

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Communication networks and systems for power utility automation –
Part 8-2: Specific Communication Service Mapping (SCSM) – Mapping to
Extensible Messaging Presence Protocol (XMPP)**

**Réseaux et systèmes de communication pour l'automatisation des systèmes
électriques –
Partie 8-2: Mapping des services de communication spécifiques (SCSM) –
Mapping avec le protocole XMPP (Extensible Messaging Presence Protocol)**



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2018 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 l'IEC ou du Comité national de l'IEC du pays du demandeur. Si vous avez des questions sur le copyright de l'IEC ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de l'IEC de votre pays de résidence.

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

Tel.: +41 22 919 02 11
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.

IEC Catalogue - webstore.iec.ch/catalogue

The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - webstore.iec.ch/advsearchform

The advanced search enables 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 online and also once a month by email.

Electropedia - www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing 21 000 terms and definitions in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary

67 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC 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: sales@iec.ch.

A propos de l'IEC

La Commission Electrotechnique Internationale (IEC) 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 IEC

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

Catalogue IEC - webstore.iec.ch/catalogue

Application autonome pour consulter tous les renseignements bibliographiques sur les Normes internationales, Spécifications techniques, Rapports techniques et autres documents de l'IEC. Disponible pour PC, Mac OS, tablettes Android et iPad.

Recherche de publications IEC - webstore.iec.ch/advsearchform

La recherche avancée permet de trouver des publications IEC 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.

IEC Just Published - webstore.iec.ch/justpublished

Restez informé sur les nouvelles publications IEC. 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 de termes électroniques et électriques. Il contient 21 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans 16 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

Glossaire IEC - std.iec.ch/glossary

67 000 entrées terminologiques électrotechniques, en anglais et en français, extraites des articles Termes et Définitions des publications IEC parues depuis 2002. Plus certaines entrées antérieures extraites des publications des CE 37, 77, 86 et CISPR de l'IEC.

Service Clients - webstore.iec.ch/csc

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

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Communication networks and systems for power utility automation –
Part 8-2: Specific Communication Service Mapping (SCSM) – Mapping to
Extensible Messaging Presence Protocol (XMPP)**

**Réseaux et systèmes de communication pour l'automatisation des systèmes
électriques –
Partie 8-2: Mapping des services de communication spécifiques (SCSM) –
Mapping avec le protocole XMPP (Extensible Messaging Presence Protocol)**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

ICS 33.200

ISBN 978-2-8322-6158-3

**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.....	13
INTRODUCTION.....	15
1 Scope.....	16
1.1 General.....	16
1.2 Namespace name and version	16
1.3 Code Component distribution	17
2 Normative references	17
3 Terms and definitions	19
4 Abbreviated terms	21
5 Overview	22
5.1 General.....	22
5.2 Mapping of client/server services	23
5.2.1 General	23
5.2.2 XML payloads.....	24
5.2.3 Implementation agreements.....	25
5.2.4 XMPP	25
5.3 Time sync services	26
6 Usage of XMPP	26
6.1 Principles.....	26
6.2 Connection establishment.....	26
6.2.1 General	26
6.2.2 Usage of TLS and SASL	26
6.2.3 Stream Compression	27
6.3 Mapping of ACSI services.....	27
6.4 Usage of XMPP presence.....	28
6.5 Usage of the Roster	28
6.6 XMPP extensions.....	28
6.6.1 Usage of XMPP PING – XEP 0199	28
6.6.2 Usage of Stream Management – XEP 0198	28
6.7 Implementation agreements – XMPP PING – XEP 0199.....	28
7 End-to-end security	29
8 Payload description	29
8.1 XSD overview	29
8.2 Objects of IEC 61850.....	29
8.2.1 General	29
8.2.2 Logical Node (LN).....	30
8.2.3 Mapping of references of Logical Nodes to VariableAccessSpecifications	33
8.2.4 Mapping of DataObjects reference to VariableAccessSpecifications	34
8.2.5 Mapping of DataAttributes (DataAttr) reference to VariableAccessSpecifications	35
8.2.6 Usage of alternate access for DataObjects and DataAttributes references	35
8.3 Mapping of IEC 61850-7-2 data attributes	40
8.3.1 BasicTypes.....	40
8.3.2 Additional definitions of BasicType	41
8.3.3 Common ACSITypes	43

8.3.4	Mapping of quality common data attribute type specified in IEC 61850-7-2	54
8.4	General mapping of data values within XML payloads	55
8.5	Extended behaviour for optimization of bandwidth	56
9	Server class model	56
9.1	General	56
9.2	GetServerDirectory	57
9.2.1	General	57
10	Association model	60
10.1	Association relation to communication profiles	60
10.2	Two party association model for client/server communication profile	61
10.2.1	Establishment of a secured end-to-end association	61
10.2.2	Association services	61
11	Logical device model	66
11.1	General	66
11.2	Response-	68
11.3	Extended behaviour	69
12	Logical Node model	70
12.1	General	70
12.2	GetLogicalNodeDirectory	71
12.2.1	General	71
12.2.2	Response-	73
12.2.3	Extended behaviour	73
12.3	GetAllDataValues	76
12.3.1	General	76
13	DataObject, DataAttribute, SubDataAttribute model	79
13.1	General	79
13.2	GetDataValues	79
13.3	SetDataValues	81
13.4	GetDataDirectory	82
13.4.1	General	82
13.4.2	Response-	84
13.4.3	Extended behaviour	85
13.5	GetDataDefinition	87
14	Data set class model	87
14.1	General	87
14.2	GetDataSetValues	88
14.2.1	General	88
14.2.2	Response-	90
14.3	SetDataSetValues	91
14.3.1	General	91
14.3.2	Response-	92
14.4	CreateDataSet	93
14.4.1	General	93
14.4.2	Response-	95
14.5	DeleteDataSet	96
14.5.1	General	96
14.5.2	Errors	97
14.6	GetDataSetDirectory	98

14.6.1	General	98
14.6.2	Response-	100
15	ServiceTracking model	101
16	Setting group control class model	104
16.1	Setting group control block definition	104
16.2	Setting group control class services	104
16.2.1	SelectActiveSG	104
16.2.2	SelectEditSG	106
16.2.3	SetEditSGValue	106
16.2.4	ConfirmEditSGValues	107
16.2.5	GetEditSGValue	108
16.2.6	GetSGCBValues	109
17	Reporting and logging class model	110
17.1	Report model – Report control blocks	110
17.1.1	Buffered report control block	110
17.1.2	Unbuffered report control block	112
17.2	Reporting services	113
17.2.1	Report service	113
17.2.2	GetBRCBValues	116
17.2.3	SetBRCBValues	119
17.2.4	GetURCBValues	122
17.2.5	SetURCBValues	122
17.3	Log model	122
17.3.1	Overview	122
17.3.2	Description of LCB attributes	124
17.3.3	Mapping of log and log control services	124
17.3.4	Conformance	130
18	Mapping of the generic substation event model (GSE) – Generic object oriented substation event (GOOSE)	130
18.1	GOOSE control definition	130
18.2	Specialization for Layer 2 GoCB	131
18.3	Specialization for Routable GOOSE	131
18.4	GOOSE services	131
18.4.1	General	131
18.4.2	GetGoCBValues	131
18.4.3	SetGoCBValues	131
19	Transmission of sampled values class model	131
19.1	Sampled value control block	131
19.1.1	General	131
19.1.2	Specialization for Layer 2 Sampled value	131
19.1.3	Specialization for Routable Sampled value	132
19.1.4	Specialization for Unicast Sampled value	132
19.2	Sampled value services	132
19.2.1	General	132
19.2.2	GetMSVCBValues	132
19.2.3	SetMSVCBValues	132
20	Control class model	132
20.1	General	132

20.2	Overview of control services mapping	134
20.3	Select	134
20.4	SelectWithValue	136
20.4.1	SelectWithValue service parameter mapping	136
20.4.2	General mapping of the SelectWithValue service	136
20.4.3	SelectWithValue response–	139
20.5	Cancel	139
20.5.1	Cancel service parameter mapping	139
20.5.2	General mapping of the Cancel service	140
20.5.3	Cancel response–	141
20.6	Operate	142
20.6.1	Operate service parameter mapping	142
20.6.2	General mapping of the Operate service	142
20.6.3	Operate response–	144
20.6.4	CommandTermination service parameter mapping	144
20.6.5	General mapping of the CommandTermination service	144
20.7	TimeActivatedOperate	148
20.7.1	TimeActivatedOperate service parameter mapping	148
20.7.2	Mapping of the TimeActivatedOperate service	148
20.8	TimeActivatedOperateTermination service	148
20.9	AdditionalCauseDiagnosis in negative control service responses	149
20.10	Tracking of control services	152
20.10.1	General	152
20.10.2	Mapping of the Control service tracking (CTS)	152
21	Time and time synchronization model	153
22	Naming conventions	153
23	File transfer	153
23.1	File transfer model	153
23.2	File services	155
23.2.1	GetFile	155
23.2.2	SetFile	161
23.2.3	DeleteFile	163
23.2.4	GetFileAttributeValues	164
24	Conformance	166
24.1	Notation	166
24.2	PICS	166
24.2.1	Profile conformance	166
24.2.2	XML Payload conformance	168
24.3	PICS Statement	189
24.3.1	General	189
24.3.2	Substation configuration language	189
25	Substation Configuration Language (SCL)	189
Annex A (normative)	Communication stack	190
A.1	Overview	190
A.1.1	General	190
A.1.2	XMPP communication profiles	191
A.1.3	Non-XMPP communication profiles	191
A.2	Communication stack	192

A.2.1	Overview of the protocol usage.....	192
A.2.2	Client/server services and communication profiles.....	192
A.2.3	Time sync.....	194
Annex B	(informative) Deployment of XMPP infrastructure.....	196
B.1	General.....	196
B.2	Deployment of XMPP within one XMPP domain	196
B.2.1	Use case facility	196
B.2.2	Use case hierarchy within a facility	197
B.3	Deployment of XMPP while interconnecting more than one XMPP domain	198
B.3.1	Interconnection of XMPP Domain	198
B.3.2	Definition of a federation communication between XMPP domains.....	204
B.3.3	Interconnection of Domain with federation	205
B.4	Communication path outage and recovery.....	209
Annex C	(informative) Security for DER integration based on XMPP	210
C.1	General.....	210
C.2	Assumptions and boundary conditions	211
C.3	Derivation of security requirements.....	211
C.4	Mapping of security options to XMPP based integration of DER	212
C.5	Sequence diagrams	213
C.5.1	General	213
C.5.2	XMPP and Stream opening.....	213
C.5.3	Stream establishment, ROSTER and presence	214
C.5.4	Communication outage	215
C.5.5	Request Response (Clear Transfer).....	218
Annex D	(normative) Mapping of services and errors over XMPP stanzas.....	220
Annex E	(informative) Intentional deviations from IEC 61850-8-1 SCSM	223
Annex F	(informative) SCL conformance	224
Annex G	(normative) XML schema definitions for the XML payload	225
G.1	General.....	225
G.2	XML schema of the Virtual API for IEC 61850-8-2	225
G.3	XML schema of the applicative payload for IEC 61850-8-2.....	225
G.4	Extension of IEC 62351-4 when E2E security is turned off	252
Bibliography		253
Figure 1	– Overview of functionality and profiles	22
Figure 2	– Example of XML Payload	25
Figure 3	– Generic structure of client/server ACSI services.....	27
Figure 4	– Algorithm for logical node mapping	30
Figure 5	– Ordered list of functional constraints	31
Figure 6	– Example of Logical Node type description	32
Figure 7	– List of the flattened Named Variables corresponding to an LN.....	33
Figure 8	– XML mapping of a LNReference with direct access	34
Figure 9	– XML mapping of a LNReference with alternate access	34
Figure 10	– Direct XML mapping of a FCD	35
Figure 11	– Direct XML mapping of a FCDA.....	35
Figure 12	– Alternate access without array element	36

Figure 13 – Alternate access with array element	38
Figure 14 – Alternate access with flattened variable and array element	39
Figure 15 – XML structure of GetServerDirectory-Request (LD)	58
Figure 16 – XML structure of GetServerDirectory-Response (LD)	58
Figure 17 – XML structure of GetServerDirectory-Request (FILE)	60
Figure 18 – XML structure of GetServerDirectory-Response (FILE).....	60
Figure 19 – XML structure of Associate-Request.....	63
Figure 20 – XML structure of Associate-Response	63
Figure 21 – XML structure of GetLogicalDeviceDirectory-Request	68
Figure 22 – XML structure of GetLogicalDeviceDirectory-Response	68
Figure 23 – XML structure of extended GetLogicalDeviceDirectory-Request	70
Figure 24 – XML structure of extended GetLogicalDeviceDirectory-Response	70
Figure 25 – XML structure of GetLogicalNodeDirectory-Request.....	72
Figure 26 – XML structure of GetLogicalNodeDirectory-Response	73
Figure 27 – XML structure of extended GetLogicalNodeDirectory-Request (step 1).....	75
Figure 28 – XML structure of extended GetLogicalNodeDirectory-Response (step 1)	75
Figure 29 – XML structure of extended GetLogicalNodeDirectory-Request (step 2).....	75
Figure 30 – XML structure of extended GetLogicalNodeDirectory-Response (step 2)	76
Figure 31 – XML structure of GetAllDataValues-Request	78
Figure 32 – XML structure of GetAllDataValues-Response.....	79
Figure 33 – XML structure of GetDataValues-Request	80
Figure 34 – XML structure of GetDataValues-Response.....	81
Figure 35 – XML structure of SetDataValues-Request.....	82
Figure 36 – XML structure of SetDataValues-Response	82
Figure 37 – XML structure of GetDataDirectory-Request.....	83
Figure 38 – XML structure of GetDataDirectory-Response	84
Figure 39 – XML structure of extended GetDataDirectory-Request.....	86
Figure 40 – XML structure of extended GetDataDirectory-Response	87
Figure 41 – Mapping of reference to persistent data set within logical device.....	87
Figure 42 – Mapping of reference to persistent data set outside logical device	88
Figure 43 – Mapping of reference to non-persistent data set.....	88
Figure 44 – XML structure of GetDataSetValues-Request	89
Figure 45 – XML structure of GetDataSetValues-Response	90
Figure 46 – XML structure of SetDataSetValues-Request	92
Figure 47 – XML structure of SetDataSetValues-Response.....	92
Figure 48 – XML structure of CreateDataSet-Request.....	94
Figure 49 – XML structure of CreateDataSet-Response	95
Figure 50 – XML structure of DeleteDataSet-Request	97
Figure 51 – XML structure of DeleteDataSet-Response	97
Figure 52 – XML structure of GetDataSetDirectory-Request.....	99
Figure 53 – XML structure of GetDataSetDirectory-Response	100
Figure 54 – XML structure of SelectActiveSG-Request.....	105
Figure 55 – XML structure of SelectActiveSG-Response+	105

