
  </tt:Parameters>
  <tt:MessageDescription IsProperty="true">
    <tt:Source>
      <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="Rule" Type="xs:string"/>
    </tt:Source>
    <tt:Key>
      <tt:SimpleItemDescription Name="ObjectId" Type="tt:ObjectIdType"/>
    </tt:Key>
    <tt>Data>
```

```

    <tt:SimpleItemDescription Name="IsInside" Type="xs:boolean"/>
  </tt:Data >
  <tt:ParentTopic>
    tns1:RuleEngine/FieldDetector/ObjectsInside
  </tt:ParentTopic>
</tt:MessageDescription>
</tt:RuleDescription>

```

À partir de la propriété Inside, un client peut déduire les paramètres Entering et Leaving du détecteur. Un client peut simuler des événements Entering et Leaving en ajoutant un filtre MessageContent à l'abonnement, qui laisse passer uniquement les messages ObjectsInside, la valeur true (vrai) ou false (faux) étant respectivement affectée à l'attribut IsInside.

17.2.4 Opérations sur les règles

Si le dispositif prend en charge un moteur de règles comme défini par l'ONVIF, il doit mettre en œuvre les opérations suivantes de gestion des règles. Les opérations Create/Delete/Modify (créer/supprimer/modifier) sont atomiques, ce qui signifie que toutes les modifications peuvent être traitées ou que l'opération complète doit échouer.

17.2.4.1 Obtention des règles prises en charge

Le dispositif doit indiquer les règles qu'il prend en charge en mettant en œuvre l'opération suivante (voir Tableau 233). Il retourne une liste de descriptions de règle conformément au langage de description de règle décrit en 17.2.2. De plus, il contient une liste d'URL qui indiquent l'emplacement des fichiers de schéma. Ces fichiers de schéma décrivent les types et éléments utilisés dans les descriptions de règle. Si des descriptions de règle référencent des types ou des éléments du fichier de schéma ONVIF, ce dernier doit être explicitement mentionné.

Tableau 233 – Commande GetSupportedRules

GetSupportedRules	Demande-réponse
Nom du message	Description
GetSupportedRulesRequest	Le message de demande contient le VideoAnalyticsConfigurationToken pour lequel il convient de répertorier les règles prises en charge. tt:ReferenceToken ConfigurationToken [1][1]
GetSupportedRulesResponse	La réponse contient les règles prises en charge. tt: SupportedRules SupportedRules [1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoConfig	VideoAnalyticsConfiguration n'existe pas.

17.2.4.2 Obtention de règles

L'opération suivante extrait les règles actuellement installées (voir Tableau 234).

Tableau 234 – Commande GetRules

GetRules		Demande-réponse
Nom du message	Description	
GetRulesRequest	<i>Le message de demande spécifie le VideoAnalyticsConfigurationToken pour lequel il convient que les règles soient rapportées.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetRulesResponse	<i>La réponse est une liste de règles installées pour la configuration spécifiée.</i> tt:Config Rule [0][non limité]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.</i>	

17.2.4.3 Création de règles

L'opération suivante ajoute des règles à VideoAnalyticsConfiguration (voir Tableau 235). Si toutes les règles ne peuvent pas être créées comme requis, le dispositif répond par un message de défaut.

Tableau 235 – Commande CreateRules

CreateRules		Demande-réponse
Nom du message	Description	
CreateRulesRequest	<i>Le message de demande spécifie le VideoAnalyticsConfigurationToken auquel il convient que d'ajouter les règles énumérées.</i> tt:ReferenceToken ConfigurationToken [1][1] tt:Config Rule [1][non limité]	
CreateRulesResponse	Ceci est un message vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.</i>	
env:Sender ter:InvalidArgVal ter:InvalidRule	<i>La configuration de règles suggérée n'est pas valide sur le dispositif.</i>	
env:Sender ter:InvalidArgVal ter:RuleAlreadyExistent	<i>Le même nom de règle existe déjà dans la configuration.</i>	
enc:Receiver ter:Action ter:TooManyRules	<i>Espace insuffisant dans le dispositif pour ajouter les règles à la configuration.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Le dispositif ne peut pas créer les règles sans créer une configuration conflictuelle.</i>	

17.2.4.4 Modification de règles

L'opération suivante modifie plusieurs règles (voir Tableau 236). Si toutes les règles ne peuvent pas être modifiées comme requis, le dispositif répond par un message de défaut.

Tableau 236 – Commande ModifyRules

ModifyRules		Demande-réponse
Nom du message	Description	
ModifyRulesRequest	<i>Le message de demande spécifie le VideoAnalyticsConfigurationToken pour lequel il convient de modifier les règles énumérées.</i> tt:ReferenceToken ConfigurationToken [1][1] tt:Config Rule[1][non limité]	
ModifyRulesResponse	Ceci est un message vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.	
env:Sender ter:InvalidArgVal ter:InvalidRule	La configuration de règles suggérée n'est pas valide sur le dispositif.	
env:Sender ter:InvalidArgs ter:RuleNotExistent	Le nom ou des noms de règle n'existe(nt) pas.	
enc:Receiver ter:Action ter:TooManyRules	Espace insuffisant dans le dispositif pour ajouter les règles à la configuration.	
env:Receiver ter:Action ter:ConflictingConfig	Le dispositif ne peut pas modifier les règles sans créer une configuration conflictuelle.	

17.2.4.5 Suppression de règles

L'opération suivante supprime plusieurs règles (voir Tableau 237). Si toutes les règles ne peuvent pas être supprimées comme requis, le dispositif répond par un message de défaut.

Tableau 237 – Commande DeleteRules

DeleteRules		Demande-réponse
Nom du message	Description	
DeleteRulesRequest	<i>Le message de demande spécifie le VideoAnalyticsConfigurationToken duquel il convient de supprimer les règles énumérées.</i> tt:ReferenceToken ConfigurationToken [1][1] xs:string RuleName [1][non limité]	
DeleteRulesResponse	<i>La réponse est un message vide.</i>	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.</i>	
env:Receiver ter:Action ter:ConflictingConfig	<i>Le dispositif ne peut pas supprimer les règles sans créer une configuration conflictuelle.</i>	
env:Sender ter:InvalidArgs ter:RuleNotExistent	<i>Le nom ou des noms de règle n'existe(nt) pas.</i>	

17.3 Interface de module d'analyse

La configuration d'analyse vidéo est composée de deux parties (voir 0). La première partie configure le moteur d'analyse vidéo créant la SceneDescription (description de scène). La seconde partie configure le moteur de règles. Le paragraphe 17.3.1 définit une structure XML pour la première partie qui communique la configuration de modules d'analyse. Le paragraphe 17.3.2 définit le langage de description de la configuration d'un module d'analyse spécifique. Le paragraphe 17.3.3 définit les opérations requises par l'interface de module d'analyse. Si le dispositif prend en charge un moteur d'analyse comme défini par l'ONVIF, il doit mettre en œuvre l'intégralité de l'interface de module d'analyse.

17.3.1 Configuration de module d'analyse

La configuration de module d'analyse est identique à la configuration de règle (voir 17.2.1). L'exemple suivant présente une possible configuration d'un élément ObjectTracker spécifique au fournisseur. Ce dispositif de poursuite permet de configurer les tailles d'objet minimale et maximale pour la géométrie de trame traitée.

```
<tt:VideoAnalyticsConfig>
  <tt:AnalyticsEngineConfig>
    <tt:AnalyticsModule Name="MyObjectTracker" Type="nn:ObjectTracker">
      <tt:Parameters>
        <tt:SimpleItem Name="MinObjectWidth" Value="0.01"/>
        <tt:SimpleItem Name="MinObjectHeight" Value="0.01"/>
        <tt:SimpleItem Name="MaxObjectWidth" Value="0.5"/>
        <tt:SimpleItem Name="MaxObjectHeight" Value="0.5"/>
      </tt:Parameters>
    </tt:AnalyticsModule>
  </tt:AnalyticsEngineConfig>
</tt:VideoAnalyticsConfig>
```

```

</tt:AnalyticsModule>
</tt:AnalyticsEngineConfig>
<tt:RuleEngineConfig>
...
</tt:RuleEngineConfig>
</tt:VideoAnalyticsConfig>

```

17.3.2 Langage de description de module d'analyse

Le module d'analyse réutilise le langage de description de règle (voir 17.2.2). L'élément `AnalyticsModuleDescription` remplace l'élément `RuleDescription`:

```

<xs:element name="AnalyticsModuleDescription"

type="tt:ConfigDescription"/>

```

De même, les modules d'analyse génèrent des événements et doivent figurer dans la description de module d'analyse. La description suivante correspond à l'exemple de la section précédente. L'exemple de module génère un événement `SceneTooCrowded` lorsque la scène devient trop complexe pour le module.