Figure 56 – XML structure of SelectActiveSG-Response-.....	105
Figure 57 – XML structure of SelectEditSG-Request	106
Figure 58 – XML structure of SetEditSGValue-Request.....	107
Figure 59 – XML structure of ConfirmEditSGValues	108
Figure 60 – XML structure of GetEditSGValue-Request	108
Figure 61 – XML structure of GetEditSGValue-Response.....	109
Figure 62 – XML structure of GetSGCBValues-Request.....	109
Figure 63 – XML structure of GetSGCBValues-Response	110
Figure 64 – XML structure of Report	115
Figure 65 – XML structure of GetBRCBValues-Request	117
Figure 66 – XML structure of GetBRCBValues-Response	119
Figure 67 – XML structure of SetBRCBValues-Request	121
Figure 68 – XML structure of SetBRCBValues-Response.....	122
Figure 69 – Relationship of LCB attributes to IEC 61850-7-2 log definitions	123
Figure 70 – XML structure of QueryLogByTime-Request.....	126
Figure 71 – XML structure of QueryLogByTime-Response	127
Figure 72 – XML structure of QueryLogAfter-Request	129
Figure 73 – XML structure of Select-Request.....	135
Figure 74 – XML structure of Select-Response	136
Figure 75 – XML structure of SelectWithValue-Request	138
Figure 76 – XML structure of SelectWithValue-Response+	139
Figure 77 – XML structure of SelectWithValue-Response-	139
Figure 78 – XML structure of CommandTermination Request+.....	146
Figure 79 – XML structure of CommandTermination Request-.....	148
Figure 80 – XML structure of InformationReport with AdditionalCauseDiagnosis	150
Figure 81 – Mapping of ACSI GetFile to FileOpen, FileRead, FileClose	156
Figure 82 – XML example of FileOpen Request	158
Figure 83 – XML example of FileOpen Response+.....	158
Figure 84 – XML example of FileRead Request (first)	158
Figure 85 – XML example of FileRead Response+ (first).....	159
Figure 86 – XML example of FileRead Request (second).....	159
Figure 87 – XML example of FileRead Response+ (second)	159
Figure 88 – XML example of FileClose Request.....	159
Figure 89 – XML example of FileClose Response	159
Figure 90 – Mapping of ACSI SetFile service	161
Figure 91 – XML example of ObtainFile Request	162
Figure 92 – XML example of ObtainFile Response	162
Figure 93 – XML example of DeleteFile Request.....	164
Figure 94 – XML example of DeleteFile Response	164
Figure 95 – XML example of GetFileAttributeValues Request	165
Figure 96 – XML example of GetFileAttributeValues Response	166
Figure 97 – VariableSpecification for LDevice/MHA11.HA.phsAHar(7).cVal.mag.f.....	178

Figure 98 – Shorter VariableSpecification for LDevice/MHAI1.HA.phsAHar(7).cVal.mag.f	179
Figure 99 – Non conformant VariableSpecification I	180
Figure 100 – Non conformant VariableSpecification II	181
Figure 101 – VariableSpecification for LDevice/MHAI1.HA.phsAHar(7) [MX]	182
Figure 102 – Shorter VariableSpecification for LDevice/MHAI1.HA.phsAHar(7) [MX]	183
Figure A.1 – Overview of functionality and profiles	191
Figure A.2 – OSI reference model and profiles	192
Figure B.1 – Facility domain	196
Figure B.2 – Hierarchical Aggregation at facility	197
Figure B.3 – DER Management System at facility	198
Figure B.4 – Facility Management integration at DSO	199
Figure B.5 – Multiple facilities at DSO	200
Figure B.6 – VPP and contracted DERs	201
Figure B.7 – indirect control using VPP JIDs	202
Figure B.8 – VPP direct control using VPP JIDs	202
Figure B.9 – VPP direct control using DSO JIDs	203
Figure B.10 – DSO Indirect control with VPP JIDs	203
Figure B.11 – DSO direct control	204
Figure B.12 – Concept of federation in XMPP	205
Figure B.13 – Federation DSO – Facility	206
Figure B.14 – Use of federation with VPP	206
Figure B.15 – communication with VPP JIDs – indirect control	207
Figure B.16 – VPP communication with VPP JIDs – direct control	207
Figure B.17 – VPP communication with DSO JIDs	208
Figure B.18 – DSO Communication with VPP JIDs	209
Figure C.1 – Base system for discussion of IT security requirements	210
Figure C.2 – XMPP Stream establishment – IEC 61850 Server to the XMPP Server	214
Figure C.3 – XMPP Stream establishment – IEC 61850 Client to the XMPP Server	214
Figure C.4 – Stream establishment, roster and presence	215
Figure C.5 – Communication outage – Loss of link	216
Figure C.6 – Communication outage – Presence unavailable	217
Figure C.7 – Request response	218
Figure C.8 – Request – Abort	219
Table 1 – Services requiring client/server Communication Profile	23
Table 2 – Mapping of ACSI classes on MMS concepts	30
Table 3 – Mapping of ACSI BasicTypes	40
Table 4 – PhyComAddr structure for Layer 2 communication	44
Table 5 – PhyComAddr for UPD/IP communication	45
Table 6 – GetNameList conflicting IEC 61850 objectClass and objectScope	47
Table 7 – Service error mappings for ACSI services using GetNameList	47
Table 8 – Read service error mappings	48

Table 9 – Write service error mappings	49
Table 10 – GetFileAttributeValues service error mappings	50
Table 11 – Encoding of IEC 61850-7-2 TimeQuality	51
Table 12 – Encoding of the TriggerConditions	52
Table 13 – Encoding of the ReasonForInclusionInReport	52
Table 14 – Encoding of the ReasonForInclusionInLog	53
Table 15 – Encoding of the RCBReportOptions	53
Table 16 – Encoding of the SVMMessageOptions	53
Table 17 – Encoding of the CheckConditions	54
Table 18 – Encoding of IEC 61850-7-2 quality	55
Table 19 – Examples of data values encoding	56
Table 20 – Mapping of ACSI GetServerDirectory (LOGICAL DEVICE)	57
Table 21 – Mapping of ACSI GetServerDirectory (FILE)	59
Table 22 – Association model versus communication profiles	60
Table 23 – Mapping of ACSI Associate service	62
Table 24 – Description of Associate request elements	62
Table 25 – Description of Associate response elements	63
Table 26 – Associate ACSI service error mappings	64
Table 27 – Mapping of ACSI Release service	66
Table 28 – Release service error mappings	66
Table 29 – Mapping of ACSI GetLogicalDeviceDirectory	67
Table 30 – Extended mapping of ACSI GetLogicalDeviceDirectory	69
Table 31 – Objectclasses for GetLogicalNodeDirectory service	70
Table 32 – Mapping of ACSI GetLogicalNodeDirectory	72
Table 33 – Extended mapping of ACSI GetLogicalNodeDirectory	74
Table 34 – Mapping of ACSI GetAllDataValues	77
Table 35 – Mapping of GetDataValues service parameters	80
Table 36 – Mapping of SetDataValues service parameters	81
Table 37 – Mapping of GetDataDirectory service parameters	83
Table 38 – GetDataDirectory service error mappings	85
Table 39 – Extended mapping of ACSI GetDataDirectory	86
Table 40 – Mapping of GetDataSetValues service parameters	88
Table 41 – GetDataSetValues error mappings	91
Table 42 – Mapping of SetDataSetValues service parameters	91
Table 43 – SetDataSetValues error mappings	93
Table 44 – Mapping of CreateDataSet service parameters	94
Table 45 – CreateDataSet service error mappings	96
Table 46 – Mapping of DeleteDataSet service parameters	96
Table 47 – DeleteDataSet service error mappings	98
Table 48 – Mapping of GetDataSetDirectory service parameters	99
Table 49 – GetDataSetDirectory service error mappings	101
Table 50 – Mapping of ACSI ServiceType values	101
Table 51 – Mapping of ACSI errorCode values	103

Table 52 – Mapping of CDC LTS.....	103
Table 53 – Mapping of CDC GTS.....	104
Table 54 – Mapping of SGCB.....	104
Table 55 – BRCB structure	111
Table 56 – URCB structure	112
Table 57 – Order of AccessResults for Report	113
Table 58 – Mapping of GetBRCBValues service parameters	116
Table 59 – Mapping of SetBRCBValues service parameters	120
Table 60 – LCB structure	123
Table 61 – Mapping of QueryLogByTime service parameters	125
Table 62 – ServiceError mappings for Log services	129
Table 63 – Mapping of QueryLogAfter-Request parameters	129
Table 64 – Log conformance requirements	130
Table 65 –TypeDescription definition for GoCB structure	130
Table 66 – Controllable service parameters	133
Table 67 – Mapping of IEC 61850-7-2 control model to control components.....	133
Table 68 – Mapping of control services	134
Table 69 – Mapping of Select parameters	135
Table 70 – SelectWithValue service parameter mapping.....	136
Table 71 – Mapping of SelectWithValue parameters	137
Table 72 – SelectWithValue, Oper and Cancel DataAccessError specification	139
Table 73 – Cancel service parameter mapping.....	140
Table 74 – Mapping of the Cancel service.....	141
Table 75 – Operate service parameter mapping.....	142
Table 76 – Mapping of the Operate service.....	143
Table 77 – Mapping of the CommandTermination service	145
Table 78 – Definition of LastAppError variable structure.....	149
Table 79 – Mapping of ACSI AddCause values	152
Table 80 – Mapping of CDC CTS	153
Table 81 – Mapping of ACSI file class to MMS file object.....	153
Table 82 – Reserved file suffixes	154
Table 83 – Mapping of ACSI GetFile service.....	157
Table 84 – GetFile service error mappings.....	158
Table 85 – Mappings of ACSI ServiceErrors to FileOpen Service Errors	160
Table 86 – Mappings of ACSI ServiceErrors to FileRead Service Errors	160
Table 87 – Mappings of ACSI ServiceErrors to FileClose Service Errors.....	161
Table 88 – Mapping of ACSI SetFile parameters	162
Table 89 – Mappings of ACSI ServiceErrors to ObtainFile Service Errors	163
Table 90 – Mapping of ACSI DeleteFile service	163
Table 91 – Mappings of ACSI ServiceErrors to DeleteFile Service Errors.....	164
Table 92 – Mapping of ACSI GetFileAttributeValues parameters.....	165
Table 93 – PICS for A-Profile support	167
Table 94 – PICS for Time Sync A-Profile support.....	167

Table 95 – PICS for T-Profile support	168
Table 96 – MMS InitiateRequest general parameters	168
Table 97 – MMS InitiateResponse general parameters	169
Table 98 – MMS service supported conformance table	169
Table 99 – MMS Parameter CBB	172
Table 100 – GetNameList conformance statement	173
Table 101 – GetCapabilityList conformance statement	173
Table 102 – GetDomainAttributes conformance statement	174
Table 103 – Status conformance statement	174
Table 104 – Cancel conformance statement	175
Table 105 – Identify conformance statement	175
Table 106 – AlternateAccess conformance statement	176
Table 107 – AlternateAccessSelection conformance statement	176
Table 108 – VariableAccessSpecification conformance statement	183
Table 109 – VariableSpecification conformance statement	184
Table 110 – Read conformance statement	184
Table 111 – Write conformance statement	184
Table 112 – InformationReport conformance statement	185
Table 113 – GetVariableAccessAttributes conformance statement	185
Table 114 – DefineNamedVariableList conformance statement	185
Table 115 – GetNamedVariableListAttributes conformance statement	186
Table 116 – DeleteNamedVariableList conformance statement	186
Table 117 – ReadJournal conformance statement	186
Table 118 – EntryContent conformance statement	187
Table 119 – FileDirectory conformance statement	188
Table 120 – FileOpen conformance statement	188
Table 121 – FileRead conformance statement	188
Table 122 – FileClose conformance statement	189
Table 123 – Allowed P-Type definitions for client/server addressing	189
Table A.1 – Service and protocols for client/server communication A-Profile	193
Table A.2 – Service and protocols for client/server XMPP T-Profile	193
Table A.3 – Time sync A-Profile	194
Table A.4 – Time sync T-Profile	195
Table D.1 – ACSI services mapping over XMPP stanzas	220
Table D.2 – Error mapping over XMPP stanzas	222
Table F.1 – SCL conformance degrees	224
Table F.2 – Supported ACSI services for SCL.2	224

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**COMMUNICATION NETWORKS AND
SYSTEMS FOR POWER UTILITY AUTOMATION –****Part 8-2: Specific Communication Service Mapping (SCSM) –
Mapping to Extensible Messaging Presence Protocol (XMPP)**

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 61850-8-2 has been prepared by IEC technical committee 57: Power systems management and associated information exchange.

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

FDIS	Report on voting
57/2020/FDIS	57/2039/RVD

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

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

This IEC standard includes Code Components i.e components that are intended to be directly processed by a computer. Such content is any text found between the markers <CODE BEGINS> and <CODE ENDS>, or otherwise is clearly labeled in this standard as a Code Component.

The purchase of this IEC standard carries a copyright license for the purchaser to sell software containing Code Components from this standard to end users either directly or via distributors, subject to IEC software licensing conditions, which can be found at: www.iec.ch/CCv1.

A list of all parts in the IEC 61850 series, published under the general title *Communication networks and systems for power utility automation*, can be found on the IEC website.

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

This part of IEC 61850 is part of a set of specifications which details layered utility communication architecture.

The usage of the IEC 61850 communication standard is largely spreading over all the domains connected to the smart grid, pushing the usage of technologies adapted to the connection of a very large number of applications and devices across the intra/inter-net. The involved domains use already well established protocols typically for exchanging data with IT level applications like resource planning, asset and maintenance management, etc. Therefore, it becomes imperative to provide an integration strategy that allows the integration of IEC 61850 into these various disparate protocols and information.

In this context, this part of IEC 61850 describes a specific communication service mapping (SCSM) over the Extensible Messaging and Presence Protocol (XMPP), providing detailed information on how to create and exchange concrete communication messages that implement abstract services and models specified in IEC 61850-7-4, IEC 61850-7-3, and IEC 61850-7-2.

This mapping is intended to be utilized between all kinds of utility Distributed Energy Resource devices and their related management systems, in particular over public networks.

NOTE This part of IEC 61850 does not provide tutorial material. For this purpose, IEC 61850-5 and IEC 61850-7-1 can be read in conjunction with IEC 61850-7-2.

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

COMMUNICATION NETWORKS AND SYSTEMS FOR POWER UTILITY AUTOMATION –

Part 8-2: Specific Communication Service Mapping (SCSM) – Mapping to Extensible Messaging Presence Protocol (XMPP)

1 Scope

1.1 General

This part of IEC 61850 specifies a method of exchanging data through any kinds of network, including public networks. Among the various kinds of services specified in IEC 61850-7-2, only the client/server and time synchronization services are considered so far.

NOTE Client/server services of GOOSE and SMV models are mapped as well (see Table 1).

For the client/server services, the principle is to map the objects and services of the ACSI (Abstract Communication Service Interface defined in IEC 61850-7-2) to XML messages transported over XMPP. The mapping description includes mainly three aspects:

- The usage of the XMPP protocol itself, describing in details which features are really used and how they are used by the mapping (see Clause 6).
- How to achieve end-to-end secured communications (see Clause 7).
- The description of the XML payloads corresponding to each ACSI service thanks in particular to the XML Schema and XML message examples (starting at Clause 9).

NOTE 1 This document does not address the detailed usage of the XMPP protocol.

NOTE 2 This document does not address system management services.

NOTE 3 For the information of people familiar with the mapping defined in IEC 61850-8-1, the XML messages defined in the present document are derived from those defined in IEC 61850-8-1 but with an XML encoding instead of a binary one. In this way implementing gateways between IEC 61850-8-1 and IEC 61850-8-2 is very straightforward in both directions. However reading IEC 61850-8-1 is not necessary to understand the present document except when it is used in conjunction with one of the GOOSE mappings described in IEC 61850-8-1.

1.2 Namespace name and version

This new section is mandatory for any IEC 61850 namespace (as defined by IEC 61850-7-1).