```

<tt:AnalyticsModuleDescription Name="nn:ObjectTracker">
  <tt:Parameters>
    <tt:SimpleItemDescription Name="MinObjectWidth" Type="xs:float"/>
    <tt:SimpleItemDescription Name="MinObjectHeight" Type="xs:float"/>
    <tt:SimpleItemDescription Name="MaxObjectWidth" Type="xs:float"/>
    <tt:SimpleItemDescription Name="MaxObjectHeight" Type="xs:float"/>
  </tt:Parameters>
  <tt:MessageDescription>
    <tt:Source>
      <tt:SimpleItemDescription Name="VideoSourceConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="VideoAnalyticsConfigurationToken"
        Type="tt:ReferenceToken"/>
      <tt:SimpleItemDescription Name="AnalyticsModule" Type="xs:string"/>
    </tt:Source>
    <tt:ParentTopic>
      tns1:VideoAnalytics/nn:ObjectTracker/SceneTooCrowded
    </tt:ParentTopic>
  </tt:MessageDescription>
</tt:AnalyticsModuleDescription>

```

17.3.3 Opérations sur les modules d'analyse

Si le dispositif prend en charge un moteur d'analyse comme défini par l'ONVIF, il doit prendre en charge les opérations suivantes pour gérer les modules d'analyse. Les opérations `Create/Delete/Modify` (créer/supprimer/modifier) doivent être atomiques, toutes les modifications pouvant être traitées ou l'opération complète devant échouer.

17.3.3.1 GetSupportedAnalyticsModules

Le dispositif indique les modules d'analyse qu'il prend en charge en mettant en œuvre l'opération `GetSupportedAnalyticsModule` (voir Tableau 238). Il retourne une liste de modules d'analyse conformément au langage de description de module d'analyse (voir 17.2.2). De plus, il contient une liste d'URL qui indiquent l'emplacement des fichiers de schéma. Ces fichiers de schéma décrivent les types et éléments utilisés dans les descriptions de module d'analyse. Si les descriptions de module d'analyse référencent des types ou éléments du fichier de schéma ONVIF, ce dernier doit être explicitement mentionné.

Tableau 238 – Commande GetSupportedAnalyticsModules

GetSupportedAnalyticsModules		Demande-réponse
Nom du message	Description	
GetSupportedAnalyticsModules Request	<i>Le message de demande contient le VideoAnalyticsConfigurationToken pour lequel il convient de répertorier les modules d'analyse pris en charge.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetSupportedAnalyticsModules Response	<i>La réponse contient les modules d'analyse pris en charge.</i> SupportedAnalyticsModules [1][1]	
Codes de défaut		Description
env:Sender ter:InvalidArgs ter:NoConfig		<i>VideoAnalyticsConfiguration n'existe pas.</i>

17.3.3.2 Obtention de modules d'analyse

L'opération suivante extrait les modules d'analyse actuellement installés (voir Tableau 239).

Tableau 239 – Commande GetAnalyticsModules

GetAnalyticsModules		Demande-réponse
Nom du message	Description	
GetAnalyticsModulesRequest	<i>Le message de demande spécifie le VideoAnalyticsConfigurationToken (jeton de configuration d'analyse vidéo) pour lequel il convient que les modules d'analyse soient rapportés.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsModulesResponse	<i>La réponse est une liste des modules d'analyse installés pour la configuration spécifiée.</i> tt:Config AnalyticsModule [0][non limité]	
Codes de défaut		Description
env:Sender ter:InvalidArgs ter:NoConfig		<i>La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.</i>

17.3.3.3 Création de modules d'analyse

L'opération suivante permet d'ajouter des modules d'analyse à une VideoAnalyticsConfiguration (voir Tableau 240). Si les modules ne peuvent pas tous être créés conformément à la demande, le dispositif répond par un message de défaut.

Tableau 240 – Commande CreateAnalyticsModules

CreateAnalyticsModules		Demande-réponse
Nom du message	Description	
CreateAnalyticsModulesRequest	Le message de demande spécifie le VideoAnalyticsConfigurationToken pour lequel il convient que les modules d'analyse soient ajoutés. tt:ReferenceToken ConfigurationToken [1][1] tt:Config AnalyticsModule [1][non limité]	
CreateAnalyticsModulesResponse	Ceci est un message vide.	
Codes de défaut		Description
env:Sender ter:InvalidArgs ter:NoConfig		La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.
env:Sender ter:InvalidArgs ter:NameAlreadyExistent		Le même nom de module d'analyse existe déjà dans la configuration.
enc:Receiver ter:Action ter:TooManyModules		Espace insuffisant dans le dispositif pour ajouter les modules d'analyse à la configuration.
env:Receiver ter:Action ter:ConfigurationConflict		Le dispositif ne peut pas créer les modules d'analyse sans générer une configuration conflictuelle.
env:Sender ter:InvalidArgVal ter:InvalidModule		La configuration de module suggérée n'est pas valide sur le dispositif.

17.3.3.4 Modification de modules d'analyse

L'opération suivante permet de modifier plusieurs modules d'analyse (voir Tableau 241). Si tous les modules d'analyse ne peuvent pas être modifiés conformément à la demande, le dispositif répond par un message de défaut.

Tableau 241 – Commande ModifyAnalyticsModules

ModifyAnalyticsModules		Demande-réponse
Nom du message	Description	
ModifyAnalyticsModulesRequest	<i>Le message de demande spécifie le VideoAnalyticsConfigurationToken (jeton de configuration d'analyse vidéo) pour lequel il convient que les modules d'analyse énumérés soient modifiés.</i> tt:ReferenceToken ConfigurationToken [1][1] tt:Config AnalyticsModule [1][non limité]	
ModifyAnalyticsModulesResponse	La réponse est un message vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:NoConfig	<i>La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.</i>	
env:Sender ter:InvalidArgs ter:NameNotExistent	<i>Le module d'analyse avec le nom demandé n'existe pas.</i>	
enc:Receiver ter:Action ter:TooManyModules	<i>Espace insuffisant dans le dispositif pour ajouter les modules d'analyse à la configuration.</i>	
env:Receiver ter:Action ter:ConfigurationConflict	<i>Le dispositif ne peut pas modifier les modules d'analyse sans générer une configuration conflictuelle.</i>	
env:Sender ter:InvalidArgVal ter:InvalidModule	<i>La configuration de module suggérée n'est pas valide sur le dispositif.</i>	

17.3.3.5 Suppression de modules d'analyse

L'opération suivante permet de supprimer plusieurs modules d'analyse (voir Tableau 242). Si tous les modules d'analyse ne peuvent pas être supprimés conformément à la demande, le dispositif répond par un message de défaut.

Tableau 242 – Commande DeleteAnalyticsModules

DeleteAnalyticsModules		Demande-réponse
Nom du message	Description	
DeleteAnalyticsModulesRequest	<i>Le message de demande spécifie le VideoAnalyticsConfigurationToken duquel il convient de supprimer les modules d'analyse énumérés.</i> tt:ReferenceToken ConfigurationToken [1][1] xs:string AnalyticsModuleName [1][non limité]	
DeleteAnalyticsModulesResponse	La réponse est un message vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgs ter:NoConfig	La VideoAnalyticsConfiguration (configuration d'analyse vidéo) n'existe pas.	
env:Receiver ter:Action ter:ConfigurationConflict	Le dispositif ne peut pas supprimer les modules d'analyse sans générer une configuration conflictuelle.	
env:Sender ter:InvalidArgs ter:NameNotExistent	Le module d'analyse avec le nom demandé n'existe pas.	

17.4 Codes de défaut spécifiques au service

Le Tableau 243 présente les codes de défaut spécifiques au service d'analyse. Une commande peut générer un défaut générique. Voir Tableau 6.

Les défauts spécifiques sont définis en tant que sous-code d'un défaut générique (voir 5.11.2.1). Le sous-code générique parent est le *sous-code* en haut de chaque ligne ci-dessous, le *sous-code* de défaut spécifique se trouvant en bas de la cellule.

Tableau 243 – Codes de défaut spécifiques au service d'analyse

Code de défaut	Sous-code parent	Raison de défaut	Description
	Sous-code		
env:Receiver	ter:Action	Plus d'espace disponible	Espace insuffisant dans le dispositif pour ajouter les règles à la configuration.
	ter:TooManyRules		
env:Receiver	ter:Action	Plus d'espace disponible.	Espace insuffisant dans le dispositif pour ajouter les modules d'analyse à la configuration.
	ter:TooManyModules		
env:Receiver	ter:Action	Conflit d'utilisation des nouveaux paramètres	Les nouveaux paramètres conduisent à une configuration incohérente.
	ter:ConfigurationConflict		
env:Sender	ter:InvalidArgVal	Configuration introuvable	La VideoAnalyticsConfiguration demandée n'existe pas.
	ter:NoConfig		
env:Sender	ter:InvalidArgVal	La règle est non valide	La configuration de règle suggérée n'est pas valide.
	ter:InvalidRule		
env:Sender	ter:InvalidArgVal	Le module est non valide	La configuration de module d'analyse suggérée n'est pas valide sur le dispositif.
	ter:InvalidModule		
env:Sender	ter:InvalidArgVal	La règle existe déjà	Le même nom de règle existe déjà dans la configuration.
	ter:RuleAlreadyExistent		
env:Sender	ter:InvalidArgs	La règle n'existe pas	Le nom ou des noms de règle n'existe(nt) pas.
	ter:RuleNotExistent		
env:Sender	ter:InvalidArgs	Le nom existe déjà	Le même nom de module d'analyse existe déjà dans la configuration.
	ter:NameAlreadyExistent		
env:Sender	ter:InvalidArgs	Le nom n'existe pas	Le module d'analyse avec le nom demandé n'existe pas.
	ter:NameNotExistent		

18 Dispositif d'analyse

Il faut que le service de dispositif d'analyse soit utilisé pour les dispositifs d'analyse autonomes qui exécutent des processus d'évaluation sur les flux multimédia ou les flux multimédia améliorés par métadonnées. Il peut également être utilisé pour d'autres entités. Les évaluations peuvent concerner plusieurs flux multimédia ou flux multimédia améliorés par métadonnées à la fois.

Le service de dispositif d'analyse reçoit les flux multimédia ou flux multimédia améliorés par métadonnées de la part de dispositifs de génération active ou de stockage. Il peut être doté de fonctionnalités de décodeur si l'analyse est réalisée sur des données non compressées.

Un flux multimédia amélioré par métadonnées décrit un flux contenant des métadonnées et des métadonnées assignées.

Le service de dispositif d'analyse est utilisé par les clients pour configurer les propriétés et fonctionnalités d'un dispositif d'analyse autonome ou réaliser d'autres opérations d'analyse sur une entité assurant ce service.

Les fonctionnalités de voie de retour ne sont pas fournies par les dispositifs d'analyse autonomes.

Le service de dispositif d'analyse s'appuie sur le service de récepteur des données provenant d'autres dispositifs grâce aux objets de réception identifiés par ReceiverTokens. Il faut prévoir des mécanismes pour attribuer différents suivis dans le flux RTSP reçus à l'élément AnalyticsEngine approprié.

Les modifications apportées aux paramètres de caméra, par exemple, lors de l'analyse peuvent influencer les résultats de l'analyse. Par conséquent, il faut répercuter les modifications des paramètres d'entrée dans la structure AnalyticsEngineInput.

18.1 Présentation

L'élément central de la configuration d'un service de dispositif d'analyse est l'AnalyticsEngineControl. Il est composé des jetons et descriptions nécessaires au service, et offre la possibilité d'activer/de désactiver l'AnalyticsEngineControl particulier.

Un AnalyticsEngine peut être un algorithme unique ou une application complète (des bagages perdus, par exemple). Plusieurs jeux de paramètres (VideoAnalyticsConfiguration) peuvent exister en parallèle pour permettre à un AnalyticsEngine de basculer d'une configuration diurne à une configuration nocturne, par exemple. De plus, une structure est fournie (AnalyticsEngineInputInfo) pour décrire les exigences de configuration d'entrée de l'AnalyticsEngine particulier.

Pour pouvoir adapter l'AnalyticsEngine à différentes données d'entrée, il faut fournir la description de l'entrée proposée à l'AnalyticsEngine dans l'élément AnalyticsEngineInput.

Il faut que toutes les structures existent au moins une fois après le démarrage de l'entité NVA, et des valeurs par défaut peuvent être saisies, le cas échéant.

18.2 Entrée de moteur d'analyse

La structure AnalyticsEngineInput décrit l'entrée vidéo et de métadonnées fournie à un AnalyticsEngine particulier. Si plusieurs sources d'entrée sont utilisées, il faut qu'il y ait un élément AnalyticsEngineInput par source.

Sourceldentification: identifie la source de l'entrée (identification du groupe de caméras, la caméra particulière et le profil utilisés, par exemple)

VideoSource: informations relatives à la source vidéo, en particulier les paramètres de compression utilisés

MetadataInput: décrit l'approvisionnement de métadonnées source à utiliser pour l'analyse

18.2.1 GetAnalyticsEngineInputs

Cette opération permet de répertorier toutes les entrées de moteur d'analyse disponibles pour le dispositif (voir Tableau 244). Le service de dispositif d'analyse doit prendre en charge la liste des entrées de moteur d'analyse disponibles grâce à la commande GetAnalyticsEngineInputs.

Tableau 244 – Commande GetAnalyticsEngineInputs

GetAnalyticsEngineInputs		Demande-Réponse
Nom du message	Description	
GetAnalyticsEngineInputsRequest	<i>Ceci est un message vide.</i>	
GetAnalyticsEngineInputsResponse	<i>Contient une liste de structures décrivant les éléments AnalyticsEngineInput disponibles.</i> tt:AnalyticsEngineInput Configuration [1][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

18.2.2 GetAnalyticsEngineInput

La commande GetAnalyticsEngineInput extrait la configuration d'entrée si le jeton de configuration d'entrée de moteur d'analyse est connu. Un service de dispositif d'analyse doit prendre en charge la liste d'une configuration d'entrée de moteur d'analyse grâce à la commande GetAnalyticsEngineInput (voir Tableau 245).

Tableau 245 – Commande GetAnalyticsEngineInput

GetAnalyticsEngineInput		Demande-Réponse
Nom du message	Description	
GetAnalyticsEngineInputRequest	<i>Contient le jeton d'une configuration d'entrée de moteur d'analyse existante.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsEngineInputResponse	<i>Contient la configuration d'entrée de moteur d'analyse demandée.</i> tt:AnalyticsEngineInput Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	

18.2.3 SetAnalyticsEngineInput

Cette commande modifie la configuration d'entrée de moteur d'analyse. Un service de dispositif d'analyse doit prendre en charge la modification de sa configuration d'entrée de moteur d'analyse grâce à cette commande (voir Tableau 246).

Tableau 246 – Commande SetAnalyticsEngineInput

SetAnalyticsEngineInput		Demande-Réponse
Nom du message	Description	
SetAnalyticsEngineInput - Request	<i>La configuration doit être une nouvelle configuration.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AnalyticsEngineInput Configuration[1][1] xs:boolean ForcePersistence [1][1]	
SetAnalyticsEngineInputResponse	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	La configuration ne peut pas être définie	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée indiquée avec ConfigurationToken n'existe pas.	

18.2.4 CreateAnalyticsEngineInputs

Cette commande génère des configurations d'entrée de moteur d'analyse. Un service de dispositif d'analyse doit prendre en charge la génération des configurations d'entrée de moteur d'analyse grâce à cette commande (voir Tableau 247).

Tableau 247 – Commande CreateAnalyticsEngineInputs

CreateAnalyticsEngineInputs		Demande-Réponse
Nom du message	Description	
CreateAnalyticsEngineInputsRequest	<i>La configuration doit être une nouvelle configuration.</i> <i>L'élément ForcePersistence détermine si la configuration doit être stockée et conservée après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AnalyticsEngineInput Configuration[1][non limité] xs:boolean ForcePersistence [1][non limité]	
CreateAnalyticsEngineInputsResponse	<i>Contient les configurations incluant les jetons générés.</i> tt:AnalyticsEngineInput Configuration[1][non limité]	
Codes de défaut		Description
env:Sender		<i>Les configurations ne peuvent pas être définies</i>
ter:InvalidArgVal		
ter:invalidConfig		
env:Receiver		<i>Le nombre maximal d'objets AnalyticsEngineInput pris en charge a été atteint.</i>
ter:Action		
ter:MaxAnalyticsEngineInput		

18.2.5 DeleteAnalyticsEngineInputs

Cette commande supprime des configurations d'entrée de moteur d'analyse. Un service de dispositif d'analyse doit prendre en charge la suppression des configurations d'entrée de moteur d'analyse grâce à cette commande (voir Tableau 248).

Tableau 248 – Commande DeleteAnalyticsEngineInputs

DeleteAnalyticsEngineInputs		Demande-Réponse
Nom du message	Description	
DeleteAnalyticsEngineInputsRequest	<i>Contient des ConfigurationTokens identifiant les éléments AnalyticsEngineInput à supprimer.</i> tt:ReferenceToken ConfigurationToken [1][non limité]	
DeleteAnalyticsEngineInputsResponse	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineInput	L'élément AnalyticsEngineInput demandé indiqué avec ConfigurationToken n'existe pas.	
env:Sender ter:Action ter:CannotDeleteEngineInput	Un élément AnalyticsEngineInput spécifié ne peut pas être supprimé.	

18.3 Configuration d'analyse vidéo

18.3.1 GetVideoAnalyticsConfiguration

La commande GetVideoAnalyticsConfiguration extrait la configuration d'analyse vidéo si le jeton de configuration d'analyse vidéo est connu. Un service de dispositif d'analyse doit prendre en charge la liste de la configuration d'analyse vidéo grâce à la commande GetVideoAnalyticsConfiguration (voir Tableau 249).

Un jeton de configuration d'analyse vidéo pertinent peut être trouvé dans les configurations AnalyticsEngine disponibles.

Tableau 249 – Commande GetVideoAnalyticsConfiguration

GetVideoAnalyticsConfiguration		Demande-Réponse
Nom du message	Description	
GetVideoAnalyticsConfigurationRequest	<i>Contient le jeton d'une configuration d'analyse vidéo existante.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetVideoAnalyticsConfigurationResponse	<i>Contient la configuration d'analyse vidéo demandée.</i> tt:VideoAnalyticsConfiguration Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée indiquée avec ConfigurationToken n'existe pas.	

18.3.2 SetVideoAnalyticsConfiguration

Cette commande modifie la configuration d'analyse vidéo. Un service de dispositif d'analyse doit prendre en charge la modification de sa configuration de moteur d'analyse grâce à cette commande (voir Tableau 250). Si le service de dispositif d'analyse reçoit la commande SetVideoAnalyticsConfiguration, les modifications doivent être également appliquées à la configuration concernée, si elle est en cours d'utilisation.

Tableau 250 – Commande SetVideoAnalyticsConfiguration

SetVideoAnalyticsConfiguration		Demande-Réponse
Nom du message	Description	
SetVideoAnalyticsConfiguration Request	–	La configuration doit être une nouvelle configuration. L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage. tt:VideoAnalyticsConfiguration.Configuration[1][1] xs:boolean ForcePersistence [1][1]
SetVideoAnalyticsConfigurationResponse	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	La configuration ne peut pas être définie	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée indiquée avec ConfigurationToken n'existe pas.	

18.4 Moteurs d'analyse

La structure retournée par les commandes définies ici contient une liste des éléments VideoAnalyticsConfiguration disponibles pour l'élément AnalyticsEngine particulier, avec les éléments AnalyticsEngineInputInfo appropriés pour chaque VideoAnalyticsConfiguration.

VideoAnalyticsConfiguration: description des possibilités de configuration du moteur d'analyse

AnalyticsEngineInputInfo: informations relatives aux exigences d'entrée du moteur d'analyse

18.4.1 GetAnalyticsEngines

Cette opération permet de répertorier tous les moteurs d'analyse disponibles du dispositif. Le service de dispositif d'analyse doit prendre en charge la liste des moteurs d'analyse disponibles grâce à la commande GetAnalyticsEngines (voir Tableau 251).

Tableau 251 – Commande GetAnalyticsEngines

GetAnalyticsEngines		Demande-Réponse
Nom du message	Description	
GetAnalyticsEnginesRequest	<i>Ceci est un message vide.</i>	
GetAnalyticsEnginesResponse	<i>Contient une liste de structures décrivant les éléments AnalyticsEngine disponibles.</i> tt:AnalyticsEngine Configuration [1][non limité]	
Codes de défaut	Description	
	<i>Pas de défauts spécifiques à la commande!</i>	

18.4.2 GetAnalyticsEngine

La commande GetAnalyticsEngine extrait le moteur d'analyse si le jeton de moteur d'analyse est connu. Un service de dispositif d'analyse doit prendre en charge la liste d'une configuration de moteur d'analyse grâce à la commande GetAnalyticsEngine (voir Tableau 252).

Tableau 252 – Commande GetAnalyticsEngine

GetAnalyticsEngine		Demande-Réponse
Nom du message	Description	
GetAnalyticsEngineRequest	<i>Contient le jeton d'un moteur d'analyse existant.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsEngineResponse	<i>Contient la configuration de moteur d'analyse demandée.</i> tt:AnalyticsEngine Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée indiquée avec ConfigurationToken n'existe pas.	

18.5 Contrôle de moteur d'analyse

La structure AnalyticsEngineControl doit être utilisée pour exercer le contrôle grâce aux commandes définies ci-après.

Name: description conviviale

EngineToken: jeton du moteur d'analyse (AnalyticsEngine) à contrôler

EngineConfigToken: jeton de la configuration de moteur d'analyse (VideoAnalyticsConfiguration) en vigueur

InputToken: jetons de la configuration d'entrée (AnalyticsEngineInput) appliquée

ReceiverToken: jetons du récepteur fournissant les données d'entrée multimédia. L'ordre des éléments ReceiverToken doit correspondre exactement à celui des éléments InputToken.

Multicast: paramètre de multidiffusion utilisé pour configurer et commander la multidiffusion du flux de métadonnées

Subscription: description des rubriques auxquelles réagit le moteur contrôlé

Mode: indique le statut en cours de l'analyse contrôlée (doit être "Idle" ou "Active")

18.5.1 GetAnalyticsEngineControls

Cette opération permet de répertorier tous les contrôles de moteur d'analyse disponibles pour le dispositif. Le service de dispositif d'analyse doit prendre en charge la liste des contrôles de moteur d'analyse disponibles grâce à la commande GetAnalyticsEngineControls (voir Tableau 253).

Tableau 253 – Commande GetAnalyticsEngineControls

GetAnalyticsEngineControls		Demande-Réponse
Nom du message	Description	
GetAnalyticsEngineControlsRequest	Ceci est un message vide.	
GetAnalyticsEngineControlsResponse	Contient une liste de structures décrivant les éléments AnalyticsEngineControl disponibles.	
	tt:AnalyticsEngineControl [1][non limité]	AnalyticsEngineControls
Codes de défaut	Description	
	Pas de défauts spécifiques à la commande!	

18.5.2 GetAnalyticsEngineControl

La commande GetAnalyticsEngineControl extrait le contrôle de moteur d'analyse si le jeton de contrôle de moteur d'analyse est connu. Un service de dispositif d'analyse doit prendre en charge la liste de la configuration de contrôle de moteur d'analyse grâce à la commande GetAnalyticsEngineControl (voir Tableau 254).

Tableau 254 – Commande GetAnalyticsEngineControl

GetAnalyticsEngineControl		Demande-Réponse
Nom du message	Description	
GetAnalyticsEngineControlRequest	<i>Contient le jeton d'un élément AnalyticsEngineControl existant.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsEngineControlResponse	<i>Contient la configuration AnalyticsEngineControl demandée.</i> tt:AnalyticsEngineControl Configuration [1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoConfig	<i>La configuration demandée indiquée avec ConfigurationToken n'existe pas.</i>	

18.5.3 SetAnalyticsEngineControl

Cette commande modifie la configuration AnalyticsEngineControl. Un service de dispositif d'analyse doit prendre en charge la modification de sa configuration de contrôle de moteur d'analyse grâce à cette commande (voir Tableau 255).