The parameters which identify this release of the SCSM_8_2 namespace `xmlns="http://www.iec.ch/61850/2018/SCSM_8_2"` are:

- Namespace Version: 2018
- Namespace Revision: A
- Namespace Release: 1
- Namespace release date: 2018-12

Edition	Publication date	Webstore	Namespace
Edition 1.0	2018-12	IEC 61850-8-2:2018	IEC 61850-8-2:2018

1.3 Code Component distribution

The Code Components included in this IEC standard are also available as electronic machine readable file at:

http://www.iec.ch/tc57/supportdocuments/IEC_61850-8-2.2018_ed1.0.XSD.2018A1.full.zip

The Code Component(s) included in this IEC standard are potentially subject to maintenance works and users shall select the latest release in the repository located at: <https://www.iec.ch/tc57/supportdocuments>.

The latest version/release of the document will be found by selecting the file IEC 61850-8-2.2018_ed1.0.XSD.{VersionStateInfo}.full.zip with the filed VersionStateInfo of the highest value.

In case of any differences between the downloadable code mentioned above and the IEC pdf published content, the downloadable code(s) is(are) the valid one; it may be subject to updates. See history files.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61850 (all parts), *Communication networks and systems for power utility automation*

IEC TS 61850-2, *Communication networks and systems in substations – Part 2: Glossary*

IEC 61850-5, *Communication networks and systems for power utility automation – Part 5: Communication requirements for functions and device models*

IEC 61850-6, *Communication networks and systems for power utility automation – Part 6: Configuration description language for communication in electrical substations related to IEDs*

IEC 61850-7-1, *Communication networks and systems for power utility automation – Part 7-1: Basic communication structure – Principles and models*

IEC 61850-7-2:2010, *Communication networks and systems for power utility automation – Part 7-2: Basic information and communication structure – Abstract communication service interface (ACSI)*

IEC 61850-7-2:2010/AMD1:2018

IEC 61850-7-3, *Communication networks and systems for power utility automation – Part 7-3: Basic communication structure – Common data classes*

IEC 61850-7-4, *Communication networks and systems for power utility automation – Part 7-4: Basic communication structure – Compatible logical node classes and data object classes*

IEC 61850-8-1:2011, *Communication networks and systems for power utility automation – Part 8-1: Specific communication service mapping (SCSM) – Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3*

IEC 61850-8-1:2011/AMD1:2018

IEC 62351 (all parts), *Power systems management and associated information exchange – Data and communications security*

IEC 62351-4:2018, *Power systems management and associated information exchange – Data and communications security – Part 4: Profiles including MMS and derivatives*¹

IEC 62351-6, *Power systems management and associated information exchange – Data and communications security – Part 6: Security for IEC 61850*²

IEC TR 62357-200, *Power systems management and associated information exchange – Part 200: Guidelines for migration from Internet Protocol version 4 (IPv4) to Internet Protocol version 6 (IPv6)*

ISO/IEC 7498-1:1994, *Information technology – Open systems interconnection – Basic reference model: The basic model*

ISO/IEC 8824-1:2015 [ITU-T X.680:2015], *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*

ISO/IEC 8825-4:2015 [ITU-T X.693:2015], *Information technology – ASN.1 encoding rules: XML Encoding Rules (XER)*

ISO 9506 (all parts), *Industrial automation systems – Manufacturing message specification*

ISO 9506-1:2003, *Industrial automation systems – Manufacturing message specification – Part 1: Service definition*

ISO 9506-2:2003, *Industrial automation systems – Manufacturing message specification – Part 2: Protocol specification*

IEEE C37.111:1999, *IEEE standard common format for transient data exchange (COMTRADE) for power systems*

IEEE 754:1985, *IEEE Standard for binary floating-point arithmetic*

RFC 768, *User datagram protocol*, IETF, available at <http://www.ietf.org>

RFC 791, *Internet protocol – DARPA Internet Program – Protocol specification*, IETF, available at <http://www.ietf.org>

RFC 792, *Internet control message protocol – DARPA internet program – Protocol specification*, IETF, available at <http://www.ietf.org>

RFC 793, *Transmission control protocol – DARPA internet program – Protocol specification*, IETF, available at <http://www.ietf.org>

RFC 826, *An Ethernet address resolution protocol or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware*, IETF, available at <http://www.ietf.org>

RFC 919, *Broadcasting internet datagrams*, IETF, available at <http://www.ietf.org>

¹ Under preparation. Stage at the time of publication: IEC/DECFDIS 62351-4:2018.

² At the time of publication, a new edition was under preparation.

RFC 922, *Broadcasting internet datagrams in the presence of subnets*, IETF, available at <http://www.ietf.org>

RFC 950, *Internet standard subnetting procedure*, IETF, available at <http://www.ietf.org>

RFC 1112, *Host extensions for IP multicasting*, IETF, available at <http://www.ietf.org>

RFC 2460, *Internet protocol, Version 6 (IPv6) specification*, IETF, available at <http://www.ietf.org>

RFC 3629, *UTF-8, a transformation format of ISO 1646 – IETF*, available at <http://www.ietf.org>

RFC 4422, *Simple authentication and security layer (SASL)*, available at <http://www.ietf.org>

RFC 5246:2008, *The TLS protocol version 1.2*, available at <http://www.ietf.org>

RFC 5905, *Network time protocol (NTP) Version 4: Protocol and algorithms specification*, IETF, available at <http://www.ietf.org>

RFC 6120, *Extensible messaging and presence protocol (XMPP): Core*, available at <http://www.ietf.org>

RFC 6121, *Extensible messaging and presence protocol (XMPP): Instant messaging and presence*, available at <http://www.ietf.org>

RFC 6122, *Extensible messaging and presence protocol (XMPP): Address format*, available at <http://www.ietf.org>

XEP-0198, *Stream management*

NOTE This specification defines an XMPP protocol extension for active management of an XML stream between two XMPP entities, including features for stanza acknowledgements and stream resumption.

XEP-0199, *XMPP Ping*

NOTE This specification defines an XMPP protocol extension for sending application-level pings over XML streams. Such pings can be sent from a client to a server, from one server to another, or end-to-end.

3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC TS 61850-2 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

3.1

class

description of a set of objects that share the same attributes, services, relationships, and semantics

[SOURCE: IEC 61850-7-2:2010, 3.1]

3.2

client

entity that requests a service from a server and that receives unsolicited messages from a server

[SOURCE: IEC 61850-7-2:2010, 3.2]

3.3

device

entity that performs control, actuating and/or sensing functions and interfaces to other such entities within an automation system

Note 1 to entry: Devices alone do not perform energy transport functions.

[SOURCE: IEC 61850-7-2:2010, 3.3]

3.4

logical device

entity that represents a set of typical substation functions

[SOURCE: IEC 61850-7-2:2010, 3.6]

3.5

logical node

entity that represents a typical substation function

[SOURCE: IEC 61850-7-2:2010, 3.7]

3.6

physical device

entity that represents the physical parts of a device (hardware and operating system, etc.)

Note 1 to entry: Physical devices host logical devices.

[SOURCE: IEC 61850-7-2:2010, 3.8]

3.7

application and transport profiles

A-Profile and T-Profile

set of protocols for a specific purpose

3.8

data attribute

named element of data object which has a specific type

3.9

data object

meaningful structured information of applications located in an automation device which can be read or written

Note 1 to entry: This definition is in accordance with the DataObject definition of IEC 61850-7-2. The MMS use of DataObject refers to data types and potential instantiations of MMS variables.

3.10

server

entity that provides services to clients or issues unsolicited messages

3.11**private network**

network used by a unique entity which masters all the data flows and whose performance is guaranteed in terms of bandwidth, throughput, transmission delay, availability, etc.

Note 1 to entry: If these conditions are not fulfilled, the network is considered as public.

Note 2 to entry: A private network may be based on a public or shared infrastructure, as soon as the level of services can be guaranteed.

3.12**smart grid**

application of computer intelligence and networking abilities to the electricity delivery system so it monitors, protects and automatically optimizes the operation of its interconnected elements, from the central and distributed generator through the high-voltage network and distribution system, to industrial users and building automation systems, to energy storage installations and to end-use consumers and their thermostats, electric vehicles, appliances and other household devices

4 Abbreviated terms

ACSI	Abstract Communication Service Interface
BRCB	Buffered Report Control Block
CDC	Common Data class
DER	Distributed Energy Resource
DNS	Domain Name System
DSO	Distribution System Operator
E2E	End-to-end
FC	Functional Constraint
FCD	Functionally Constrained Data
IED	Intelligent Electronic Device
IETF	Internet Engineering Task Force
IP	Internet Protocol
JID	Jabber Identifier
LAN	Local Area Network
LCB	Log Control Block
LD	Logical Device
LN	Logical Node
MMS	Manufacturing Message Specification (ISO 9506)
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation Extra Information
RBAC	Role-Based Access Control
RFC	Request for Comments
SBO	Select Before Operate
SCL	substation automation System Configuration Language (IEC 61850-6)
SCSM	Specific Communication Service Mapping
SCSM 8-1	Specific Communication Service Mapping as defined by IEC 61850-8-1
SCSM 8-2	Specific Communication Service Mapping as defined by IEC 61850-8-2
SG	Setting Group

SNTP	Simple Network Time Protocol
SV	Sampled Values
TCP	Transmission Control Protocol
UCA	Utility Communication Architecture
URCB	Unbuffered Report Control Block
UTC	Coordinated universal time
VMD	Virtual Manufacturing Device
VPP	Virtual Power Plant
WAN	Wide Area Network
XER	XML Encoding Rules
XMPP	Extensible Messaging Presence Protocol
XSD	XML Schema Definition

5 Overview

5.1 General

The purpose of this document is to provide detailed instructions/specifications as to the mechanisms and rules required to implement the core ACSI services specified in IEC 61850-7-2, while making use of XMPP, as well as the usage of SNTP for time synchronization (see Figure 1).

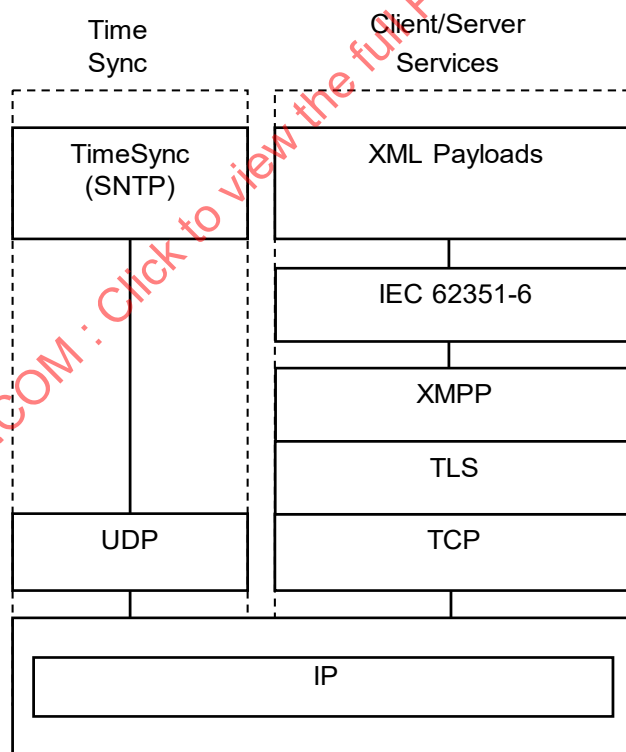


Figure 1 – Overview of functionality and profiles

Further details pertaining to the use and actual specification of each profile may be found in Clause 6.

NOTE The stack profiles described here start at the IP layer. Which specifications must be used for the layers below IP is out of the scope of this SCSM.

5.2 Mapping of client/server services

5.2.1 General

The client/server communication profile shall be used for any implementation claiming conformance to this document and declaring support for one of the IEC 61850-7-2 services shown in Table 1.

Table 1 – Services requiring client/server Communication Profile

IEC 61850-7-2 model	IEC 61850-7-2 service
Server	GetServerDirectory
Association	Associate
	Abort
	Release
Logical Device	GetLogicalDeviceDirectory
Logical Node	GetLogicalNodeDirectory
	GetAllDataValues
Data	GetDataValues
	SetDataValues
	GetDataDirectory
	GetDataDefinition
Data Set	GetDataSetValues
	SetDataSetValues
	CreateDataSet
	DeleteDataSet
	GetDataSetDirectory
Setting Group Control Block	SelectActiveSG
	SelectEditSG
	SetEditSGValue
	ConfirmEditSGValues
	GetEditSGValue
	GetSGCBValues
Report Control Block	Report
	GetBRCBValues
	SetBRCBValues
	GetURCBValues
	SetURCBValues
LOG Control Block	GetLCBValues
	SetLCBValues
	GetLogStatusValues
	QueryLogByTime
	QueryLogAfter
GOOSE ^a	GetGoCBValues
	SetGoCBValues
Multicast Sampled Values ^a	GetMSVCBValues
	SetMSVCBValues
Control	Select
	SelectWithValue
	Cancel
	Operate
	CommandTermination
	TimeActivatedOperate
FILE transfer	GetFile
	SetFile
	DeleteFile
	GetFileAttributeValues
^a The services SendGOOSEMessage, GetGoReference, GetGOOSEElementNumber, SendMSVMessage, GetMsvReference and GetMSVElementNumber are not mapped within the present document because an XMPP mapping would not fulfill the required performances.	

The request and response messages of the client/server ACSI services shall be represented by XML messages transported between the caller application and the called application over the XMPP protocol. Per the XMPP protocol, an XML stream has to be first established between the two applications and an XMPP server. Then the XML messages representing the ACSI services shall be embedded within XML constructs defined by XMPP and called stanzas. Each stanza shall be sent by the caller within its XMPP stream, then routed by the XMPP server and finally delivered to the called application through its own XMPP stream.

NOTE See A.2.2 for a formal description of the communication layers involved in this profile.

5.2.2 XML payloads

When building the XML messages transported over XMPP, implementations shall be aware that each XML message is made up of three separate and nested parts (see Figure 2):

- The service PDU representing the ACSI service itself is the deeper nested level. It is represented in the XML messages by an element such as <confirmed-RequestPDU> or <confirmed-ResponsePDU> for respectively a service request and a service response. A major goal of the present document is to describe how to build these applicative payloads. Their structure shall be formally defined by the XML schema provided in Annex G.3.
- The service PDU is then wrapped within end-to-end security elements, holding security related information such as certificates, session keys and other parameters. This information is used to establish and manage an end-to-end secured channel between the caller and called applications.
- The association context element is the direct child element of the XMPP stanza, wrapping the service PDU and the security elements. It transports an identifier used to create and manage the association between the applications. Figure 2 presents an example of XML payload corresponding to an ACSI GetDataValues request, assuming that encryption of the applicative payload is not activated.

The two latter parts mentioned above are specified in IEC 62351-4.

<i><Element(s) for management of association context></i>	Association context
<i><Elements for management of end-to-end security></i>	Security elements
<pre> < > [...]</pre>	
<pre> <confirmed-RequestPDU> <invokeID>2676121</invokeID> <ConfirmedServiceRequest> <read specificationWithResult="false" > <variableAccessSpecification> <listOfVariable> <SEQUENCE> <variableSpecification> <name> <domain-specific> <domainId>IEDNameLDINst</domainId> <itemId>LLN0\$ST\$Beh\$stVal</itemId> </domain-specific> </name> </variableSpecification> </SEQUENCE> </listOfVariable> </variableAccessSpecification> </read> </ConfirmedServiceRequest> </confirmed-RequestPDU></pre>	Service PDU (applicative payload)
<pre> </ > [...]</pre>	
<i></Elements for management of end-to-end security></i>	Security elements
<i></Element(s) for management of association context ></i>	Association context

Figure 2 – Example of XML Payload

5.2.3 Implementation agreements

Support of the end-to-end security mechanism mentioned above is mandatory for any implementation conforming to the present document. However, for a given deployment the end-to-end security feature may be turned off. In that case the behavior of both client and server applications shall be as follows:

- Turning-off the end-to-end security mechanism is a dedicated configuration of both IEC 61850 applications, client and server. This configuration is out of the scope of the present document.
- When the two communicating applications are properly configured for operating without end-to-end security, the exchanged payloads shall not include any security elements and the service PDU shall be directly wrapped within the association context part (see G.4).
- The usage of the XMPP stanzas and the management of the association context shall remain identical to the case when end-to-end security is applied and shall be therefore as described in IEC 62351-4.
- In case the client and server entities are configured differently, one using the end-to-end security and the other not, no interaction is possible. Any request for initiating a communication shall fail.