Tableau 255 – Commande SetAnalyticsEngineControl

SetAnalyticsEngineControl		Demande-Réponse
Nom du message	Description	
SetAnalyticsEngineControlRequest	<i>La configuration doit être une nouvelle configuration.</i> <i>L'élément ForcePersistence détermine si les modifications de configuration doivent être stockées et conservées après le redémarrage. Si la valeur "true" est attribuée, les modifications doivent être persistantes. Si la valeur "false" est attribuée, les modifications PEUVENT revenir aux valeurs précédentes après le redémarrage.</i> tt:AnalyticsEngineControl Configuration[1][1] xs:boolean ForcePersistence [1][1]	
SetAnalyticsEngineControlResponse	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:invalidConfig	La configuration ne peut pas être définie	
env:Sender ter:InvalidArgVal ter:NoConfig	La configuration demandée indiquée avec ConfigurationToken n'existe pas.	

18.5.4 CreateAnalyticsEngineControl

CreateAnalyticsEngineControl doit créer un objet de contrôle. Son mode doit être "idle". Pour passer en mode "active", la commande SetAnalyticsEngineControl peut être utilisée. Un service de dispositif d'analyse doit prendre en charge la création d'objets de contrôle grâce à cette commande (voir Tableau 256).

Tableau 256 – Commande CreateAnalyticsEngineControl

CreateAnalyticsEngineControl		Demande-Réponse
Nom du message	Description	
CreateAnalyticsEngineControlRequest	<i>La configuration doit être une nouvelle configuration.</i> tt:AnalyticsEngineControl Configuration[1][1]	
CreateAnalyticsEngineControlResponse	<i>Contient la configuration incluant le jeton généré.</i> tt:AnalyticsEngineControl Configuration[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:AnalyticsEngineControlExists	<i>Un élément AnalyticsEngineControl avec le jeton ConfigurationToken existe déjà.</i>	
env:Receiver ter:Action ter:MaxAnalyticsEngineControl	<i>Le nombre maximal d'objets AnalyticsEngineControl pris en charge a été atteint.</i>	
env:Sender ter:InvalidArgVal ter:invalidConfig	<i>La configuration ne peut pas être définie</i>	

18.5.5 DeleteAnalyticsEngineControl

DeleteAnalyticsEngineControl doit supprimer un objet de contrôle. Un service de dispositif d'analyse doit prendre en charge la suppression d'objets de contrôle grâce à cette commande (voir Tableau 257).

Tableau 257 – Commande DeleteAnalyticsEngineControl

DeleteAnalyticsEngineControl		Demande-Réponse
Nom du message	Description	
DeleteAnalyticsEngineControlRequest	<i>Contient le ConfigurationToken de l'élément AnalyticsEngineControl à supprimer</i> tt:ReferenceToken ConfigurationToken [1][1]	
DeleteAnalyticsEngineControlResponse	Ce message est vide.	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	L'élément AnalyticsEngineControl demandé indiqué avec ConfigurationToken n'existe pas.	
env:Sender ter:Action ter:CannotDeleteControl	L'élément AnalyticsEngineControl spécifié ne peut pas être supprimé.	

18.6 GetAnalyticsState

GetAnalyticsState retourne des informations relatives à l'objet AnalyticsEngineControl référencé (voir Tableau 258). La structure AnalyticsStateInformation est développable. Le développement doit être utilisé pour acheminer des informations d'état supplémentaires relatives aux sous-structures. Par exemple, un élément AnalyticsEngine est composé de différents algorithmes d'analyse dont il convient de fournir les informations d'état. L'élément d'état d'AnalyticsStateInformation contient toujours un état agrégé de toutes les sous-structures.

Un service de dispositif d'analyse doit prendre en charge l'approvisionnement de ces informations d'état grâce à cette commande.

ConfigurationToken doit être l'identification de l'AnalyticsEngineControl pour lequel les informations d'état sont demandées

State doit contenir l'état agrégé de toutes les sous-structures d'AnalyticsStateInformation. Si l'état est "Error", l'erreur peut être renseignée avec une valeur définie par la mise en œuvre. Un dispositif doit appliquer les règles suivantes pour déterminer l'état agrégé:

Idle	L'état de toutes les sous-structures est "Idle"
PartiallyActive	Au moins une des sous-structures est à l'état "Active", toutes les autres étant à l'état "Idle".
Active	L'état de toutes les sous-structures est "Active"
Error	Au moins une des sous-structures est à l'état "Error"

Error, le cas échéant, doit contenir une valeur de chaîne définie par la mise en œuvre et décrivant l'erreur.

Tableau 258 – GetAnalyticsState

GetAnalyticsState		Demande-Réponse
Nom du message	Description	
GetAnalyticsStateRequest	<i>Contient le ConfigurationToken de l'élément AnalyticsEngineControl pour lequel obtenir l'état.</i> tt:ReferenceToken ConfigurationToken [1][1]	
GetAnalyticsStateResponse	<i>L'élément State doit contenir l'état d'AnalyticsEngineControl.</i> tt:AnalyticsStateInformation State[1][1]	
Codes de défaut	Description	
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl	<i>Le ConfigurationToken ne fait pas référence à un AnalyticsEngineControl existant.</i>	

18.7 Configuration de flux de sortie en continu

18.7.1 Généralités

Un service de dispositif d'analyse offre une interface de transmission continue en temps réel tel que spécifié à l'article "Transmission continue en temps réel" en agissant comme un serveur RTSP. Le jeton identifiant l'AnalyticsEngineControl est utilisé dans un service de dispositif d'analyse à la place du jeton identifiant le profil utilisé dans un profil multimédia.

18.7.2 Demande d'URI de flux

Cette opération demande un URI qui peut être utilisé pour initier un flux actif en utilisant RTSP en tant que protocole de commande. L'URI est valide uniquement tel qu'il est spécifié dans la réponse ou tant que le contrôle de moteur d'analyse n'est pas reconfiguré. Le service de dispositif d'analyse doit prendre en charge l'extraction d'un URI de flux pour un contrôle de moteur d'analyse spécifique grâce à la commande GetAnalyticsDeviceStreamUri (voir Tableau 259).

Tableau 259 – Commande GetAnalyticsDeviceStreamUri

GetAnalyticsDeviceStreamUri		Demande-Réponse
Nom du message	Description	
GetAnalyticsDeviceStreamUriRequest	<i>L'élément StreamSetup contient deux parties. Le StreamType définit si un flux multimédia à diffusion unilatérale ou multidiffusion est demandé. Transport spécifie une chaîne de protocoles de transport définissant la tunnellation du flux multimédia via différents protocoles réseau.</i> <i>L'élément AnalyticsEngineControlToken doit indiquer le contrôle de moteur d'analyse à utiliser.</i> tt:StreamSetup StreamSetup [1][1] tt:ReferenceToken AnalyticsEngineControlToken [1][1]	
GetAnalyticsDeviceStreamUriResponse	<i>Contient l'URI à utiliser dans le cadre de la demande de flux multimédia.</i> xs:anyURI Uri [1][1]	
Codes de défaut		Description
env:Sender ter:InvalidArgVal ter:NoAnalyticsEngineControl		<i>La configuration demandée indiquée avec AnalyticsEngineControlToken n'existe pas.</i>
env:Sender ter:InvalidArgVal ter:InvalidStreamSetup		<i>La spécification de la partie StreamType ou Transport dans StreamSetup n'est pas prise en charge.</i>
env:Sender ter:OperationProhibited ter:StreamConflict		<i>La spécification de la partie StreamType ou Transport dans StreamSetup génère un conflit avec d'autres flux.</i>

19 Contrôle d'enregistrement

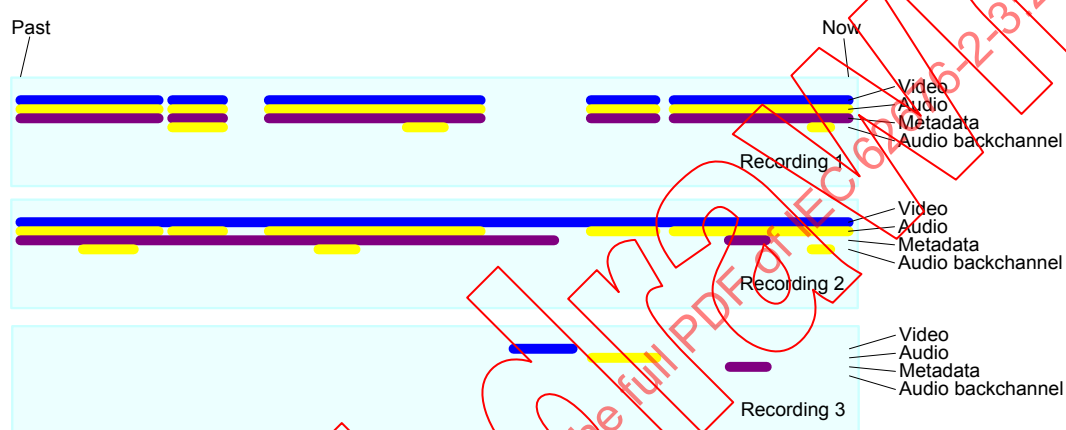
19.1 Généralités

Le service d'enregistrement permet à un client de gérer les enregistrements et de configurer le transfert de données entre les sources de données et les enregistrements. La gestion des enregistrements comprend la création et la suppression des enregistrements et des suivis, ainsi que le verrouillage et le déverrouillage des plages d'enregistrement et la suppression des données enregistrées.

L'enregistrement des travaux permet de transférer des données d'une source d'enregistrement vers un enregistrement. Une source d'enregistrement peut être un objet de réception créé avec le service de récepteur, ou un profil multimédia qui code les données sur un dispositif local. Le profil multimédia peut être utilisé comme une source sur une caméra à stockage intégré.

Le terme *enregistrement* est utilisé dans la présente norme pour indiquer un conteneur d'un ensemble de pistes audio, vidéo et de métadonnées. Un enregistrement peut comporter un certain nombre de pistes. Une piste est perçue comme une chronologie infinie contenant des données à certains moments.

La Figure 24 présente trois enregistrements, chacun avec une piste vidéo, une piste de métadonnées et deux pistes audio. Une deuxième piste audio est utilisée pour le stockage de la voie de retour audio.



IEC 2764/13

Légende

Anglais	Français
Past	Passé
Now	Présent
Video	Vidéo
Audio	Audio
Metadata	Métadonnées
Audio backchannel	Voie de retour audio
Recording	Enregistrement

Figure 24 – Exemple d'enregistrements et de pistes

Un enregistrement doit être au moins capable de contenir trois pistes: une piste audio, une piste vidéo et une piste de métadonnées. Certaines mises en œuvre du service d'enregistrement peuvent prendre en charge plusieurs pistes de chaque type.

Pour sauvegarder des données sur un enregistrement, un client crée en premier lieu un enregistrement et vérifie que l'enregistrement dispose des pistes nécessaires. Ensuite, le client crée un travail d'enregistrement qui extrait les données d'une ou de plusieurs sources et les stocke sur les pistes de l'enregistrement.

Les clients peuvent configurer plusieurs travaux d'enregistrement dans le même enregistrement. Si plusieurs travaux d'enregistrement sont actifs, le dispositif utilise un schéma de priorité pour faire un choix parmi les pistes définies dans les travaux d'enregistrement. Les clients peuvent à tout moment changer le mode des travaux

d'enregistrement, en permettant la mise en œuvre de caractéristiques telles que l'enregistrement d'alarme ou l'enregistrement manuel.

Le travail d'enregistrement s'appuie sur le service de récepteur pour recevoir des données provenant d'autres dispositifs grâce aux objets de récepteur identifiés par ReceiverTokens

Si un client utilise un objet de réception avec un seul travail d'enregistrement, le service d'enregistrement peut créer et supprimer automatiquement les objets de réception. La création automatique est signalée par le drapeau AutoCreateReceiver dans la structure de configuration du travail d'enregistrement. Les objets de réception créés de cette manière doivent être automatiquement supprimés si plus aucun travail d'enregistrement ne les utilise. Des valeurs vides doivent être attribuées aux champs d'un objet de réception automatiquement créé. Il convient que le client configure l'objet de réception après avoir créé le travail d'enregistrement.

La vue ONVIF des enregistrements est une vue logique indépendante de la manière dont les enregistrements sont physiquement stockés sur le disque. Par exemple, certaines mises en œuvre de caméra enregistrent des alarmes en créant un fichier distinct sur un système de fichiers FLAT pour chaque alarme déclenchée. Même si chaque fichier peut être représenté comme étant un enregistrement ONVIF différent, le modèle de la présente Norme a pour objet de regrouper tous ces fichiers dans un seul enregistrement. En recherchant l'événement "DataPresent" avec la méthode FindEvents du service de recherche, un client peut rechercher les heures de début de l'enregistrement vidéo et à quel endroit il s'est arrêté.