5.2.4 XMPP

XMPP is a protocol defined at the Internet Engineering Task Force (IETF) by RFC 6120, RFC 6121 and RFC 6122. Beside the core specifications, the XMPP Standards Foundation develops extensions called XEPs.

For the present SCSM the role of XMPP is to transport the XML payloads between two IEC 61850 applications

Clause 6 gives details about how the present mapping uses XMPP including both the core specifications and the possible XEPs.

5.3 Time sync services

The time synchronization services specified in IEC 61850-7-2 shall be mapped on the IETF Simple Network Time Protocol (SNTP).

NOTE 1 Using SNTP on public networks may require particular engineering techniques for ensuring appropriate cyber-security (e.g. usage of redundant clocks, discovery of trusted clocks). Defining or recommending such techniques goes beyond the scope of this document.

NOTE 2 See A.2.3 for a formal description of the communication layers involved in this profile.

6 Usage of XMPP

6.1 Principles

XMPP is based on a distributed architecture of XMPP servers. From an XMPP point of view, an application complying with the present SCSM shall act as an XMPP client connected to one or several XMPP servers.

NOTE The XMPP servers can be operated either by one of the actors involved in the IEC 61850 interactions at applicative level or by a third party provider.

Each IED shall have an account on the XMPP server to which it is connected and the resource part of the JID used to identify a given connection shall be fixed. In order that the same connection can be used by any IEC protocol based on XMPP, the generic value IEC shall be used.

Example of a full JID: myIED@DSO.org/IEC.

6.2 Connection establishment

6.2.1 General

For establishing a connection to the XMPP server, an IED shall perform the following steps:

- Initiate a TCP connection to the server
- Open an XML stream
- Negotiate stream features
- Authenticate with the XMPP server
- Select a resource
- Request the roster and send initial presence

These different steps are specified in details by XMPP (RFC 6120). Additionally the rules specified in Subclauses 6.2.2 to 6.2.3 shall apply.

6.2.2 Usage of TLS and SASL

TLS shall be used in any case and shall be implemented as specified in IEC 62351-6. However several cases shall be considered:

1. XMPP over TLS where the TLS negotiation is part of the XMPP Stream negotiation;
2. XMPP over TLS where the TLS session is a tunnel over which the XMPP Stream is established;

3. XMPP over TLS where the TLS negotiation is part of the XMPP Stream negotiation; however within an existing TLS tunnel over which the XMPP Stream is established (combination of 1 and 2).

SASL shall be used in any case for authentication of the XMPP endpoints. However several cases shall be considered, depending on how TLS is used:

- SASL EXTERNAL mechanism (where authentication is given by the certificate exchanged during the <stream:feature/> <starttls/> negotiation) may be achieved for 1 and 3, as far as the TLS certificate used during the negotiation includes the JID identity.
- SASL PLAIN mechanism (a simple clear text password mechanism) may also be deployed for 1 and 3, and it becomes mandatory as soon as the TLS certificate used during the negotiation does not include the JID identity.
- SASL PLAIN mechanism shall be used as the only option for 2 as the certificate used in this case by TLS would not be available at the XMPP component level.

One of the proposed option for both TLS and SASL shall be selected and deployed, depending on the transport infrastructure, local regulation and/or local policy.

6.2.3 Stream Compression

XMPP Stream Compression may be used (see XEP 0138), but note that it will be not efficient for stanzas involved in an end-to-end security session with encryption.

6.3 Mapping of ACSI services

Most of the client/server services are solicited services defined by three elements: the Request, the positive Response and the negative Response (see Figure 3).

Parameter name
Request
<request parameters>
Response+
<response parameters>
Response-
<service errors>

Figure 3 – Generic structure of client/server ACSI services

Solicited services shall be implemented by <iq> stanzas. The Request message shall be implemented by an <iq> stanza with the attribute “type” equal to “set” or “get” and the Response+ message shall be implemented by an <iq> stanza with attribute “type” set to “result”. Response- messages shall be transmitted as an <iq/> stanza with type=“result” or as a <message/> with type=“normal”.

Unsolicited services shall be implemented as <message/> stanzas of type “normal”.

NOTE The payloads representing the ACSI services, i.e. the Service PDUs specified by the present document, are embedded within elements used for management of end-to-end security and association context (see Figure 2). The two latter kinds of elements are specified by IEC 62351-4. The errors encountered during the management of these elements is specified in IEC 62351-4 as well.

Errors occurring at XMPP level shall use the error mechanism defined by XMPP.

Table D.1 defines for each ACSI service which kind of stanza and which value of the “type” attribute shall be used. Note that the mapping of the Associate service (last three rows of the table) is rather described in IEC 62351-4.

The various error cases (Response- and errors raised at XMPP level) are described in 8.3.3.4 and Table D.2.

6.4 Usage of XMPP presence

Two-party associations shall be monitored by using the presence information of the IEC 61850 entities (i.e. the IEC 61850 client and the IEC 61850 server).

During the Associate request, the IEC 61850 client shall subscribe to the presence of the IEC 61850 server (if not already done before). The server may also subscribe to the client presence. An entity detecting the unavailable presence status of its associated peer shall generate a local Abort.

The establishment of TCP connections and XMPP streams between an IEC 61850 communicating entity and an XMPP server is not dedicated to one association and therefore not part of it. Both shall exist before the Associate service is called.

6.5 Usage of the Roster

As recommended in RFC 6121, an XMPP client (and then an IEC 61850 client or server) shall request its Roster at the end of the stream creation.

Then, for security consideration, the XMPP client shall check the consistency of its Roster and eventually update it accordingly.

6.6 XMPP extensions

6.6.1 Usage of XMPP PING – XEP 0199

XMPP servers may use various methods for checking the presence of the connected clients, typically TCP keepalive which detects only broken TCP connections or XMPP PING (XEP 0199), which detects problems both on TCP and XMPP levels. As a minimum requirement an XMPP client shall be able to answer to XMPP PING stanzas as defined in XEP 0199.

NOTE Other usages of this XEP are outside the scope of this SCSM, for example entities may also send XMPP pings for checking the availability of a peer, either IEC 61850 entity or XMPP server.

6.6.2 Usage of Stream Management – XEP 0198

When they want to improve the network reliability, XMPP entities may enable the stream management in conformance with the XEP 0198. The term XMPP entity covers here not only the XMPP clients but also the XMPP servers.

An implementation declaring support for XEP 0198 shall manage the acknowledgements as defined by this XEP, for checking that the number of stanzas sent by one entity corresponds to the number of stanzas handled by the receiving entity.

It is recommended that an entity resume the stream and resend the non handled stanzas when it detects that these two numbers are different, or when the XMPP stream has been broken. For a minimum functional level, the resumption may be replaced by a comprehensive error raised to the upper stack layers.

NOTE The strategy for exchanging the acks (periodically, every stanza or every N stanza), depends on the scenarios.

As resumption scenarios can imply the resending of already handled stanzas, all entities (even those not supporting the XEP 0198) shall check the stanza "id" attribute in order to ignore the duplicates.

6.7 Implementation agreements – XMPP PING – XEP 0199

The delay between two XMPP PING depends on the XMPP server configuration. The recommended default value is 60 s.

7 End-to-end security

The end-to-end security mechanism used in the present SCSM is specified by IEC 62351-4. IEC 62351-4 applies to both the present SCSM and IEC 61850-8-1 and gives the principles and the abstract definitions of the security elements exchanged between the caller and the called endpoints.

The end-to-end security session is created together with the creation of the ACSI association during the initial exchange of XMPP messages corresponding to the ACSI Associate. Within these messages, the payload corresponding to the ACSI Associate service is completed by security elements required for establishing the end-to-end session: tokens holding the necessary cryptographic parameters like for example digital certificates, as well as signatures.

During this handshake, the mutual authentication of the two endpoints is performed based on the exchanged digital certificates and signatures. At the same time a Diffie Hellman key exchange takes place, providing the session keys necessary to secure the subsequent data exchanges by using symmetric cryptographic algorithms for integrity protection and data confidentiality during data transfer.

The details of the mechanism including in particular the available options (e.g. only signing or also encrypting the payload) are defined in IEC 62351-4.

8 Payload description

8.1 XSD overview

The XML payloads transported over XMPP consist of several parts which are not all defined within the present document. The first levels of child elements within the XMPP stanzas are in fact defined in IEC 62351-4 and are mainly related to the management of the end-to-end security. The present clause describes only the applicative part of the payload which is formally defined by the XML schema presented in annex G.3.

The MMSpdu complexType is the entry point within the XSD. It proposes a choice of different envelops for wrapping the various ACSI services. The most commonly used envelops are <confirmed-RequestPDU/> and <confirmed-ResponsePDU/> for mapping ACSI confirmed services and <unconfirmed-PDU/> for mapping ACSI unconfirmed services. Within the envelops for confirmed ACSI services, the complexType ConfirmedServiceRequest (and also ConfirmedServiceResponse) proposes a choice of different service elements (e.g. <read/>, <write/>, <getNameList/>). The detail of the available envelops and service elements is described in Table D.1. How to map the different ACSI services to these envelops and service elements is described in the detailed mapping of each ACSI service starting at Clause 9.

Note that the end-to-end security mechanism may imply to encrypt the applicative payload. In that case the element defined by IEC 62351-4 for containing the applicative payload, would not be of type MMSpdu. It would contain instead the encrypted form of the equivalent value of type MMSpdu which would be used for mapping the same ACSI service when no encryption is used.

8.2 Objects of IEC 61850

8.2.1 General

For mapping the objects defined in IEC 61850-7-2, the present SCSM uses some concepts already used by IEC 61850-8-1 and inherited from the protocol MMS. Their names appear in the XML payloads but it is not necessary to understand these concepts in details. In general it will be enough to know how the ACSI class are mapped on these concepts (see Table 2).

Table 2 – Mapping of ACSI classes on MMS concepts

ACSI Classes	MMS Concepts
Server	VMD
Logical Device	Domain
Logical Node	NamedVariable
Data Object	NamedVariable
All Control Block classes	NamedVariable
DataSet	NamedVariableList
Log	Journal
Files	Files

8.2.2 Logical Node (LN)

The construction of the content of a Logical Node is defined in IEC 61850-7-2, IEC 61850-7-3 and IEC 61850-7-4.

This SCSM maps each Logical Node instance to a Named Variable (concept inherited from MMS), whose name is the name of the Logical Node instance and which have a hierarchical complex structure consisting in multiples levels of components.

The type description of this structure can be created by the algorithm in Figure 4.

```

For each functional constraint (c) found in Figure 53

    For each DataObject (d) in the LN

        If the FCD generated from d,c is not empty, then add a component of ComponentName (d) of the
        ComponentType determined by the FCD to structure (s).

    End For

If the structure (s) is not empty then add a component of ComponentName (c) of the
ComponentType (s) to the TypeDescription being generated.

End For

```

Figure 4 – Algorithm for logical node mapping

The order of the resulted components is recommended to be as shown in Figure 5.

³ The recommended ordering provides compatibility with existing UCA 2.0 implementations.

MX
ST
CO
CF
DC
SP
SG
RP
LG
BR
GO
GS (deprecated)
SV
SE
MS
US (deprecated)
EX
SR
OR
BL
RG
RS

Figure 5 – Ordered list of functional constraints

NOTE 1 FCs are defined as follows: RP – unbuffered report control blocks; LG – log control blocks; BR – buffered report control blocks; GO – GOOSE control blocks; GS – deprecated GSSE control blocks; SV – substituted values; SE – Setting group editing; MS – Multicast Sampled Values control block (from IEC 61850-9-2); US – deprecated Unicast Sampled Values control block; EX – name space for model extension; SR – service tracking; CO – Control service parameters; OR – order received; BL for blocking; RG for Routable GOOSE control blocks; and RS for Routable Multicast Sampled Values control blocks.

NOTE 2 The FC “XX” defined in IEC 61850-7-2 as the wildcard FC for services only is not mapped to the MMS SC SM. Therefore any request that uses FC=“XX” will be acknowledged negatively.

If there is no DataObject (d) that instantiates a specific functional constraint, that functional constraint shall not appear as part of the NamedVariable TypeDescription.

Within a component representing a functional constraint, the order of the sub-components is determined by the order of the names of the instantiated compatible data classes as SCL DOs (see IEC 61850-6) whose data attributes have the specific functional constraint.

Within a component representing IEC 61850-7-4 DataObjects, the order of the sub-components is determined by the order of the names of the attributes of the common data classes as defined in IEC 61850-7-3 or IEC extensions to it for IEDs claiming conformance to those standards, and the SCL shall reflect the same order.

Figure 6 is the graphical illustration of a hierarchical type description of a logical node instance, as produced by the algorithm of Figure 4.

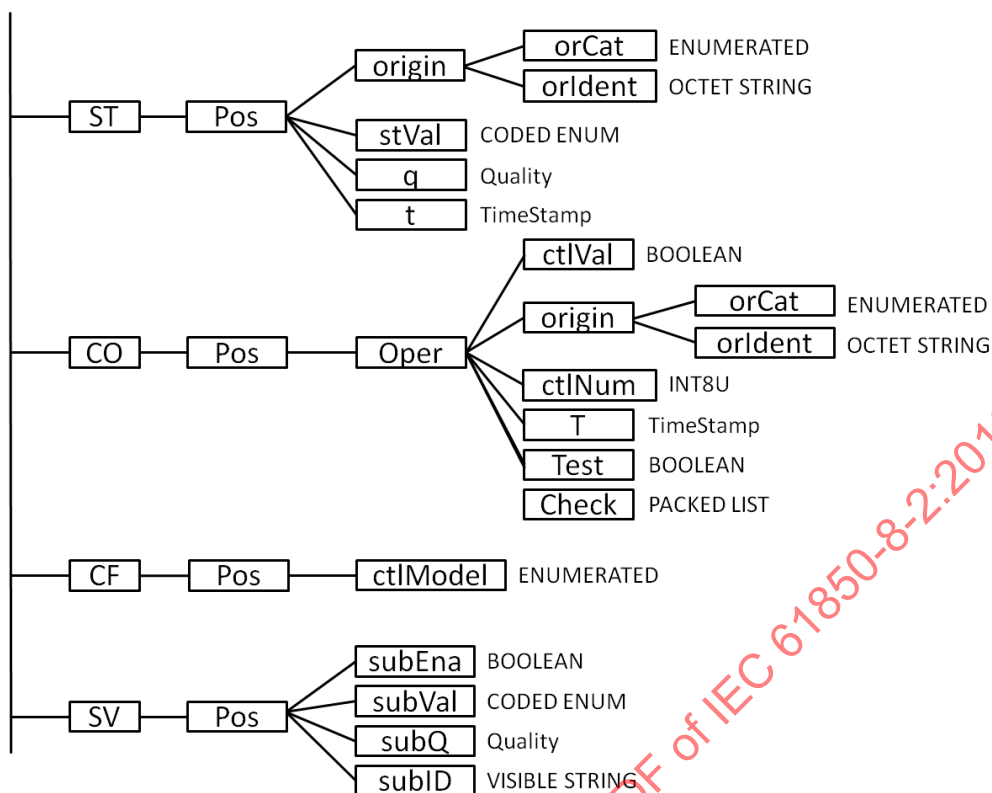


Figure 6 – Example of Logical Node type description

Additionally to this mapping to one Named Variable with a hierarchical structure, a Logical Node can also be seen as a collection of separated Named Variables, one for each component (node or leaf) in the hierarchical structure. The name of the Named Variable corresponding to a given component of the tree is created through the concatenation of the component names present in the path to the given component, plus the name of the given component itself, with all names separated by "\$". The nesting of the concatenation shall be constrained by the maximum identifier size: 64 characters. Figure 7 presents the list of the flattened Named Variables derived from the LN structure of Figure 6.