Si les métadonnées sont enregistrées, elles peuvent également contenir tous les événements générés par la source de données (voir l'Article 15 relatif au traitement d'événements et 11.10 relatif à l'objet MetadataConfiguration). De plus, d'un point de vue conceptuel, un dispositif enregistre également les événements d'historique définis par l'ONVIF (voir Enregistrement des descriptions d'événement dans le service de recherche), cela incluant des informations telles que le début et la fin d'une plage de données enregistrée. D'un point de vue conceptuel, un dispositif peut également enregistrer les événements d'historique spécifiques au fournisseur. Les événements générés par le dispositif ne sont pas insérés dans les pistes de métadonnées existantes des enregistrements. La méthode FindEvents du service de recherche peut trouver tous les événements enregistrés. La plupart des mises en œuvre de dispositif suppriment automatiquement les données enregistrées les plus anciennes du stockage afin de libérer de l'espace pour les nouveaux enregistrements. Les verrous offrent un mécanisme permettant à l'utilisateur de sélectionner des plages de données. Une plage de données verrouillée n'est pas automatiquement supprimée. La prise en charge des verrous est réservée aux éditions ultérieures de la norme.

19.2 Exigences générales

Tous les objets créés dans le service d'enregistrement doivent être persistants, c'est-à-dire qu'ils doivent résister à un cycle d'alimentation. De même, toutes les données de configuration des objets doivent être persistantes.

19.3 Structures de données

19.3.1 RecordingConfiguration

La structure RecordingConfiguration doit être utilisée pour configurer les enregistrements grâce à CreateRecordings et Get/SetRecordingConfiguration.

MaximumRetentionTime spécifie la durée maximale pendant laquelle les données d'une piste de l'enregistrement doivent être stockées. Le dispositif doit supprimer toutes les données plus anciennes que le temps de rétention maximal. Ces données ne doivent plus être accessibles. Si la valeur 0 est attribuée à MaximumRetentionPeriod, le dispositif ne doit pas limiter le temps de rétention des données stockées, sauf par des contraintes de ressource. Quelle que soit la valeur de MaximumRetentionTime, le dispositif peut

automatiquement supprimer des enregistrements pour libérer de l'espace de stockage pour de nouveaux enregistrements.

Aucun des autres champs définis dans cette structure ne doit être utilisé par le dispositif. Elle stocke simplement ces informations et doit les retourner grâce aux méthodes *GetRecordingConfiguration* et *GetRecordingInformation* (voir 20.5).

19.3.2 TrackConfiguration

La structure *TrackConfiguration* doit être utilisée pour configurer les pistes grâce à *CreateTrack* et *Get/SetTrackConfiguration*.

TrackConfiguration contient les champs suivants:

TrackType définit le type de données de la piste. Il doit être égal aux chaînes "Video", "Audio" ou "Metadata". La piste doit uniquement pouvoir contenir des données de ce type.

Aucun des autres champs définis dans cette structure ne doit être utilisé par le dispositif. Elle stocke simplement ces informations et doit les retourner grâce aux méthodes *GetTrackConfiguration* et *GetRecordingInformation* (voir 20.5).

19.3.3 RecordingJobConfiguration

La structure *RecordingJobConfiguration* doit contenir la configuration d'un travail d'enregistrement. La Figure 25 illustre *RecordingJobConfiguration* comme un schéma UML.

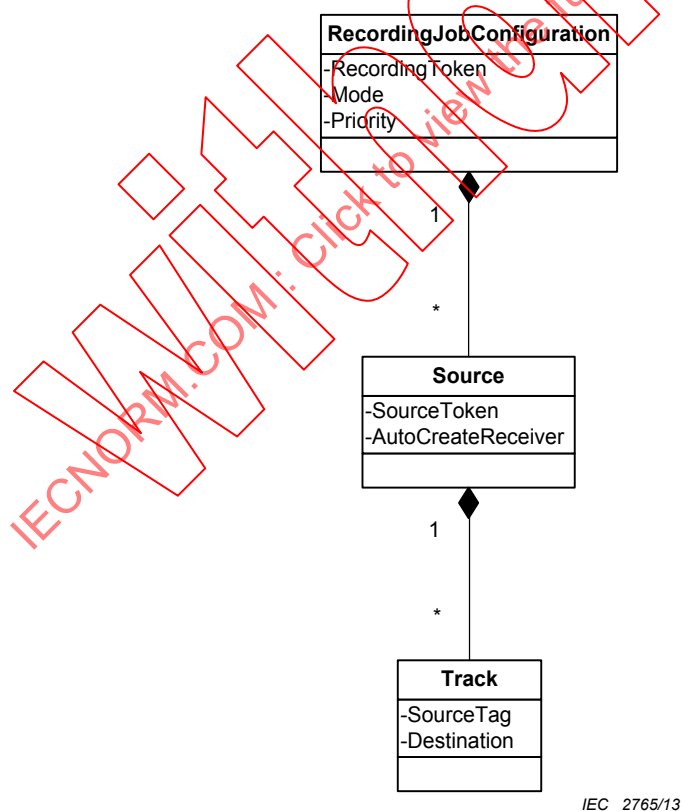


Figure 25 – Schéma des éléments *RecordingJobConfiguration*

RecordingJobConfiguration contient les champs suivants:

RecordingToken: identifie l'enregistrement dans lequel ce travail doit stocker les données reçues.

Mode: mode du travail. Si le mode est "Idle", il ne doit rien se passer. Si le mode est "Active", le dispositif doit tenter d'obtenir les données auprès des récepteurs. Un client doit utiliser `GetRecordingJobState` pour déterminer si le transfert de données a réellement lieu. Les seules valeurs valides de Mode doivent être "Idle" et "Active".

Priority: Il doit s'agir d'un nombre positif. Si plusieurs travaux d'enregistrement stockent des données sur la même piste, le dispositif stocke uniquement les données dont le travail d'enregistrement présente la priorité la plus élevée. La priorité est spécifiée par travail d'enregistrement, mais le dispositif doit déterminer la priorité de chaque piste individuellement. Si deux travaux d'enregistrement présentent la même priorité, le dispositif doit enregistrer les données correspondant au travail d'enregistrement qui a été activé en dernier.

La valeur 0 indique la priorité la plus basse. Des valeurs plus élevées doivent indiquer une priorité plus élevée.

SourceToken: Ce champ doit être une référence à la source des données. Le type de la source est déterminé par l'attribut Type de la structure `SourceToken`. Si le Type est `http://www.onvif.org/ver10/schema/Receiver`, le jeton est un `ReceiverReference`. Dans ce cas, le dispositif doit recevoir les données sur le réseau. Si le Type est `http://www.onvif.org/ver10/schema/Profile`, le jeton identifie un profil multimédia, demandant au dispositif d'obtenir les données auprès d'un profil qui existe sur le dispositif local.

Si **SourceToken** est ignoré, **AutoCreateReceiver** doit être vrai.

AutoCreateReceiver: Si la valeur de ce champ est TRUE, et que **SourceToken** est ignoré, le dispositif doit créer un objet de récepteur (grâce au service de récepteur) et attribuer `ReceiverReference` au champ **SourceToken**. Lors de l'extraction de `RecordingJobConfiguration` du dispositif, le champ **AutoCreateReceiver** ne doit jamais être présent.

SourceTag: Si le flux RTSP reçu contient plusieurs pistes de même type, **SourceTag** fait la distinction entre elles.

Destination: La destination est le jeton de la piste sur laquelle le dispositif doit stocker les données reçues.

19.4 CreateRecording

`CreateRecording` doit créer un enregistrement (voir Tableau 260). Le nouvel enregistrement doit être créé avec une piste vidéo, une piste audio et une piste de métadonnées.

Cette méthode est facultative. Elle doit être disponible si la valeur de la fonctionnalité `Recording/DynamicRecordings` est TRUE.

Tableau 260 – Commande CreateRecording

CreateRecording	
Nom du message	Description
CreateRecordingRequest	Contient la configuration initiale de l'enregistrement. tt:RecordingConfiguration RecordingConfiguration [1][1]
CreateRecordingResponse	Retourne la référence à l'enregistrement créé tt:RecordingReference RecordingToken [1][1]
Codes de défaut	Description
env:Receiver ter:Action ter:MaxRecordings	Le dispositif ne peut pas créer d'enregistrement car il a déjà atteint le nombre maximal d'enregistrements qu'il prend en charge.
env:Sender ter:InvalidArgVal ter:BadConfiguration	RecordConfiguration n'est pas valide.
env:Receiver ter:ActionNotSupported ter:NotImplemented	La méthode facultative n'est pas mise en œuvre

S'il a été exécuté avec succès, CreateRecording doit avoir créé trois pistes avec les configurations suivantes:

TrackToken	TrackType
VIDEO001	Vidéo
AUDIO001	Audio
META001	Métadonnées

La valeur 0 (illimité) doit être attribuée à l'élément MaximumRetentionTime de toutes les TrackConfigurations, une chaîne vide devant être attribuée à Description.

19.5 DeleteRecording

DeleteRecording doit supprimer un objet d'enregistrement (voir Tableau 261). A chaque fois qu'un enregistrement est supprimé, le dispositif doit supprimer toutes les pistes qui font partie de l'enregistrement et doit supprimer tous les travaux d'enregistrement. Pour chaque travail d'enregistrement supprimé, le dispositif doit également supprimer tous les objets de récepteur associés au travail d'enregistrement automatiquement créés à l'aide du champ AutoCreateReceiver de la structure de configuration du travail d'enregistrement, et qui ne sont pas utilisés dans un autre travail d'enregistrement.

Cette méthode est facultative. Elle doit être disponible si la valeur de la fonctionnalité Recording/DynamicRecordings est TRUE.

Tableau 261 – Commande DeleteRecording

DeleteRecording	
Nom du message	Description
DeleteRecordingRequest	Identifie l'enregistrement qui doit être supprimé tt:RecordingReference RecordingToken [1][1]
DeleteRecordingResponse	Ce message doit être vide.
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecording	Le RecordingToken ne fait pas référence à un enregistrement existant.
env:Receiver ter: ActionNotSupported ter:NotImplemented	Le dispositif ne peut pas supprimer les enregistrements
env:Receiver ter:Action ter:CannotDelete	Cet enregistrement spécifique ne peut pas être supprimé

19.6 GetRecordings

GetRecordings doit retourner une description de tous les enregistrements dans le dispositif (voir Tableau 262). Cette description doit inclure une liste de toutes les pistes pour chaque enregistrement.

Tableau 262 – Commande GetRecordings

GetRecordings	
Nom du message	Description
GetRecordingsRequest	Il doit s'agir d'un message vide
GetRecordingsResponse	RecordingItem identifie un enregistrement et sa configuration en cours tt:GetRecordingsResponseItem RecordingItem [0][non limité]
Codes de défaut	Description
Pas de défauts spécifiques à la commande	

19.7 SetRecordingConfiguration

SetRecordingConfiguration doit modifier la configuration d'un enregistrement (voir Tableau 263).

Tableau 263 – Commande SetRecordingConfiguration

SetRecordingConfiguration	
Nom du message	Description
SetRecordingConfigurationRequest	RecordingToken doit identifier l'enregistrement qui doit être modifié. RecordingConfiguration doit être la nouvelle configuration de cet enregistrement tt:RecordingReference RecordingToken[1][1] tt:RecordingConfiguration RecordingConfiguration[1][1]
SetRecordingConfigurationResponse	Ce message doit être vide.
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:BadConfiguration	La configuration n'est pas valide.
env:Sender ter:InvalidArgVal ter:NoRecording	Le <i>RecordingToken</i> ne fait pas référence à un enregistrement existant.

19.8 GetRecordingConfiguration

GetRecordingConfiguration doit extraire la configuration d'un enregistrement (voir Tableau 264).

Tableau 264 – Commande GetRecordingConfiguration

GetRecordingConfiguration	
Nom du message	Description
GetRecordingConfigurationRequest	RecordingToken doit identifier l'enregistrement dont la configuration doit être extraite. tt:RecordingReference RecordingToken[1][1]
GetRecordingConfigurationResponse	RecordingConfiguration doit être la configuration en cours de l'enregistrement spécifié tt:RecordingConfiguration RecordingConfiguration[1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecording	Le <i>RecordingToken</i> ne fait pas référence à un enregistrement existant.

19.9 CreateTrack

Cette méthode doit créer une piste dans un enregistrement (voir Tableau 265).

Cette méthode est facultative. Elle doit être disponible si la valeur de la fonctionnalité Recording/DynamicTracks est TRUE.

Tableau 265 – Commande CreateTrack

CreateTrack	
Nom du message	Description
CreateTrackRequest	RecordingToken doit identifier l'enregistrement auquel une piste doit être ajoutée. TrackConfiguration doit fournir la configuration de la nouvelle piste. tt:RecordingReference RecordingToken[1][1] tt:TrackConfiguration TrackConfiguration[1][1]
CreateTrackResponse	TrackToken doit identifier la piste qui vient d'être créée. TrackToken doit être unique dans l'enregistrement auquel appartient la nouvelle piste. tt:TrackReference TrackToken[1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecording	Le RecordingToken ne fait pas référence à un enregistrement existant.
env:Receiver ter:Action ter:MaxTracks	La piste ne peut pas être créée car le nombre maximal de pistes que le dispositif prend en charge pour cet enregistrement a été atteint.
env:Sender ter:InvalidArgVal ter:BadConfiguration	TrackConfiguration n'est pas valide.
env:Receiver ter:ActionNotSupported ter:NotImplemented	La méthode facultative n'est pas mise en œuvre

Un **TrackToken** en lui-même n'identifie pas de manière unique une piste particulière. Les pistes de différents enregistrements peuvent avoir le même **TrackToken**.

19.10 DeleteTrack

DeleteTrack doit supprimer une piste d'un enregistrement. Toutes les données de la piste doivent être supprimées. Cette méthode est facultative. Elle doit être disponible si la valeur de la fonctionnalité Recording/DynamicTracks est TRUE (voir Tableau 266).

Tableau 266 – Commande DeleteTrack

DeleteTrack	
Nom du message	Description
DeleteTrackRequest	RecordingToken doit identifier l'enregistrement duquel supprimer la piste. TrackToken identifie la piste à supprimer. tt:RecordingReference RecordingToken[1][1] tt:TrackReference TrackToken[1][1]
DeleteTrackResponse	Ce message doit être vide.
Codes de défaut	Description
env:Receiver ter:ActionNotSupported ter:NotImplemented	Le dispositif ne met pas en œuvre la méthode DeleteTrack.
env:Sender ter:InvalidArgVal ter:NoTrack	TrackToken ne fait pas référence à une piste existante de l'enregistrement.
env:Sender ter:InvalidArgVal ter:NoRecording	Le RecordingToken ne fait pas référence à un enregistrement existant.
env:Receiver ter:Action ter:CannotDelete	Cette piste spécifique ne peut pas être supprimée

19.11 GetTrackConfiguration

GetTrackConfiguration doit extraire la configuration d'une piste spécifique (voir Tableau 267).

Tableau 267 – Commande GetTrackConfiguration

GetTrackConfiguration	
Nom du message	Description
GetTrackConfigurationRequest	RecordingToken et TrackToken doivent identifier l'enregistrement duquel extraire la configuration de la piste. tt:RecordingReference RecordingToken[1][1] tt:TrackReference TrackToken[1][1]
GetTrackConfigurationResponse	tt:TrackConfiguration TrackConfiguration [1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoTrack	<i>TrackToken ne fait pas référence à une piste existante de l'enregistrement.</i>
env:Sender ter:InvalidArgVal ter:NoRecording	<i>Le RecordingToken ne fait pas référence à un enregistrement existant.</i>

19.12 SetTrackConfiguration

SetTrackConfiguration doit modifier la configuration d'une piste (voir Tableau 268).

Tableau 268 – Commande SetTrackConfiguration

SetTrackConfiguration	
Nom du message	Description
SetTrackConfigurationRequest	RecordingToken et TrackToken doivent identifier la piste pour laquelle définir la configuration de piste. TrackConfiguration est la nouvelle configuration de la piste. tt:RecordingReference RecordingToken[1][1] tt:TrackReference TrackToken[1][1] tt:TrackConfiguration TrackConfiguration[1][1]
SetTrackConfigurationResponse	Ce message doit être vide.
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoTrack	<i>TrackToken ne fait pas référence à une piste existante de l'enregistrement.</i>
env:Sender ter:InvalidArgVal ter:NoRecording	<i>Le RecordingToken ne fait pas référence à un enregistrement existant.</i>
env:Sender ter:InvalidArgVal ter:BadConfiguration	<i>Le contenu de l'objet de configuration n'est pas valide.</i>

19.13 CreateRecordingJob

CreateRecordingJob doit créer un travail d'enregistrement (voir Tableau 269).

Tableau 269 – Commande CreateRecordingJob

CreateRecordingJob	
Nom du message	Description
CreateRecordingJobRequest	JobConfiguration doit contenir la configuration du nouveau travail d'enregistrement. tt:RecordingJobConfiguration JobConfiguration[1][1]
CreateRecordingJobResponse	JobToken doit identifier le travail d'enregistrement créé. La structure JobConfiguration doit être la configuration telle qu'elle est utilisée par le dispositif. Elle peut être différente de la structure JobConfiguration transmise à <i>CreateRecordingJob</i>. tt:RecordingJobReference JobToken[1][1] tt:RecordingJobConfiguration JobConfiguration[1][1]
Codes de défaut	Description
env:Receiver ter:Action ter:MaxRecordingJobs	Le nombre maximal de travaux d'enregistrement que le dispositif peut gérer a été atteint.
env:Sender ter:InvalidArgVal ter:BadConfiguration	Le contenu de JobConfiguration n'est pas valide.
env:Receiver ter:Action ter:MaxReceivers	Si la valeur TRUE est attribuée au drapeau AutoCreateReceivers , cette erreur peut être retournée si le service de récepteur ne peut pas créer de récepteur.

Le **JobConfiguration** retourné depuis *CreateRecordingJob* doit être identique au **JobConfiguration** transmis dans *CreateRecordingJob*, sauf pour **ReceiverToken** et **AutoCreateReceiver**. Dans la structure retournée, **ReceiverToken** doit être présent et valide, et le champ **AutoCreateReceiver** doit être ignoré.

19.14 DeleteRecordingJob

DeleteRecordingJob supprime un travail d'enregistrement (voir Tableau 270). Il doit également supprimer de manière implicite tous les objets de récepteur associés au travail d'enregistrement automatiquement créés à l'aide du champ **AutoCreateReceiver** de la structure de configuration du travail d'enregistrement, et qui ne sont pas utilisés dans un autre travail d'enregistrement.

Tableau 270 – Commande DeleteRecordingJob

DeleteRecordingJob	
Nom du message	Description
DeleteRecordingJobRequest	<i>JobToken doit identifier le travail d'enregistrement qui doit être supprimé.</i> tt:RecordingJobReference JobToken [1][1]
DeleteRecordingJobResponse	<i>Le message doit être vide.</i>
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	<i>JobToken ne fait pas référence à un travail existant.</i>

19.15 GetRecordingJobs

GetRecordingJobs doit retourner une liste de tous les travaux d'enregistrement dans le dispositif (voir Tableau 271).

Tableau 271 – Commande GetRecordingJobs

GetRecordingJobs	
Nom du message	Description
GetRecordingJobsRequest	<i>Ce message doit être vide.</i>
GetRecordingJobsResponse	<i>JobItem identifie un travail dans le dispositif et contient sa configuration en cours.</i> tt:GetRecordingJobsResponseItem JobItem [0][non limité]
Codes de défaut	Description
<i>Pas de défauts spécifiques à la commande</i>	

19.16 SetRecordingJobConfiguration

SetRecordingJobConfiguration doit modifier la configuration d'un travail d'enregistrement (voir Tableau 272).

Tableau 272 – Commande SetRecordingJobConfiguration

SetRecordingJobConfiguration	
Nom du message	Description
SetRecordingJobConfiguration Request	Le <i>JobConfiguration</i> retourné depuis <i>SetRecordingJobConfiguration</i> doit être identique au <i>JobConfiguration</i> transmis dans <i>SetRecordingJobConfiguration</i>, sauf pour <i>ReceiverToken</i> et <i>AutoCreateReceiver</i>. Dans la structure retournée, <i>ReceiverToken</i> doit être présent et valide, et le champ <i>AutoCreateReceiver</i> doit être ignoré. tt:RecordingJobReference JobToken[1][1] tt:RecordingJobConfiguration JobConfiguration[1][1]
SetRecordingJobConfiguration Response	La structure JobConfiguration doit être la configuration telle qu'elle est utilisée par le dispositif. Elle peut être différente de la structure <i>JobConfiguration</i> transmise à <i>CreateRecordingJob</i>. tt:RecordingJobConfiguration JobConfiguration[1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	<i>JobToken</i> ne fait pas référence à un travail existant.
env:Sender ter:InvalidArgVal ter:BadConfiguration	Le contenu de <i>JobConfiguration</i> n'est pas valide.
env:Receiver ter:Action ter:MaxReceivers	Si la valeur <i>TRUE</i> est attribuée au drapeau <i>AutoCreateReceivers</i> , cette erreur peut être retournée si le service de récepteur ne peut pas créer de récepteur.

SetRecordingJobConfiguration doit supprimer de manière implicite tous les objets de récepteur qui ont été créés automatiquement s'ils ne sont plus utilisés suite à une modification de la configuration du travail d'enregistrement.

19.17 GetRecordingJobConfiguration

GetRecordingJobConfiguration doit retourner la configuration en cours d'un travail d'enregistrement (voir Tableau 273).

Tableau 273 – Commande GetRecordingJobConfiguration

GetRecordingJobConfiguration	
Nom du message	Description
GetRecordingJobConfiguration Request	JobToken doit identifier le travail d'enregistrement pour lequel extraire la configuration. tt:RecordingJobReference JobToken [1][1]
GetRecordingJobConfiguration Response	JobConfiguration doit contenir la configuration en cours du travail d'enregistrement. tt:RecordingJobConfiguration JobConfiguration [1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	<i>JobToken ne fait pas référence à un travail existant.</i>

19.18 SetRecordingJobMode

SetRecordingJobMode doit modifier le mode du travail d'enregistrement (voir Tableau 274). L'utilisation de cette méthode doit équivaloir à extraire la configuration du travail d'enregistrement et la réécrire dans un mode différent.

Tableau 274 – Commande SetRecordingJobMode

SetRecordingJobMode	
Nom du message	Description
SetRecordingJobModeRequest	JobToken doit identifier le travail d'enregistrement pour lequel modifier le mode d'enregistrement. Le mode doit être le nouveau mode du travail d'enregistrement. tt:RecordingJobReference JobToken [1][1] tt:RecordingJobMode Mode [1][1]
SetRecordingJobModeResponse	<i>Ce message doit être vide.</i>
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	<i>JobToken ne fait pas référence à un travail existant.</i>
env:Sender ter:InvalidArgVal ter:BadMode	<i>Le Mode n'est pas valide.</i>

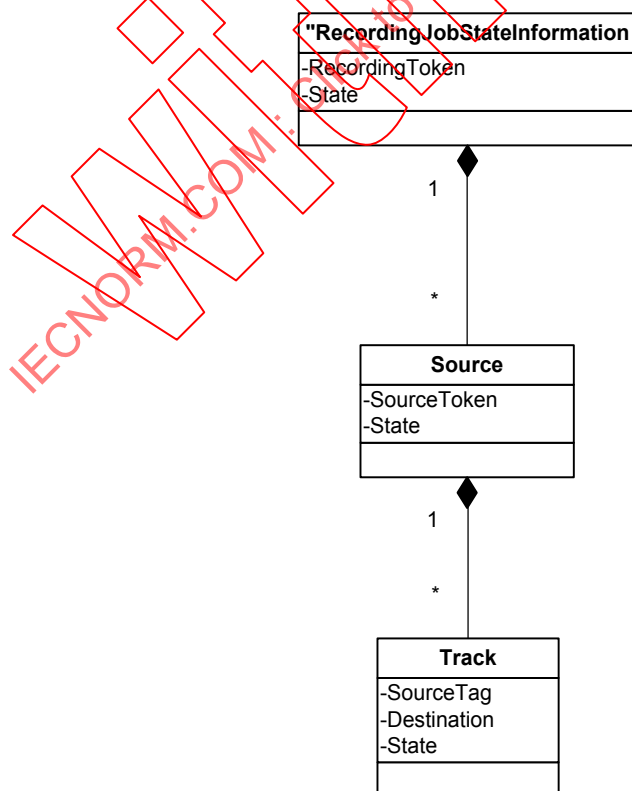
19.19 GetRecordingJobState

GetRecordingJobState retourne le statut d'un travail d'enregistrement (voir Tableau 275). Il inclut un état agrégé et un état pour chaque piste du travail d'enregistrement.

Tableau 275 – Commande GetRecordingJobState

GetRecordingJobState	
Nom du message	Description
GetRecordingJobState Request	JobToken doit identifier le travail d'enregistrement pour lequel extraire l'état. tt:RecordingJobReference JobToken[1][1]
GetRecordingJobState Response	L'élément State doit contenir l'état du travail d'enregistrement. tt:RecordingJobStateInformation State[1][1]
Codes de défaut	Description
env:Sender ter:InvalidArgVal ter:NoRecordingJob	<i>JobToken ne fait pas référence à un travail existant.</i>

La représentation UML de la structure RecordingJobStateInformation est illustrée en Figure 26.



IEC 2766/13

Figure 26 – Schéma des éléments RecordingJobStateInformation

RecordingToken doit être l'identification de l'enregistrement dans lequel s'enregistre le travail d'enregistrement.

State (partie de RecordingJobStateInformation) doit contenir l'état agrégé sur l'ensemble de la structure RecordingJobInformation.

SourceToken doit identifier la source de données du travail d'enregistrement.

State (partie de RecordingJobStateSource) doit contenir l'état agrégé sur toutes les sous-structures de RecordingJobStateSource.

SourceTag doit identifier la piste de la source de données qui fournit les données.

Destination doit indiquer la piste de destination

State (partie de RecordingJobTrackState) doit fournir l'état de travail de la piste. Les valeurs valides de l'état doivent être "Idle", "Active" et "Error". Si l'état est "Error", le champ Error peut être renseigné avec une valeur définie par la mise en œuvre.

Error, le cas échéant, doit contenir une valeur de chaîne définie par la mise en œuvre et décrivant l'erreur. Il convient que la chaîne soit en anglais.

Un dispositif doit appliquer les règles suivantes pour déterminer l'état agrégé

Idle	Toutes les valeurs d'état des sous-nœuds sont "Idle"
PartiallyActive	L'état de certains sous-nœuds est "Active" et celui d'autres sous-nœuds est "Idle"
Active	L'état de tous les sous-nœuds est "Active"
Error	Au moins un des sous-nœuds est à l'état "Error"

19.20 Événements

Le service d'enregistrement doit expédier les événements grâce au service d'événement. Il doit être capable de générer les événements figurant dans cet article à chaque fois que la condition qui déclenche l'événement se produit.

Certains de ces événements s'apparentent à ceux générés automatiquement et qui peuvent faire l'objet d'une recherche par la méthode FindEvents du service de recherche. Voir 20.17.

19.20.1 Enregistrement des modifications d'état de travail

Si le champ d'état de la structure RecordingJobStateInformation change, le dispositif doit envoyer l'événement suivant:

```

Topic: tns1:RecordingConfig/JobState
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingJobToken"
  Type="tt:RecordingJobReference"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItemDescription Name="State" Type="xs:String"/>
    <tt:ElementItemDescription Name="Information"
  Type="tt:RecordingJobStateInformation"/>
  </tt>Data>
</tt:MessageDescription>

```

19.20.2 Modifications de configuration

Si la configuration d'un enregistrement est modifiée, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/RecordingConfiguration
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItemDescription Name="Configuration"
Type="tt:RecordingConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

Si la configuration d'une piste est modifiée, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/TrackConfiguration
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItemDescription Name="Configuration"
Type="tt:TrackConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

Si la configuration d'un travail enregistrement est modifiée, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/RecordingJobConfiguration
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingJobToken"
Type="tt:RecordingJobReference"/>
  </tt:Source>
  <tt>Data>
    <tt:ElementItemDescription Name="Configuration"
Type="tt:RecordingJobConfiguration"/>
  </tt>Data>
</tt:MessageDescription>
```

19.20.3 Suppression de données

A chaque fois que des données sont supprimées, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/DeleteTrackData
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt>Data>
    <tt:SimpleItemDescription Name="StartTime" Type="xsDateTime"/>
    <tt:SimpleItemDescription Name="EndTime" Type="xsDateTime"/>
  </tt>Data>
</tt:MessageDescription>
```


19.20.4 Enregistrement et création et suppression de piste

A chaque fois qu'un enregistrement est créé, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/CreateRecording
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

A chaque fois qu'un enregistrement est supprimé, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/DeleteRecording
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

A chaque fois qu'une piste est créée, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/CreateTrack
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

A chaque fois qu'une piste est supprimée, le dispositif doit envoyer l'événement suivant:

```
Topic: tns1:RecordingConfig/DeleteTrack
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="RecordingToken"
Type="tt:RecordingReference"/>
    <tt:SimpleItemDescription Name="TrackToken" Type="tt:TrackReference"/>
  </tt:Source>
  <tt:Data>
  </tt:Data>
</tt:MessageDescription>
```

19.21 Exemples

19.21.1 Exemple 1: Configuration de l'enregistrement d'une seule caméra

L'opération comporte deux étapes. La première consiste à configurer le NVS

```
; Create recording (this implicitly creates an A, V and M track)
```

```
RecordToken = CreateRecording(RecordConfiguration)
```

; The tracktokens are predefined. We don't have to find them on the device

```
TrackToken1 = "VIDEO001"
```

```
TrackToken2 = "AUDIO001"
```

```
TrackToken3 = "META001"
```

; Create a recording job, assume that we set mode to idle, auto create receiver

```
JobToken, ActualJobConfig = CreateRecordingJob(JobConfiguration)
```

; Configure the receiver

```
ConfigureReceiver(ActualJobConfiguration.ReceiverToken,
ReceiverConfiguration)
```

Il s'agit de l'étape de configuration.

Enfin, pour véritablement démarrer l'enregistrement, une entité appelle

```
; Activate the recording job
SetRecordingJobMode(JobToken, Active)
```

pour activer le travail. Cela permet au NVS de configurer une connexion RTSP avec le dispositif.

Par conséquent, pour démarrer et arrêter l'enregistrement, tout cela est nécessaire pour appeler SetRecordingJobMode sur des travaux d'enregistrement préalablement configurés. Et étant donné que les objets de configuration intégrés sont persistants, il est nécessaire de réaliser le cycle de configuration une fois seulement.

19.21.2 Exemple 2: Enregistrement de plusieurs flux d'une caméra vers un seul enregistrement

Cet exemple est très similaire à l'exemple 1. La configuration du travail contient des références à deux objets de récepteur. Chaque objet de récepteur est configuré à partir du même dispositif, mais à partir d'un autre flux.

```
; Create recording (this implicitly creates an A, V and M track)
```

```
RecordToken = CreateRecording(RecordConfiguration)
```

; The tracktokens are predefined. We don't have to find them on the device

```
TrackToken1 = "VIDEO001"
```

```
TrackToken2 = "AUDIO001"
```

```
TrackToken3 = "META001"
```

```
; Create three additional tracks
```

```
TrackToken4 = CreateTrack(RecordToken, AudioConfig)
```

```
TrackToken5 = CreateTrack(RecordToken, VideoConfig)
```

```
TrackToken6 = CreateTrack(RecordToken, MetadataConfig)
```

; Create a recording job, assume that we set mode to idle, auto create two receivers

```
JobToken, ActualJobConfiguration = CreateRecordingJob(JobConfiguration)
```

```
; Configure the receivers
```

```
ConfigureReceiver(ActualJobConfiguration.ReceiverToken[1],
```

```
Receiver1Configuration)
```

```
ConfigureReceiver(ActualJobConfiguration.ReceiverToken[2],
```

```
Receiver2Configuration)
```

Pour véritablement démarrer l'enregistrement, une entité appelle