CSWI1
CSWI1\$ST
 CSWI1\$ST\$Pos
 CSWI1\$ST\$Pos\$origin\$orCat
 CSWI1\$ST\$Pos\$origin\$orIdent
 CSWI1\$ST\$Pos\$stVal
 CSWI1\$ST\$Pos\$q
 CSWI1\$ST\$Pos\$t
CSWI1\$CO
 CSWI1\$CO\$Pos
 CSWI1\$CO\$Pos\$Oper\$ctlVal
 CSWI1\$CO\$Pos\$Oper\$origin\$orCat
 CSWI1\$CO\$Pos\$Oper\$origin\$orIdent
 CSWI1\$CO\$Pos\$Oper\$ctlNum
 CSWI1\$CO\$Pos\$Oper\$T
 CSWI1\$CO\$Pos\$Oper\$Test
 CSWI1\$CO\$Pos\$Oper\$Check
CSWI1\$CF
 CSWI1\$CF\$Pos
 CSWI1\$CF\$Pos\$ctlModel
CSWI1\$SV
 CSWI1\$SV\$Pos
 CSWI1\$SV\$Pos\$subEna
 CSWI1\$SV\$Pos\$subVal
 CSWI1\$SV\$Pos\$subQ
 CSWI1\$SV\$Pos\$subID

Figure 7 – List of the flattened Named Variables corresponding to an LN

The type description of a flattened Named Variable corresponding to a given component is represented by the sub-tree starting at this component within of the overall LN structure.

8.2.3 Mapping of references of Logical Nodes to VariableAccessSpecifications

ACSI services parameters representing references to logical nodes are mapped within the XML payloads on XML elements of type VariableAccessSpecification. The VariableSpecification of such a mapping shall include an ObjectName type. If a LogicalDevice name is present within the LNReference, the child element within the ObjectName shall be domain-Specific. If an “@” is present, the child element shall be aa-specific. If neither is present, the child element shall be vmd-specific.

Within a domain-specific element, the domainId shall be the name of the LogicalDevice and the ItemId shall be the name of the LogicalNode. Figure 8 is a mapping example for the reference LDName/LNName = CONTROL/LLN0.

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL </domainId>
            <itemId> LLN0 </itemId>
          </domain-specific>
        </name>
      </variableSpecification>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 8 – XML mapping of a LNReference with direct access

An alternate specification shall be supported when the LNReference is constrained to one specific FC (in the case of GetAllDataValues see 12.3.1). Figure 9 is an example of VariableAccessSpecification where an alternateAccess element is used for encoding the functional constraint ST.

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL </domainId>
            <itemId> GGIO1 </itemId>
          </domain-specific>
        </name>
      </variableSpecification>
      <alternateAccess>
        <CHOICE>
          <unnamed>
            <selectAlternateAccess>
              <accessSelection>
                <component> ST </component>
              </accessSelection>
            </selectAlternateAccess>
          </unnamed>
        </CHOICE>
      </alternateAccess>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 9 – XML mapping of a LNReference with alternate access

8.2.4 Mapping of DataObjects reference to VariableAccessSpecifications

Logical Nodes, according to IEC 61850-7-2, consist of one or more DataObjects. The DataObjects themselves can be constructed with SubDataObjects which can themselves contain SubDataObjects. Flattened named variables can be associated to DataObjects and SubDataObjects whose name is based upon the hierarchically named component of the data found within the Logical Node. Each level of hierarchy will be delimited through the use of a "\$".

EXAMPLES: <LNVariableName>\$<FC>\$<LNDataObjectName1> (e.g. XCBR1\$ST\$Pos)

<LNVariableName>\$<FC>\$<LNDataObjectName1>\$<SubDataObjectName1> (e.g. MMXU1\$MX\$A\$phsA)

The preferred way is to map such references as a VariableSpecification within a VariableAccessSpecification, and to encode the flattened named variable within the itemId element (see example in Figure 10).

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL</domainId>
            <itemId> XCBR1$ST$Pos</itemId>
          </domain-specific>
        </name>
      </variableSpecification>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 10 – Direct XML mapping of a FCD

Any reference of a SubDataObject that contains an array shall use an AlternateAccessSpecification in the variableAccessSpecifications as soon as one of the array element is addressed (see 8.2.6).

8.2.5 Mapping of DataAttributes (DataAttr) reference to VariableAccessSpecifications

The data attributes DataAttr of the DataObjects map in a similar manner to the DataObjects. However, the attribute name is also included within the hierarchy (see examples below and Figure 11).

EXAMPLES: <LNVariableName>\${FC}\${LNDDataName1}\${AttributeNames1} (e.g. XCBR1\$ST\$Pos\$stVal)

<LNVariableName>\${FC}\${LNDDataName1}\${AttributeNames1}\${subDataAttributeName1}

(e.g. XBCR1\$ST\$Pos\$origin\$orCat)

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL</domainId>
            <itemId> XCBR1$ST$Pos$stVal</itemId>
          </domain-specific>
        </name>
      </variableSpecification>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 11 – Direct XML mapping of a FCDA

Any reference of a DataAttribute, SubDataAttribute that contains an array shall use an AlternateAccessSpecification in the variableAccessSpecifications as soon as one of the array element is addressed (see 8.2.6).

8.2.6 Usage of alternate access for DataObjects and DataAttributes references

A reference shall use an alternate access as soon as an array element is addressed. However the alternate access may be used in other cases. The alternate access consists in adding an

element <alternateAccess/> after the element <variableSpecification/>. See also clauses regarding AlternateAccess conformance in 24.2.2.2.1.

The character '\$' is not allowed within values of <component/> elements in the alternate access part.

It is recommended to always use the choice <unnamed/> within the alternate access part.

The examples below illustrate the usage of such an alternate access in the case respectively of:

Figure 12: a nested component without any array.

NOTE This reference might have been encoded without alternate access with itemId= LLN0\$ST\$Beh\$stVal

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDDevice</domainId>
      <itemId>LLN0</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <component>ST</component>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>Beh</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <component>stVal</component>
                        </accessSelection>
                        <alternateAccess/>
                      </selectAlternateAccess>
                    </unnamed>
                  </CHOICE>
                </alternateAccess>
              </selectAlternateAccess>
            </unnamed>
          </CHOICE>
        </alternateAccess>
      </selectAlternateAccess>
    </unnamed>
  </CHOICE>
</alternateAccess>

```

Figure 12 – Alternate access without array element

Figure 13: a nested component within an array element. Note that the component path before the index may have been encoded as a flattened variable within the <variableSpecification/> element (see Figure 14).

```
<variableSpecification>  
  <name>  
    <domain-specific>  
      <domainId>LDevice</domainId>  
      <itemId>MHA11</itemId>  
    </domain-specific>  
  </name>  
</variableSpecification>  
<alternateAccess>  
  <CHOICE>  
    <unnamed>  
      <selectAlternateAccess>  
        <accessSelection>  
          <component>MX</component>  
        </accessSelection>  
      </selectAlternateAccess>  
    </CHOICE>  
  </alternateAccess>  
  <CHOICE>  
    <unnamed>  
      <selectAlternateAccess>  
        <accessSelection>  
          <component>HA</component>  
        </accessSelection>  
      </selectAlternateAccess>  
    </CHOICE>  
  </alternateAccess>  
  <CHOICE>  
    <unnamed>  
      <selectAlternateAccess>  
        <accessSelection>  
          <component>phsAHar</component>  
        </accessSelection>  
      </selectAlternateAccess>  
    </CHOICE>  
  </alternateAccess>  
  <CHOICE>  
    <unnamed>  
      <selectAlternateAccess>  
        <accessSelection>  
          <index>7</index>  
        </accessSelection>  
      </selectAlternateAccess>  
    </CHOICE>  
  </alternateAccess>  
  <CHOICE>  
    <unnamed>  
      <selectAlternateAccess>  
        <accessSelection>  
          <component>cVal</component>  
        </accessSelection>  
      </selectAlternateAccess>  
    </CHOICE>  
  </alternateAccess>  
  <CHOICE>  
    <unnamed>  
      <selectAlternateAccess>  
        <accessSelection>  
          <component>mag</component>  
        </accessSelection>  
      </selectAlternateAccess>  
    </CHOICE>  
  </alternateAccess>  
  <CHOICE>  
    <unnamed>  
      <selectAlternateAccess>  
        <accessSelection>  
          <component>f</component>  
        </accessSelection>  
      </selectAlternateAccess>  
    </CHOICE>  
  </alternateAccess>  
</CHOICE>  
</alternateAccess>
```

```

</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>

```

Figure 13 – Alternate access with array element

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDevice</domainId>
      <itemId>MHAI1$MX$HA$phsAHar</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <index>7</index>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>cVal</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <component>mag</component>
                        </accessSelection>
                        <alternateAccess>
                          <CHOICE>
                            <unnamed>
                              <selectAlternateAccess>
                                <accessSelection>
                                  <component>f</component>
                                </accessSelection>
                                <alternateAccess/>
                              </selectAlternateAccess>
                            </unnamed>
                          </CHOICE>
                        </alternateAccess>
                      </selectAlternateAccess>
                    </unnamed>
                  </CHOICE>
                </alternateAccess>
              </selectAlternateAccess>
            </unnamed>
          </CHOICE>
        </selectAlternateAccess>
      </unnamed>
    </CHOICE>
  </alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</variableSpecification>

```

Figure 14 – Alternate access with flattened variable and array element

8.3 Mapping of IEC 61850-7-2 data attributes

8.3.1 BasicTypes

Within the XMLpayload, the BasicTypes of IEC 61850-7-2 are mapped on the XML elements described in Table 3. These elements are those defined in XSD file for the complexType Data (see Annex G).

Table 3 – Mapping of ACSI BasicTypes

IEC 61850-7-2 BasicType name	XML element name	XML type used for the value	Comment
BOOLEAN	<boolean/>	BOOLEAN	Example: <boolean>true</boolean>
INT8	<integer/>	INTEGER	INTEGER is defined in XSD as a restriction of xsd:integer Example: <integer>-99</integer>
INT16	<integer/>		
INT32	<integer/>		
INT64	<integer/>		
INT8U	<unsigned/>		INTEGER is defined in XSD as a restriction of xsd:integer Example: <unsigned>99</unsigned>
INT16U	<unsigned/>		
INT24U	<unsigned/>		
INT32U	<unsigned/>		
FLOAT32	<floating-point/>	OCTET_STRING	OCTET_STRING representing 4 octets. Range of values and precision as specified by IEEE 754 single precision floating point. NOTE 1 The value of Not-A-Number (NaN) is valid. However, the processing of a value indicating Not-A-Number (NaN) is a local issue. NOTE 2 OCTET_STRING is a restriction of xsd:string for encoding hexadecimal values
ENUMERATED	<integer/>	INTEGER	INTEGER is defined in XSD as a restriction of xsd:integer Used here for an ordered set of values, defined where type is used. See 8.3.2.2. Example: Beh.stVal = Test (3) would be encoded as <integer>3</integer>
CODED ENUM	<bit-string/>	BIT_STRING	BIT_STRING is an xsd:string type, restricted to zero or more "0" and "1" characters (see 8.3.2.1). Used here for an ordered set of values, defined where type is used. See also 8.3.2.3. Example: dbPos is a 2 bit length bit string <bit-string>01</bit-string>
OCTET STRING	<octet-string/>	OCTET_STRING	OCTET_STRING is a restriction of xsd:string for encoding hexadecimal values. Max length shall be defined where type is used. See also 8.3.2.4. Example: DPC or Ident is a 64 octet string. An example with 7 octets would be encoded as <octet-string>53544154494F4E</octet-string>
VISIBLE STRING	<visible-string/>	VisibleString	VisibleString is an xsd:string restricted to visible characters. Max length shall be defined where type is used. See 8.3.2.5. Example: RptID of a ReportControl is a Visible String129 <visible-string>CONTROL/GGIO1\$BR\$simpleBReport05</visible-string>

IEC 61850-7-2 BasicType name	XML element name	XML type used for the value	Comment
UNICODE STRING	<mMSString/>	MMSString	Defined in XSD as a mixed content of text and XML elements encoding control characters. Max length shall be defined where type is used. See 8.3.2.6. Example: <mMSString>ābcdēfgh<nul/>ījkŁμñópq<soh/>r\$tùvwxyz</mMSString>
For the purposes of the rest of the clauses in this document, the precision of unsigned / integer shall be determined by the IEC 61850-7-2 or IEC 61850-7-3 unless otherwise specified.			

8.3.2 Additional definitions of BasicType

8.3.2.1 General statement for BIT STRING

Within this standard, the reference to a BIT STRING shall indicate the use of an element with name="bit-string" and type="BIT_STRING" as defined by the normative XSD (see Annex G).

BIT_STRING is a restriction of xsd:string type, restricted to zero or more "0" and "1" characters.

NOTE Characters of a BIT_STRING are however referred as bits in the present document.

Bit(0) of the BIT STRING is mapped to the left most character in the XML String.

The amount of bits conveyed by a sent variable length bit-string shall always be equal to the maximum length of the defined variable length bit-string at the sender side, so that forward semantic compatibility can be maintained. Zero (0) length bit-string are not allowed in the context of IEC 61850-8-2.

8.3.2.2 ENUMERATED

The IEC 61850-7-2 ENUMERATED shall map to an <integer/> element with an xsd:integer value. Values that are 0 or greater shall be reserved for values standardized within IEC 61850. Values that are less than 0 shall be considered private values except for the IEC 61850-7-3 Multiplier enumeration where the enumeration value delivers the multiplier value to use. The range of the allowed values is specified in IEC 61850-7-2, IEC 61850-7-3 and IEC 61850-7-4.

The first ordered value, of the standardized enumerated values, shall be assigned the value of zero (0).

Enumerated values outside the standardized range of values shall be assumed to be value extensions and shall not cause a protocol error.

For IEC 61850 enumerated values, where the actual numeric values are specified, the integer values within the context of IEC 61850-8-2 shall be the same as the numeric values specified.

8.3.2.3 CODED ENUM

The IEC 61850-7-2 CODED ENUM is mapped on a <bit-string/> element with a BIT_STRING type value, as defined in the normative XSD (see 8.3.2.1). The size of the string shall be the number of bits required to represent the maximum enumerate value defined in IEC 61850-7-2. It shall represent the unsigned integer encoding of the enumerated values in the order specified in IEC 61850-7-2.

For the purposes of IEC 61850-8-2, CODED ENUMs that appear within a PACKED LIST or an array of other CODED ENUMs are treated as special cases.

8.3.2.4 OCTET STRING

The IEC 61850-7-2 OCTET STRING shall map to an <octet-string/> element with an OCTET_STRING type value. OCTET_STRING is a restriction of xsd:string for encoding hexadecimal values, i.e. based on the character set [0-9][A-F][a-f] but including possibly extra space characters. The maximum size shall be as specified in IEC 61850-7-2 and IEC 61850-7-3 and do not include the extra spaces added for readability. The NULL string has a length 0.

NOTE The term octet is used within the present document for designating the two characters of an OCTET_STRING representing the hexadecimal value of a binary octet.

8.3.2.5 VISIBLE STRING

The IEC 61850-7-2 VISIBLE STRING shall map to a <visible-string/> element with a VisibleString type value. VisibleString is defined in the XSD as a an xsd:string restricted to visible characters, i.e. the range from Space (ascii code 32) to Tilde (ascii code 126). The maximum size shall be as specified in IEC 61850-7-2 and IEC 61850-7-3. The NULL string has a length 0.

The character set shall be constrained to ISO646String.

8.3.2.6 UNICODE STRING

The IEC 61850-7-2 UNICODE STRING shall map to a <mMSString/> element with a mMSString complexType value. mMSString complexType may contain a mix of text (UTF8 encoding) and control characters encoded as explicit XML elements like <nul/> or <soh/>. The maximum size shall be the number of Unicode characters as specified in IEC 61850-7-2 and IEC 61850-7-3. The serialization of a Unicode character occurs as specified in RFC 3629, and may involve up to 4 octets. The NULL string has a length 0.

8.3.2.7 Array

An array shall map to an <array/> XML element with one more child XML elements.

If the elements of the array are of a BasicType, the child XML elements shall be directly those described in Table 3 with one XML element corresponding to each array element.

Otherwise, the child elements shall be <structure/> elements which can recursively include BasicType elements and/or <structure/> elements.

See in XSD file of Annex G for the definition of the elements “array” within the complexType “Data”.

Example: encoding of an array of 2 points

```
<array>
  <structure>
    <floating-point>3f800000</floating-point>
    <floating-point>40000000</floating-point>
    <floating-point>40400000</floating-point>
  </structure>
</structure>
```

```

        <floating-point>40800000</floating-point>
        <floating-point>40a00000</floating-point>
        <floating-point>40c00000</floating-point>
    </structure>
</array>

```

8.3.3 Common ACSITypes

8.3.3.1 ObjectName

This type only appears in the context of ObjectReference. The allowed character set within instances of ObjectName is defined in IEC 61850-7-2.

8.3.3.2 ObjectReference

8.3.3.2.1 General

The IEC 61850-7-2 Object reference mapping depends on the context where the object reference is defined:

- Within the scope of control block attributes: see 8.3.3.2.2.
- Within the scope of data attributes (Tracking CDCs, ORG): see 8.3.3.2.3.
- Within the scope of DataRef of reports: see 8.3.3.2.4.

In all cases, the ObjectReference is mapped as a VISIBLE STRING with a maximum length of 129 octets.

8.3.3.2.2 Object Reference within Control Blocks

The value shall be constructed as:

- AA-specific scope objects: @<Object Name>
- Domain specific: <Domain Name>/<Object Name>
- VMD-specific: /<Object Name>

The maximum Object Name and Domain Name size shall be 64.

The allowed character set is and the same as ObjectName plus the “\$”, “/” and “@” characters.

8.3.3.2.3 Object Reference within CDCs (Tracking, ORG).

The value shall be constructed as specified in IEC 61850- 7-2.

The allowed character set is defined in IEC 61850-7-2. The value shall be constructed as:

- Reference to a Logical Device:
<LDName>
- Reference to a Logical Node:
<LDName>/<LNName>
- Reference to a DataSet:
<LDName>/<LNName>.<DataSetName>
- Reference to a DataObject:

<LDName>/<LNName>.<DataObjectName>[.<SubDataObjectName>[. ...]]

- Reference to a DataAttribute:

<LDName>/<LNName>.<DataObjectName>[.<SubDataObjectName>[. ...]].<DataAttributeName>[(<NumArrayElement>)][.<SubDataAttributeName>[. ...]]

- Reference to a Control Block

<LDName>/<LNName>.<CBName>

8.3.3.2.4 Object Reference within DataRef of reports.

The value shall be constructed as:

<Domain Name>/<Object Name>

The allowed character set is defined in IEC 61850-7-2.

The value shall be constructed as:

- Reference to a FCD DataObject:

<LDName>/<LNName>\$<FC>\$<DataObjectName>[\$<SubDataObjectName>[(<NumArrayElement>)][\$...]]

Example: The datRef representing the ACSI reference LDevice/MHA11.HA.phsAHar(7) functionally constrained MX will be LD1/MMHA1\$MX\$HA\$phsAHar(7)

- Reference to a FCDA

<LDName>/<LNName>\$<FC>\$<DataObjectName>[\$<SubDataObjectName>[(<NumArrayElement>)][\$...]] \$<DataAttributeName> [(<NumArrayElement>)] [\$<SubDataAttributeName>[\$...]]

Example: The datRef representing the ACSI reference LDevice/MHA11.HA.phsAHar(7).cVal.mag.f will be LD1/MMHA1\$MX\$HA\$phsAHar(7)\$cVal\$mag\$f

8.3.3.3 PhyComAddr

8.3.3.3.1 PhyComAddr for Layer 2 communication

The PhyComAddr, used in context of Layer 2 communication by different control blocks, is mapped to the structure defined in Table 4.

Table 4 – PhyComAddr structure for Layer 2 communication

Component Name	Data Type	m/o	Comments
Addr	OCTET STRING	m	Length is 6 Octets and contains the value of the destination Media Access Control (MAC) address to which the GOOSE message is to be sent. The address shall be an Ethernet address that has the multicast bit set TRUE.
PRIORITY	INT8U	m	Range of values shall be limited from 0 to 7.
VID	INT16U	m	Range of values shall be limited from 0 to 4 095.
APPID	INT16U	m	As defined in IEC 61850-8-1:2011+AMD1:2018 CSV, Annex C

8.3.3.3.2 PhyComAddr for UDP/IP communication

The PhyComAddr, used in context of UDP/IP communication by different control blocks, is mapped to the structure defined in Table 5.

Table 5 – PhyComAddr for UDP/IP communication

Component Name	Data Type	m/o	Comments
Addr	OCTET STRING	m	The OCTET STRING has a variable length of 16 octets. It shall use a length of 4 Octet to encode an IPv4 address, whereas it shall use a length of 16 Octets to encode an IPv6 address. Addr represents the UDP/IP destination address
isIPv6	BOOLEAN	m	Shall be TRUE if the Addr is an IPv6 address. Shall be FALSE if an IPv4 is specified
PRIORITY	INT8U	m	Range of values shall be limited from 0 to 7. This value is for Layer-2 priority (e.g. through local switches on the LAN of the publisher). The value is not guaranteed to be conveyed through a router.
VID	INT16U	m	Range of values shall be limited from 0 to 4 095. The value of zero (0) is the default value and indicates that no 802.1 VLAN is specified. This value is for Layer-2 ID (e.g. through local switches on the LAN of the publisher). The value is not guaranteed to be conveyed through a router.
APPID	Unsigned16	m	As defined in IEC 61850-8-1:2011 +AMD1:2018 CSV, Annex C
gwAddr	OCTET-STRING	m	The OCTET STRING has a variable length of 16 octets. It shall use a length of 4 Octet to encode an IPv4 address, whereas it shall use a length of 16 Octets to encode an IPv6 address. GwAddr represents the address of the first hop gateway.
tOS	INT8U	m	Specifies the TOS field of the IPv4 header or the Traffic Class field of the IPv6 header Range of values shall be limited from – 0 to 2^6-1 if IPv6 is FALSE. – 0 to 2^8-1 if IPv6 is TRUE
iGMPSrc	OCTET-STRING		The OCTET STRING has a variable length of 16 octets. It shall use a length of 4 Octet to encode an IPv4 address, whereas it shall use a length of 16 Octets to encode an IPv6 address. IGMPSrc represents the unicast src address to be used for IGMPv3 subscription.

8.3.3.4 ServiceError**8.3.3.4.1 General**

There are multiple possibilities for ACSI negative service responses, depending on the layer where the error occurs. Table D.2 makes the synthesis of the various kinds of errors.

8.3.3.4.2 XMPP errors

Per XMPP, unrecoverable errors occurring during the stream setup or during the stanza transmission, are indicated by a <stream:error/> message containing a child element qualified by the namespace urn:ietf:params:xml:ns:xmpp-streams. This namespace as well as the full list of error conditions are specified in RFC 6120.

Per XMPP, recoverable errors occurring during the stanza transmission or processing (iq or message), are indicated by a stanza of the same kind (iq, message) of type="error" containing an <error/> child element, qualified by the namespace urn:ietf:params:xml:ns:xmpp-stanzas. This namespace as well as the full list of error conditions are specified in RFC 6120.

8.3.3.4.3 Association context errors

The errors raised during the management of the association context shall be mapped to XMPP stanzas as specified in IEC 62351-4.

8.3.3.4.4 Security errors

The errors raised by the end-to-end security mechanism shall be mapped to XMPP stanzas as specified in IEC 62351-4.

8.3.3.4.5 Mapping errors

When an XMPP Client receives an <iq/> or a <message/> stanza carrying a conformant payload but where the mapping over XMPP is not conformant with Table D.1 (it doesn't use the right stanza or type), it shall respond by an <message/> stanza of type="normal" containing a rejectPDU with rejectReason= unrecognized-service.

8.3.3.4.6 Reject errors

If a client issues a request for an ACSI service mapped on a service element (see Table D.1) which is not supported or in case the server cannot recognize the request as a well formed confirmed-RequestPDU the server shall return a <rejectPDU/>.

The indication of support for a service element shall be declared in the PICs/PIXIT and in the <servicesSupportedCalled/> element of the Associate response (mapped to an initiate-ResponsePDU).

If the service is not supported (e.g. the appropriate element in the servicesSupportedCalled is FALSE) and the Client issues a well formed confirmed-RequestPDU of such a service, the server shall return a Reject as specified below. A similar Reject shall be sent when a Client or a Server detects that a received Service PDU doesn't match the compatibility rules defined in clause 10.2.2.1.4:

originalInvokeID:	shall be present
rejectReason:	shall be a reason of confirmed-requestPDU with a value of unrecognized-service

If a mal-formed PDU is received, the contents of the Reject shall be a local issue.

8.3.3.4.7 ACSI Service errors

Errors occurring in the processing of a supported service are returned to the client through a <confirmed-ErrorPDU/> or another type of PDU dedicated to errors.

These errors are defined in the clause related to the mapping of each service. However, the following clauses gives some definitions shared by several services.

8.3.3.4.8 ACSI services mapped on GetNameList

If a service request mapped on <getNameList/> or <extendedGetNameList/> specifies conflicting <objectClass/> and <objectScope/> specifications (see Table 6) then a <confirmed-ErrorPDU/> shall be returned. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><service/> with an errorCode=<object-constraint-conflict/>.

Table 6 – GetNameList conflicting IEC 61850 objectClass and objectScope

ObjectClass	objectScope
	Parameter Values
domain	domainSpecific
journal	aaSpecific
domain	aaSpecific

Table 7 defines the mapping of ACSI ServiceError on the <serviceError/> element for the ACSI services using GetNameList.

Table 7 – Service error mappings for ACSI services using GetNameList

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
parameter-value-inconsistent	service	object-constraint-conflict	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	

For other requests that are processed and no matching objects are located, a response+ shall be returned with the <listOfIdentifiers/> being NULL and moreFollows set to FALSE.

NOTE This empty response+ may occur for example when <extendedGetNameList/> is used and the element <domainSpecific/> contains a reference within an existing Logical Device, to an object that doesn't exist or for which the client doesn't have access privileges.

8.3.3.4.9 ACSI services mapped on a Read Service

This subclause concerns the common errors shared by the services where the request message is a <confirmed-RequestPDU/> including a <read/> element.

However, the reference of the object to be read may be either included in the request as an element <variableSpecification/> (e.g. case of GetDataValues) or may be a Data Set on the server side, referenced as a <variableListName/> element (case of GetDataSetvalues). In this case, the objects whose access may cause the errors defined in this clause are the ones referenced as members of the Data Set.

Errors may be returned as a Response+ or as a <confirmed-ErrorPDU/> element including a <serviceError>.

The following are the definitions of the possible errors that can be returned:

- If the request causes a Response that exceeds the negotiated PDU size, a <confirmed-ErrorPDU/> shall be returned and the <serviceError/> shall be <errorClass/><service/> with an errorCode=<pdu-size/>. This shall be mapped to the ACSI ServiceError of failed-due-to-communications-constraint.

- If the Client does not have access privileges to an object to be read, a Response+ shall be returned indicating a <failure/> with DataAccessError=<object-access-denied/> or DataAccessError=<object-non-existent/>.
- If an object to be read does not exist, a Response+ shall be returned indicating a <failure/> with DataAccessError=<object-non-existent/>.
- If the accessibility of an object to be read depends on the value of another object (e.g. Setting group values). a Response+ shall be returned indicating a <failure/> with DataAccessError=<temporarily-unavailable/>.

See Table 8 for read service error mappings.

Table 8 – Read service error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
failed-due-to-communications-constraint	service	pdu-size	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	
ACSI service error	Response+		Comments
	DataAccessError		
access-violation	object-access-denied		
instance-not-available	object-non-existent		
failed-due-to-communications-constraint	Any other DataAccessErrors		
access-not-allowed-in-current-state	temporarily-unavailable		

8.3.3.4.10 ACSI services mapped on a Write Service

This subclause concerns the common errors shared by the services where the request message is a <confirmed-RequestPDU/> including a <write/> element.

However, the reference of the object to be written may be either included in the request as an element <variableSpecification/> (e.g. case of SetDataValues) or may be a Data Set on the server side, referenced as a <variableListName/> element (case of SetDataSetvalues). In this case, the objects whose access may cause the errors defined in this clause are the ones referenced as members of the Data Set.

Errors may be returned as a Response+ or as a <rejectPDU/> element. Within a Response+ there is a sequence of results that indicate the success or failure corresponding to the individual objects to be read.

The following are the definitions of the possible errors that can be returned:

- A Request that exceeds the negotiated PDU size shall cause a <rejectPDU/> with a rejectPDUType=<confirmed-requestPDU/> and a rejectCode=<value-out-of-range/>. This shall be mapped to the ACSI ServiceError of failed-due-to-communications-constraint.
- If the Client does not have access privileges to an object to be written, a Response+ shall be returned indicating a <failure/> with DataAccessError=<object-access-denied/> or DataAccessError=<object-non-existent/>.
- If an object to be written does not exist, a Response+ shall be returned indicating a <failure/> with DataAccessError=<object-non-existent/>.

- If an object to be written has been reserved by another Client (e.g. a ControlBlock or control structure) or if the writability/accessability of the object depends on the value of another object (e.g. Setting group values), a Response+ shall return a <failure/> indicating dataAccessError=<temporarily-unavailable/>.
- There are several issues that are associated with the Data provided in the request compared with their Data Type Definitions expected on the server side:
 - Fixed size Data Type is expected and the Data provided does not match the defined size: a <failure/> within the Response+ indicates dataAccessError=<type-inconsistent/>.
 - Variable length size Data Types expected and the Data provided does exceeds the maximum defined size: a <failure/> within the Response+ indicates dataAccessError=<type-inconsistent/>.
 - Variable length size, and extensible, Data Types expected (e.g. BITSTRINGS) but where the Data provided does exceeds the maximum defined size: no <failure/> may be generated.
- An attempt to write an object which is entirely read-only or one or more of its sub-components is read-only (e.g. a structure component or array element) shall result in a Response+ indicating a dataAccessError=<object-access-denied/>.
- ENUMERATIONS are mapped to some size of an Integer DataType. If the value is beyond the defined set of values allowed for the ENUMERATION, the IEC 61850 application shall return a DataAccessError=<object-value-invalid/>.

NOTE This applies to both the defined positive and supported negative ranges of enumerated values.

See Table 9 for write service error mappings.