```
; Activate the recording job
```

```
SetRecordingJobMode(JobToken, Active)
```

20 Recherche d'enregistrement

20.1 Généralités

Le service de recherche propose un certain nombre d'opérations de recherche de données dans un ensemble d'enregistrements. Le meilleur moyen de procéder consiste à rechercher

les événements inclus dans la piste de métadonnées d'un enregistrement ou associés à un enregistrement dans le dispositif (voir Enregistrement des événements ci-dessous).

GetRecordingSummary retourne un récapitulatif pour tous les enregistrements, et peut être utilisé pour fournir l'échelle d'une chronologie.

GetRecordingInformation retourne des informations relatives à un seul enregistrement (l'heure de début et le statut en cours, par exemple).

GetMediaAttributes retourne les attributs multimédia d'un enregistrement à un instant donné.

La recherche réelle a lieu sous la forme d'opérations couplées de recherche et de résultats. Chaque opération de recherche lance une session de recherche. Le client peut alors obtenir les résultats issus de la session de recherche par incrément ou en totalité, selon la mise en œuvre et l'étendue de la recherche. Il existe quatre paires d'opérations de recherche pour les enregistrements, les événements d'enregistrement, les positions PTZ et les métadonnées.

GetSearchState retourne l'état d'une session de recherche.

EndSearch met fin à une session de recherche, en interrompant la recherche et en retournant toutes les opérations de résultat bloquantes.

20.2 Concepts

20.2.1 Direction de recherche

Une recherche est réalisée à partir d'un point de départ de la chronologie vers un point d'arrivée. Si le point d'arrivée précède le point de départ, la recherche est réalisée en sens inverse. Cela peut être utile si l'événement correspondant le plus récent présente un intérêt ou s'il est commode d'extraire les résultats du plus récent au plus ancien.

Si aucun point d'arrivée n'est précisé, la recherche est toujours réalisée à partir du point de départ.

20.2.2 Événement d'enregistrement

Décrit un événement discret lié à l'enregistrement. Il est représenté sous la forme d'un message de notification, ce qui ne signifie pas nécessairement qu'il a été enregistré en tant que notification. Les événements d'enregistrement peuvent être des notifications incluses dans une piste de métadonnées enregistrées, créés par le dispositif d'enregistrement suite à un événement ou un mécanisme internes, ou insérés par un client à l'aide d'une demande WebService ou d'un flux de métadonnées. Toutefois, l'événement d'enregistrement a été créé et associé à un enregistrement particulier, la présente spécification ne précisant pas comment il est stocké en interne sur le dispositif, mais uniquement comment il convient de le représenter dans l'interface.

Même créés, les événements d'enregistrement sont toujours traités comme des notifications par rapport aux filtres de recherche et aux résultats retournés. Chaque événement d'enregistrement comporte une rubrique de notification telle que définie en 15.7 de la spécification principale. Les événements d'enregistrement prédéfinis sont décrits en 20.17.

Pour communiquer l'état d'origine des événements de propriété, des événements d'état de démarrage virtuels peuvent être retournés dans un résultat de recherche contenant la valeur d'une ou de plusieurs propriétés au point de départ de l'intervalle de recherche. Ces événements d'état de départ sont virtuels dans le sens où ils sont créés à la volée par le serveur, plutôt que collectés à partir des données enregistrées. Si le client indique que ces événements sont souhaités en définissant le drapeau approprié, il convient que les événements virtuels correspondant aux rubriques définies dans le filtre de recherche soient retournés pour tous les enregistrements de l'étendue de la recherche.

20.2.3 Session de recherche

Une session de recherche est démarrée de manière asynchrone par une opération de recherche et est identifiée par un jeton de recherche unique à cette session. Les résultats sont retournés par incréments dans des opérations `GetResult` se rapportant à la session créée par l'opération de recherche. La recherche peut prendre fin de trois manières:

- Expiration du délai `KeepAlive` – si aucune demande faisant référence à une session particulière n'est formulée par un client dans le temps imparti, la recherche prend fin;
- Une méthode `GetResult` retourne les dernières données de la session de recherche en attribuant la valeur "Completed" (terminé) dans le résultat de l'état de recherche;
- `EndSearch` – le client met explicitement fin à une session.

La fin d'une session annule une recherche en cours, le retour immédiat et la formulation d'autres requêtes adressées à la même session donnant lieu à un message d'erreur. Un dispositif ne doit pas réutiliser immédiatement le jeton, cela risquant de troubler les clients ne sachant pas qu'une session a pris fin.

20.2.4 Etendue de la recherche

L'étendue contient un certain nombre d'éléments facultatifs, chacun limitant l'ensemble de données à consulter lors de l'exécution de recherches.

20.2.4.1 Données incluses

Le client peut éventuellement définir les sources et les enregistrements dans lesquels procéder à la recherche en spécifiant des listes de jetons pour chaque type. En présence de plusieurs types, l'union des jetons spécifiés doit être utilisée. Si aucune source ni aucun jeton d'enregistrement n'est spécifié, tous les enregistrements doivent être inclus. L'étendue est affinée par le filtre d'informations d'enregistrement. Toutefois, si des enregistrements sont spécifiés, les filtres sont uniquement appliqués à ce sous-ensemble d'enregistrements.

20.2.4.2 Filtre d'informations d'enregistrement

Plutôt que de spécifier une liste de jetons d'enregistrement, les enregistrements peuvent être filtrés par un filtre XPath fonctionnant sur la structure `RecordingInformation`. Cela permet au client de filtrer tous les éléments présents dans la structure `RecordingInformation`, en utilisant des comparaisons en fonction du dialecte XPath défini en 20.18. Si un filtre d'informations d'enregistrement est fourni, seuls les enregistrements correspondant au filtre doivent faire partie de l'étendue.

Exemple de filtre incluant uniquement des enregistrements contenant l'élément audio dans l'étendue de la recherche:

```
boolean(//Tracks[TrackType = "Audio"])
```

20.2.5 Filtres de recherche

Les filtres de recherche sont spécifiques aux opérations de recherche. Voir `FindEvents`, `FindPTZPosition` et `FindMetadata`, respectivement. Ils agissent tous sur les enregistrements définis par l'étendue.

20.3 Structures de données

20.3.1 Structure `RecordingInformation`

`RecordingInformation` contient des informations relatives à un enregistrement, aux pistes dont il est composé et à la source.

- `RecordingToken` – identifiant unique de l'enregistrement;

- **EarliestRecording** – date et heure des données les plus anciennes dans l'enregistrement;
- **LatestRecording** – date et heure des données les plus récentes dans l'enregistrement;
- **Content** – description informative du contenu;
- **RecordingStatus** – statut en cours de l'enregistrement, pouvant être l'un des suivants: Initiated, Recording, Stopped, Removing, Removed;
- **RecordingSourceInformation** – structure contenant des informations relatives à la source de l'enregistrement;
- **TrackInformation** – liste des structures d'informations de piste.

20.3.2 Structure RecordingSourceInformation

Contient des informations relatives à la source d'un enregistrement.

- **SourceId** – identifiant de la source choisie par le client qui crée l'enregistrement. Cet identifiant est opaque pour le NVS. Les clients peuvent utiliser n'importe quel type d'URI pour ce champ;
- **Name** – nom informatif de la source;
- **Location** – description informative de l'emplacement de la source;
- **Description** – description informative de la source;
- **Address** – URI informatif de la source.

20.3.3 Structure TrackInformation

Contient des informations relatives à une seule piste dans un enregistrement.

- **TrackToken** – identifiant de la piste. TrackToken est unique parmi tous ceux utilisés dans un enregistrement;
- **TrackType** – identifie le type de piste (vidéo, audio ou métadonnées);
- **Description** – description informative de la piste;
- **DataFrom** – date et heure des données enregistrées les plus anciennes de la piste;
- **DataTo** – date et heure des données enregistrées les plus récentes de la piste.

20.3.4 Énumération SearchState

L'état de la recherche peut être l'un des suivants

- **Queued** (En file d'attente) – signifie que la recherche n'a pas encore commencé;
- **Searching** (Recherche en cours) – signifie que la recherche est en cours et que de nouveaux résultats peuvent être générés;
- **Completed** (Terminée) – signifie que la recherche est terminée et qu'aucun nouveau résultat ne peut être généré.

20.3.5 Structure MediaAttributes

L'élément MediaAttributes contient des informations relatives aux pistes multimédia d'un enregistrement particulier pour un bloc de temps particulier. Le bloc de temps peut être un point dans le temps, auquel cas les éléments *From* et *Until* sont identiques.

- **RecordingToken** – référence à l'enregistrement que cette structure concerne;
- **From** – point dans le temps à partir duquel les attributs sont valides pour l'enregistrement;
- **Until** – point dans le temps auquel les attributs spécifiés cessent d'être valides pour l'enregistrement;

- VideoAttributes – ensemble d'attributs vidéo, décrivant les données d'une piste vidéo enregistrée;
- AudioAttributes – ensemble d'attributs audio, décrivant les données d'une piste audio enregistrée;
- MetadataAttributes – ensemble d'attributs, décrivant le contenu possible de métadonnées d'une piste de métadonnées enregistrée.

20.3.6 Structure FindEventResult

- RecordingToken – identifiant de l'enregistrement contenant l'événement trouvé;
- TrackToken – identifiant de la piste contenant l'événement trouvé;
- Time – date et heure de l'événement trouvé;
- Event – message d'événement trouvé;
- StartStateEvent – si la valeur est True, indique que l'événement représente l'état d'origine d'une ou de plusieurs propriétés de l'enregistrement.

20.3.7 Structure FindPTZPositionResult

- RecordingToken – identifiant de l'enregistrement contenant la position correspondante;
- TrackToken – identifiant de la piste contenant la position correspondante;
- Time – date et heure de la position correspondante;
- Position – vecteur PTZ correspondant.

20.3.8 Structure PTZPositionFilter

Contient les éléments nécessaires à la définition des positions PTZ à rechercher. Les vecteurs PTZ doivent se trouver dans le même espace de coordonnées que les coordonnées PTZ stockées dans l'enregistrement.

- MinPosition – limite inférieure du volume PTZ à rechercher;
- MaxPosition – limite supérieure du volume PTZ à rechercher;
- EnterOrExit – si la valeur est True, procéder à la recherche lors de l'entrée ou de la sortie du volume PTZ spécifié.

20.3.9 Structure MetadataFilter

Contient une expression XPath à appliquer à la structure MetadataStream.

Exemple d'expression de recherche d'objets chevauchant le quadrant inférieur droit de la scène:

```
boolean(//Object/Appearance/Shape/BoundingBox[@right > "0.5"])
and
boolean(//Object/Appearance/Shape/BoundingBox[@bottom > "0.5"])
```

20.3.10 Structure FindMetadataResult

- RecordingToken – identifiant de l'enregistrement contenant les métadonnées correspondantes;
- TrackToken – identifiant de la piste contenant les métadonnées correspondantes;
- Time – date et heure des métadonnées correspondantes.

20.4 GetRecordingSummary

GetRecordingSummary permet d'obtenir une description récapitulative de toutes les données enregistrées (voir Tableau 276). Cette opération est obligatoire pour la prise en charge d'un dispositif mettant en œuvre le service de recherche d'enregistrement.

Tableau 276 – Commande GetRecordingSummary

GetRecordingSummary		Demande-Réponse
Nom du message	Description	
GetRecordingSummaryRequest	Il doit s'agir du message vide	
GetRecordingSummaryResponse	Retourne une structure contenant: <i>DataFrom</i> spécifiant la première fois que les données ont été enregistrées sur le dispositif, <i>DataUntil</i> spécifiant la dernière fois que les données ont été enregistrées sur le dispositif, et le nombre total estimé d'enregistrements et de pistes. tt:RecordingSummary summary[1][1]	
Codes de défaut	Description	
	Pas de code d'erreur spécifique à la commande	

20.5 GetRecordingInformation

Retourne des informations relatives à un seul enregistrement spécifié par un *RecordingToken* (voir Tableau 277). Cette opération est obligatoire pour la prise en charge d'un dispositif mettant en œuvre le service de recherche d'enregistrement.

Tableau 277 – Commande GetRecordingInformation

GetRecordingInformation		Demande-Réponse
Nom du message	Description	
GetRecordingInformationRequest	Description de la demande tt:ReferenceToken RecordingToken [1][1]	
GetRecordingInformationResponse	Description de la réponse tt:RecordingInformation RecordingInformation [1][1]	
Codes de défaut	Description	
env:Sender ter: InvalidArgVal ter: InvalidToken	RecordingToken n'est pas valide.	

20.6 GetMediaAttributes

Retourne un ensemble d'attributs multimédia pour toutes les pistes des enregistrements spécifiés à un instant donné (voir Tableau 278). Les clients utilisant cette opération doivent pouvoir l'utiliser comme une opération non bloquante. Un dispositif doit attribuer des valeurs égales aux heures de début et de fin de la structure MediaAttributes si le calcul de cette plage provoque le blocage de cette opération. Voir la structure MediaAttributes pour plus