Table 9 – Write service error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	The exhaustive list of error class and code is defined by the complex type ServiceError in the XSD of Annex G
ACSI service error	MMS Response+		Comments
	DataAccessError		
access-violation	object-access-denied		
instance-not-available	object-non-existent		
instance-locked-by-other-client	temporarily-unavailable		
type-conflict	type-inconsistent		
parameter-value-inconsistent	object-value-invalid		
failed-due-to-communications-constraint	Any other DataAccessErrors		
access-not-allowed-in-current-state	temporarily-unavailable		
ACSI service error	Reject		Comments
	Reject PDU Type	Reject Code	
parameter-value-inconsistent	pdu-error	Invalid-pdu	

8.3.3.4.11 ACSI services mapped on a FileDirectory

This clause concerns the common errors shared by the services where the request message is a <confirmed-RequestPDU/> including a <FileDirectory/> element, i.e. the ACSI services GetServerDirectory (when requesting FILE) and the GetFileAttributeValues.

- A request that specifies a directory that does not exist shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<file-non-existent/>.
- A request that specifies a FileDirectory for which the Client does not have access privileges shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<file-access-denied/> or <errorClass/><file/> with an errorCode=<file-non-existent/>.
- A request that specifies a FileSpecification that can not be interpreted shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<filename-syntax-error/>.
- A request that specifies a <continueAfter/> with a Filename that does not exist, shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<file-non-existent/>.

See Table 10 for GetFileAttributeValues service error mappings.

Table 10 – GetFileAttributeValues service error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	file	file-non-existent	
access-violation	file	file-access-denied	
parameter-value-inappropriate	file	filename-syntax-error	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	

8.3.3.5 EntryID

The ACSI EntryID type shall be mapped as an 8 octets fixed length OCTET STRING ACSI type (see 8.3.2.4). The content is IEC 61850 server specific. It is suggested that the format of the content be documented as part of the implementation's PIXIT statement.

The reserved value for EntryID is zero (0), i.e. value <octet-string>0000000000000000</octet-string>.

8.3.3.6 PACKED LIST

A PACKED LIST type shall be mapped as a BIT STRING ACSI type (see 8.3.2.1). The minimum size of the xsd:string shall be determined by the number of bits required to encode the members of the PACKED LIST. The order of the bits, in the xsd:string, shall be that the first member of the PACKED LIST shall be mapped to Bit(0). The other members shall be mapped in consecutive bits in the order specified by IEC 61850-7-2.

Example: quality is a 13 bits length bit-string <bit-string>0000000000000</bit-string>

A PACKED LIST shall support at least the number of bits required to represent the maximum enumerated value defined. Bits that are returned in excess of that number of bits may be ignored.

Bit 0 shall be the leftmost (most significant) bit of the first octet. Bit 7 shall be the rightmost (least significant) bit of the first octet. Bit 8 shall be the leftmost (most significant) bit of the

second octet. Bit 15 shall be the rightmost (least significant) bit of the second octet. This shall be continued in that way in further octets.

There are special cases that are individually mapped and do not conform to the general rule. These are the Timestamp type (specified in 8.3.3.7), quality type (specified in 8.3.4), TriggerConditions type (specified in 8.3.3.9) and ReasonForInclusion type (specified in 8.3.3.10 and 8.3.3.11).

8.3.3.7 Timestamp

Timestamp shall be mapped to an element with name="utc-time" and type="OCTET_STRING", with a fixed size of 8 octets (unless another mapping for timestamp is specified in this document for a dedicated case).

Example: <utc-time>017215E00000001F</utc-time>

The first four octets are encoding the SecondSinceEpoch attribute.

The fifth to seventh octets are encoding the FractionOfSecond attribute.

The quality flags are located in the eighth octet. These flags are defined in Table 11.

Table 11 – Encoding of IEC 61850-7-2 TimeQuality

Bit number	Value	Meaning
0		Leap Second Known
1		ClockFailure
2		Clock not synchronized
3-7		Time accuracy of fractions of second
	00000	0 bit of accuracy
	00001	1 bit of accuracy
	00010	2 bits of accuracy
	00011	3 bits of accuracy
	00100 - 11000	Integer value of number of bits of accuracy
	11001 - 11110	Invalid
	11111	unspecified

Bit 0 shall be the most significant bit of octet 7. Bit 7 shall be the least significant bit of octet 7.

Example: in <utc-time>017215E00000001F</utc-time>, the value 1F hexadecimal of octet 7 (representing 00011111 binary) means: no Leap Second Known, no ClockFailure, Clock synchronized, Time accuracy unspecified.

8.3.3.8 EntryTime

EntryTime shall be mapped to an element with name="binary-time" and type = "TimeOfDay".

- TimeOfDay if a restriction of OCTET_STRING. The length of the value shall be six (6) octets.
- The first four octets shall contain a value indicating the number of milliseconds since midnight for the current date. The last two octets contain the relative day since 1 January 1984.

Example: <binary-time>0316E6032DD2</ binary-time > is the encoding for 14h 23min 52s and 232 ms on 12 February 2016. "0316E603" is the number of milliseconds since midnight, 2DD2 is the number of days since 1 January 1984.

8.3.3.9 TriggerConditions

The values for TriggerConditions are encoded as a PACKED LIST, i.e. as a BIT STRING.

Bit(0) is reserved so that the minimum size shall be six(6) bits. Additional bits are reserved for future versions of the standard. The bits are assigned as specified in Table 12.

Table 12 – Encoding of the TriggerConditions

Trigger Condition	MMS bit position
Reserved (reserved to provide backward compatibility with UCA 2.0)	0
data-change	1
quality-change	2
data-update	3
integrity	4
general-interrogation	5

NOTE A future extension of the TriggerConditions will take care of defining Bit 6 as being reserved to keep a certain alignment between the TriggerConditions type, the ReasonForInclusionInReport and ReasonForInclusionInLog types.

8.3.3.10 Reason for inclusion in reports (ReasonForInclusionInReport)

The values for ReasonForInclusionInReport are encoded as a PACKED LIST, i.e. as a BIT STRING.

Bit(0) is reserved so that the minimum size shall be six(6) bits. Additional bits are reserved for future versions of the standard. The bits are assigned as specified in Table 13.

Table 13 – Encoding of the ReasonForInclusionInReport

Reason for inclusion	MMS bit position
Reserved (reserved to provide backward compatibility with UCA 2.0)	0
data-change	1
quality-change	2
data-update	3
integrity	4
general-interrogation	5

NOTE A future extension of the ReasonForInclusionInReport will take care of defining Bit 6 as being reserved to keep a certain alignment between the TriggerConditions type, the ReasonForInclusionInReport and ReasonForInclusionInLog types.

8.3.3.11 Reason for inclusion in logs (ReasonForInclusionInLog)

The values for ReasonCode are encoded as a PACKED LIST, i.e. as a BIT STRING.

Bit(0) is reserved so that the minimum size shall be seven (7) bits. Additional bits are reserved for future versions of the standard. The bits are assigned as specified in Table 14.

Table 14 – Encoding of the ReasonForInclusionInLog

Reason for inclusion	MMS bit position
Reserved (reserved to provide backward compatibility with UCA 2.0)	0
data-change	1
quality-change	2
data-update	3
integrity	4
general-interrogation	5
application-trigger	6

8.3.3.12 RCBReportOptions

The values for RCBReportOptions are encoded as a PACKED LIST, i.e. as a BIT STRING. The minimum size of the BITSTRING shall be ten (10) bits. Additional bits are reserved for future versions of the standard. The bits are assigned as specified in Table 15.

Table 15 – Encoding of the RCBReportOptions

ACSI value of BRCState	MMS bit position
Reserved	0
sequence-number	1
report-time-stamp	2
reason-for-inclusion	3
data-set-name	4
data-reference	5
buffer-overflow	6
entryID	7
conf-revision	8
segmentation	9

8.3.3.13 LCBLogEntryOptions

Not mapped. This Clause forces the reasonCodes to always appear in the Log.

8.3.3.14 SVMessageOptions

The values for SVMessageOptions are encoded as a PACKED LIST, i.e. as a BIT STRING. The minimum size of the BITSTRING shall be seven (7) bits. Additional bits are reserved for future versions of the standard. The bits are assigned as specified in Table 16.

Table 16 – Encoding of the SVMessageOptions

ACSI value of SVMessageOptions	MMS bit position
refresh-time	0
reserved	1
sample-rate	2
data-set-name	3
security	4
sample-mode	5
synch-source-identity	6

8.3.3.15 CheckConditions

The values for CheckConditions are encoded as a PACKED LIST, i.e. as a BIT STRING. The minimum size of the BITSTRING shall be two (2) bits. Additional bits are reserved for future versions of the standard. The bits are assigned as specified in Table 17.

Table 17 – Encoding of the CheckConditions

ACSI value of CheckConditions	MMS bit position
synchrocheck	0
interlockCheck	1

8.3.3.16 Originator

The type Originator shall be mapped to an element <structure> representing the Originator.

Example:

```
<structure>
  <integer>2</integer>
  <octet-string>53544154494F4E</octet-string>
</structure>
```

8.3.3.17 Currency

The type Currency shall be mapped to a visible string whose maximum length is three (3) characters. The string content shall be the ISO 4217 3-character currency code.

8.3.4 Mapping of quality common data attribute type specified in IEC 61850-7-2

The definition found in IEC 61850-7-2 of the Quality common data attribute, specifies a combination of PACKED LISTs, CODED ENUMs, and BOOLEAN values. The mapping of this data attribute shall be as prescribed in this Clause.

An IEC 61850-7-2 value of the type quality, shall be represented by a BIT STRING.

Bit values shall be as defined in Table 18. Bit(0) is the leftmost and all bits must be conveyed.

Future versions of the standard may add additional bits. The maximum allowed bit number shall be 15.

Table 18 – Encoding of IEC 61850-7-2 quality

Bit(s)	IEC 61850-7-3		Bit-String	
	Attribute name	Attribute value	Value	Default
0-1	Validity	Good	0 0	0 0
		Invalid	0 1	
		Reserved	1 0	
		Questionable	1 1	
2	Overflow		TRUE	FALSE
3	OutOfRange		TRUE	FALSE
4	BadReference		TRUE	FALSE
5	Oscillatory		TRUE	FALSE
6	Failure		TRUE	FALSE
7	OldData		TRUE	FALSE
8	Inconsistent		TRUE	FALSE
9	Inaccurate		TRUE	FALSE
10	Source	Process	0	0
		Substituted	1	
11	Test		TRUE	FALSE
12	OperatorBlocked		TRUE	FALSE

8.4 General mapping of data values within XML payloads

This clause applies both for data defined in IEC 61850-7-3 and control blocks defined in IEC 61850-7-2.

All the ACSI services used for conveying data values (for setting, getting or reporting purpose) are mapped on XML elements using the complexType Data. This clause explains how to instantiate this Data type in real payloads.

If the conveyed data is of a BasicType, the value is mapped to a single element as mentioned in Table 3 (e.g. <bit-string>0111110010</bit-string>).

If the conveyed data is a component including potentially (and recursively) sub-components, the component and the sub-components shall be mapped to a <structure/> element including (potentially recursively also) BasicType elements and/or <structure/> elements. Additionally, all the elements included in the component shall be individually wrapped into a <Data/> element.

Table 19 gives examples of how is encoded the values returned by a GetDataValues ACSI service when applied respectively on the FCD "CONTROL/GGIO1\$ST\$Ind3" and FCDA "CONTROL/GGIO1\$ST\$Ind3\$stVal".

Table 19 – Examples of data values encoding

Input parameter of GetDataValues request	Value of element <success/> in GetDataValues response
FCD CONTROL/GGIO1\$ST\$Ind3	<pre> <structure> <Data> <boolean>false</boolean> </Data> <Data> <bit-string>000000000000</bit-string> </Data> <Data> <utc-time>017215E30000001F</utc-time> </Data> </structure> </pre>
FCDA CONTROL/GGIO1\$ST\$Ind3\$stVal	<pre> <boolean>true</boolean> </pre>

8.5 Extended behaviour for optimization of bandwidth

As mentioned in Clause 1, the present SCSM uses globally the same abstract MMS definitions for the messages than the existing IEC 61850-8-1. However, the MMS definitions have some well-known drawbacks that the present SCSM can overcome.

The concerned ACSI services are GetLogicalDeviceDirectory, GetLogicalNodeDirectory and GetDataDirectory. For these services, a straightforward XML encoding of the abstract MMS messages produces large Response+ messages where most of the information is not required and must be therefore filtered at client side.

The idea is to keep this functioning as the default but to propose optionally an extended one, where the Response+ includes only the mandatory information. The choice between the default and extended behaviour is made at association time, thanks to the exchange of a list of supported services including possibly the value extendedGetNameList. When the extended behaviour is selected, the concerned ACSI services shall use the choice extendedGetNameList instead of their normal choice (which is respectively getNameList for GetLogicalDeviceDirectory and GetLogicalNodeDirectory, and getVariableAccessAttributes for GetDataDirectory) in their ConfirmedServiceRequest and ConfirmedServiceResponse elements. The structure of the getNameList and extendedGetNameList elements is strictly identical.

The detailed default and extended mappings are specified in the clause related to each of the affected ACSI services.

9 Server class model

9.1 General

NOTE 1 In this clause as well as in all the following clauses defining how the ACSI services are mapped on XML messages transmitted over XMPP, the association context part and the secured elements are omitted. For understanding the complete layout of the messages transmitted within the XMPP streams, please see 5.2.2, 7 and 8.

NOTE 2 In the present document, the term "PDU size" refers to the size of the Service PDU as defined in 5.2.2. This size is negotiated during the Associate service (see 10.2.2.1).

An IEC 61850-8-2 implementation shall support one or several ServiceAccessPoints dedicated to XMPP communication. Each access point shall be defined by one or more JIDs.

9.2 GetServerDirectory

9.2.1 General

The GetServerDirectory service shall be mapped differently depending on the ACSI class being requested: LOGICAL-DEVICE and FILE.

LOGICAL-DEVICE class

The GetServerDirectory request maps to an XML element <confirmed-RequestPDU/> including an element <getNameList/> where the <objectClass/> element shall have a value of <domain/>. Figure 15 gives the structure of the XML message corresponding to the user data (i.e. without the security elements which will wrap the user message in the real message on the wire).

The GetServerDirectory response maps to an XML element <confirmed-ResponsePDU/> including an element <GetNameList/>. Figure 16 gives the structure of the XML message corresponding to the user data

The ACSI ServiceErrors are detailed in 8.3.3.4.8.

Table 20 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped within respectively the request and response structures.

The GetServerDirectory service may be constrained by the negotiated PDU size limitation. In that case, the response may return only a subset of logical device names and a flag (moreFollows) indicating the need to request the next subset. The client shall then post another request, this time specifying the position in the list at which the server should continue with name retrieval. Due to this constraint, the mapping of the ACSI service is actually a sequence of requests/responses, until moreFollows=FALSE in the response.

Table 20 – Mapping of ACSI GetServerDirectory (LOGICAL DEVICE)

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNameList/>		See structure of XML request in Figure 15
<objectClass/> = LOGICAL-DEVICE	<objectClass/>	<domain/>	Fixed value
	<continueAfter/>	<LDNameX>	Optional: name of the last LD returned in the previous Response in case it contains moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNameList/>		See structure of XML response in Figure 16

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Reference[0..n]	<listOfIdentifier/>	<pre> <Identifier>LDName1</Identifier> <Identifier>LDName2</Identifier> ... </pre>	Response+ contains only the logical devices that can be viewed by the client application based on its access privileges. For implementations claiming conformance to IEC 61850, Response+ shall not be NULL. The list of identifiers shall be lexicographically ordered (based on ASCII encoding of characters)
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	See 8.3.3.4.8	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>78531</invokeID>
  <ConfirmedServiceRequest>
    <getNameList>
      <extendedObjectClass>
        <objectClass><domain></objectClass>
      </extendedObjectClass>
      <objectScope>
        <vmdSpecific/>
      </objectScope>
    </getNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 15 – XML structure of GetServerDirectory-Request (LD)

```

<confirmed-ResponsePDU>
  <invokeID>78531</invokeID>
  <ConfirmedServiceResponse>
    <getNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>CONTROL</Identifier>
        <Identifier>TEST</Identifier>
      </listOfIdentifier>
    </getNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 16 – XML structure of GetServerDirectory-Response (LD)

FILE class

The GetServerDirectory request maps to an XML element <confirmed-RequestPDU/> including an element <fileDirectory/>. Figure 17 gives the structure of the corresponding XML message.

The GetServerDirectory response maps to an XML element <confirmed-ResponsePDU/> including an element <fileDirectory/>. Figure 18 gives the structure of the XML corresponding message.

The ACSI ServiceErrors are detailed in 8.3.3.4.11.

Table 21 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped within respectively the request and response structures.

The GetServerDirectory service, if constrained by the negotiated PDU size limitation, may segment the process by returning a subset of file names and a flag (moreFollows) indicating the need to request the next subset. The Client shall then post another request, this time specifying the position in the list at which the server should continue with name retrieval. Due to this constraint, the mapping of the ACSI service is actually a sequence of FileDirectory request and response (until moreFollows=FALSE).

If the Filename (element <fileSpecification/>) is not present in the request, or if the wildcard '*' character is used, then the responding server shall return the Filenames present in the root directory (additional files may also be returned). The wildcard '*' character shall not be combined within a string as a wildcard for name extension, but always as a single character and is therefore equivalent to a request where filename is either empty, '/' or '\'.
 IEC 61850-8-2:2018

For servers that have logical devices, 23.1 specifies the root directory. The names of the files returned shall be the filenames of the files present in the LD directory. For servers that have no logical devices (for instance possibly a file server), the root directory shall be a local issue.

Table 21 – Mapping of ACSI GetServerDirectory (FILE)

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <fileDirectory/>		See structure of XML request in Figure 17
<objectClass/> = FILE	<fileSpecification/>	<GraphicString>*/</GraphicString>	Optional
	<continueAfter/>	FileName	Optional (see comment for parameter Reference[0..n])
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <fileDirectory/>		See structure of XML response in Figure 18
Reference[0..n]	<listOfDirectoryEntry/>	Each DirectoryEntry gives the FileName and the attributes of one file	The FileName includes one <GraphicString/> element if its size is less than 64 characters. Otherwise additional <GraphicString/> elements with a maximum size of 64 characters are appended
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	See 8.3.3.4.11	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>3819321</invokeID>
  <ConfirmedServiceRequest>
    <fileDirectory>
      <fileSpecification>
        <GraphicStringMMS>*</GraphicStringMMS>
      </fileSpecification>
    </fileDirectory>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 17 – XML structure of GetServerDirectory-Request (FILE)

```

<confirmed-ResponsePDU>
  <invokeID>3819321</invokeID>
  <ConfirmedServiceResponse>
    <fileDirectory moreFollows="true">
      <listOfDirectoryEntry>
        <DirectoryEntry>
          <fileName>
            <GraphicStringMMS>FileName1</GraphicStringMMS>
          </fileName>
          <fileAttributes>
            <sizeOfFile>383</sizeOfFile>
            <lastModified>201512160532-0500</lastModified>
          </fileAttributes>
        </DirectoryEntry>
        <DirectoryEntry>
          <fileName>
            <GraphicStringMMS>FileName2</GraphicStringMMS>
          </fileName>
          <fileAttributes>
            <sizeOfFile>6271</sizeOfFile>
            <lastModified>20160115092132.2-0500</lastModified>
          </fileAttributes>
        </DirectoryEntry>
      </listOfDirectoryEntry>
    </fileDirectory>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 18 – XML structure of GetServerDirectory-Response (FILE)

10 Association model

10.1 Association relation to communication profiles

The two profiles specified in this SCSM use the two-party association as specified in IEC 61850-7-2 (see Table 22).

Table 22 – Association model versus communication profiles

Communication profile	ACSI association model(s) supported
Client/server	Two party
Time Sync	Two party

10.2 Two party association model for client/server communication profile

10.2.1 Establishment of a secured end-to-end association

The Associate service as described in clause 10.2.2.1 is taking place within the handshake done in parallel for two additional mechanisms: creation of the association context and establishment of an end-to-end secured session. These two mechanisms are defined in IEC 62351-4 and will be therefore not described in the following subclauses.

NOTE The association context part and the end-to-end security elements are omitted within the XML messages described in the following clauses as well as in all the examples of this document.

Prior to establishing an association between two application entities, an underlying transport connection has to be established, i.e. each endpoint shall be connected to its XMPP server. If the two endpoints doesn't belong to the same XMPP domain, the two related XMPP server shall be federated.

Once the association is established, both endpoints (caller and called) shall monitor the status of the communication. In case an endpoint detects a communication outage, a local abort shall be raised up to the IEC 61850 application level:

- If a loss of link is detected by an endpoint at the XMPP client level (e.g. broken TCP connection between XMPP client and XMPP server), then a local Abort shall be raised for all Associations using this link, i.e. all the Associations where this endpoint is involved with the same JID.
- If the XMPP client of an endpoint is informed by the XMPP server that a JID involved in an association is "unavailable" (through the presence mechanism, see 6.4), then a local Abort shall be raised but only for the Associations involving this given JID.

10.2.2 Association services

10.2.2.1 Associate

10.2.2.1.1 General

The request, response+ and response- of an ACSI Associate service shall map respectively to elements <initiate-RequestPDU/>, <initiate-ResponsePDU/> and <initiate-ErrorPDU/>.

Figure 19 and Figure 20 give examples for the structure of the XML messages corresponding to the request and the response. Table 23 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped globally on these request and response messages. Nevertheless the XML elements of these messages does not correspond to any ACSI level parameter. They are instead described in Table 24 and Table 25.

NOTE During this service, the calling and called entities exchange the list of their supported services respectively within the elements <servicesSupportedCalling/> and <servicesSupportedCalled/>. Negotiation of the extended behaviour as mentioned in 8.5 will require that both the request and response include <extendedGetNameList/> as a supported service.

Mapping of response- is detailed in 10.2.2.1.2.

Table 23 – Mapping of ACSI Associate service

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<initiate-RequestPDU>		See structure of XML request in Figure 19
ServerAccessPointReference			JID of the receiving entity is mapped in the attribute "to" of the XMPP <iq/> stanza (see 6.3)
Authentication parameter			Mapped within end-to-end security elements (see Clause 7)
Response+	<initiate-ResponsePDU>		See structure of XML response in Figure 20
AssociationId			
Result			
Response–	<initiate-ErrorPDU>	See 10.2.2.1.2	
ServiceError			

Table 24 – Description of Associate request elements

Initiate Request XML elements	Value/Range	Description
<localDetailCalling/>		Proposed PDU size
<proposedMaxServOutstandingCalling/>	1 or greater	Maximum number of pending requests sent
<proposedMaxServOutstandingCalled/>	1 or greater	Maximum number of pending requests received
<proposedDataStructureNestingLevel/>	c1	Maximum number of "\$" character in a reference
<proposedVersionNumber/>	shall be 1	
<proposedParameterCBB/>		Set of proposed parameters. Shall be consistent with the supported features claimed by the implementation (see detail in 24.2.2.1.3)
<servicesSupportedCalling/>		Set of proposed services. Shall be consistent with the supported features and ACSI services claimed by the implementation (see detail in 24.2.2.1.2)
c1 Shall be five(5) or greater if ACSI logical node model support is declared.		

Table 25 – Description of Associate response elements

Initiate Response XML elements	Value/Range	Description
<localDetailCalled/>		Negotiated PDU size
<negotiatedMaxServOutstandingCalling/>	1 or greater	Maximum number of pending requests sent
<negotiatedMaxServOutstandingCalled/>	1 or greater	Maximum number of pending requests received
<negotiatedDataStructureNestingLevel/>	c1	Maximum number of "\$" character in a reference
<negotiatedVersionNumber/>	shall be 1	
<negotiatedParameterCBB/>		Set of negotiated parameters. Shall be consistent with the supported features claimed by the implementation (see detail in 24.2.2.1.3)
<servicesSupportedCalled/>		Set of negotiated services. Shall be consistent with the supported features and ACS services claimed by the implementation (see detail in 24.2.2.1.2)
c1 Shall be five(5) or greater if ACS logical node model support is declared.		

```

<initiate-RequestPDU>
  <localDetailCalling>65536</localDetailCalling>
  <proposedMaxServOutstandingCalling>5</proposedMaxServOutstandingCalling>
  <proposedMaxServOutstandingCalled>5</proposedMaxServOutstandingCalled>
  <proposedDataStructureNestingLevel>7</proposedDataStructureNestingLevel>
  <mmsInitRequestDetail>
    <proposedVersionNumber>1</proposedVersionNumber>
    <proposedParameterCBB>
      str1 str2 vnam valt vlis
    </proposedParameterCBB>
    <servicesSupportedCalling>
      getNameList informationReport
    </servicesSupportedCalling>
  </mmsInitRequestDetail>
</initiate-RequestPDU>

```

Figure 19 – XML structure of Associate-Request

```

<initiate-ResponsePDU>
  <localDetailCalled>65536</localDetailCalled>
  <negotiatedMaxServOutstandingCalling>5</negotiatedMaxServOutstandingCalling>
  <negotiatedMaxServOutstandingCalled>5</negotiatedMaxServOutstandingCalled>
  <negotiatedDataStructureNestingLevel>7</negotiatedDataStructureNestingLevel>
  <mmsInitResponseDetail>
    <negotiatedVersionNumber>1</negotiatedVersionNumber>
    <negotiatedParameterCBB>
      str1 str2 vnam valt vlis
    </negotiatedParameterCBB>
    <servicesSupportedCalled>
      status getNameList identify read write getVariableAccessAttributes
      defineNamedVariableList getNamedVariableListAttributes deleteNamedVariableList
      obtainFile readJournal getCapabilityList fileOpen fileRead fileClose fileDelete
      fileDirectory conclude cancel extendedGetNameList
    </servicesSupportedCalled>
  </mmsInitResponseDetail>
</initiate-ResponsePDU>

```

Figure 20 – XML structure of Associate-Response

10.2.2.1.2 Response-

An Associate Response- is mapped on an <initiate-ErrorPDU> which contains an <errorClass/> element with the following information:

- In case of parameter mismatches for the intended application where the proposed parameterCBB or serviceSupported does not allow the intended IEC 61850 application to operate properly: the <errorClass/> element shall indicate <initiate/> with an appropriate errorCode value (see Table 26).
- If the Associate requestor is restricted from being allowed to establish an association due to security privileges, the server shall issue an element <errorClass/><access/> with an errorCode = <object-access-denied/>.
- Any other error pertaining to an Associate service shall be mapped to the ACSI ServiceError of failed-due-to-server-constraint.

Table 26 details associate ACSI service error mappings. ACSI ServiceError values that do not appear in the table are not mapped.

Table 26 – Associate ACSI service error mappings

ACSI service error	ServiceError		Comment
	Error class	Error code	
parameter-value-inconsistent	initiate	version-incompatible	
failed-due-to-communications-constraint	initiate	parameter-CBB-insufficient nesting-level-insufficient max-services-outstanding-called-insufficient max-services-outstanding-calling-insufficient max-services-outstanding-calling-insufficient service-CCB-insufficient	
access-violation	access	object-access-denied	Result::=Failure
failed-due-to-server-constraint	Any class	Any unmapped error codes	

10.2.2.1.3 Implementation agreement

- The MMS version number shall be as specified in Table 96 and Table 97.
- The max_mms_pdu_size is defined as the maximum number of octets in an MMSpdu encoded using the negotiated transfer syntax. This size shall apply to all MMSpdu's with the exception of the initiate-Request PDU, initiate-Response PDU, and initiate-Error PDU. The max_mms_pdu_size shall be negotiated during connection initiation using the Local Detail Calling and Local Detail Called parameters of the Associate service.
- The local detail called parameter in the Associate response shall specify the negotiated max_mms_pdu_size for the application association; it is recommended that at least 8192 octets be supported. its value shall be less than or equal to the value of the local detail calling parameter of the request.
- The negotiated max_mms_pdu_size shall be applied as follows:
 - a) Any received MMSpdu whose length is less than or equal to the negotiated max_mms_pdu_size shall be properly parsed and processed.
 - b) An implementation should not send an MMSpdu whose size exceeds the negotiated max_mms_pdu_size. If an implementation sends an MMSpdu that exceeds the negotiated max_mms_pdu_size, then it shall be prepared to receive a reject pdu. Should an implementation receive an MMSpdu that exceeds the negotiated max_mms_pdu_size, it shall either reject the MMSpdu or accept the MMSpdu as if no size violation had occurred and perform the expected processing.

- c) If an implementation is unable to send a service response because the response would exceed the `max_mms_pdu_size`, then it shall return a Service response- with a `<serviceError/>` of `<errorClass/><service/>` with an error code= `<pdu-size/>`.
- d) When rejecting an MMSpdu because it exceeds the negotiated `max_mms_pdu_size`, an implementation shall use a `<rejectPDU/>` of type `<confirmed-requestPDU/>` with a reject code = `<value-out-of-range/>`.

10.2.2.1.4 Versioning and compatibility

During the association establishment, an implementation shall indicate the latest version of the XSD schema it complies with, respectively within the request, response+ and response- of the Associate service. The versioning information is made of three optional attributes: version, revision and release, whose definitions and values shall be consistent with Clause 1.2. These attributes shall be supported by the elements `<initiate-RequestPDU/>`, `<initiate-ResponsePDU/>` and `<initiate-ErrorPDU/>`.

NOTE The default values of the three versioning attributes correspond to the first published version so that these three attributes can be omitted in this first version.

If during the association handshake an endpoint detects that the version of the other communicating endpoint is not compliant with its own capabilities, then the handshake shall be terminated by one of the following means:

- If the issue is detected by the endpoint receiving the Associate request, an `<initiate-ErrorPDU/>` is returned with `<errorClass/><initiate/>` and an `errorCode` = `<version-incompatible/>`.
- If the issue is detected by the endpoint which has initiated the association, it shall simply release it.

Once the association is established, each communicating endpoint is aware of the capabilities of the associated endpoint so that it shall make its best effort to send only information compliant with the version supported by the other peer. However any endpoint shall be prepared to:

- Accept XML messages where a new unknown element has been added at the end of a list of known elements, with the condition that it is not marked with a `mustUnderstand` attribute. In that case, ignore the new element.
- If a known element contains an unknown child element with `mustUnderstand=true` attribute as its last child element, then the complete known parent element shall be ignored but only if additionally the containing element is optional in the XSD of the receiving endpoint. If the parent element is not optional the endpoint must reject the message by sending a `RejectPDU` as defined in 8.3.3.4.6.
- Ignore new attributes of known elements.

NOTE This implies that new attributes may be defined in future versions but only as optional with a default value and additionally this value shall be backward compatible to "not knowing this attribute".

10.2.2.2 Abort

ACSI Abort indications are issued locally by the underlying layers and then have no dedicated representation on the wire. The use cases and reasons for raising Abort indications are detailed in clauses above.

10.2.2.3 Release

10.2.2.3.1 General

The request and response+ of the ACSI Release service shall map to the empty elements `<conclude-RequestPDU/>` and `<conclude-ResponsePDU/>`.

The response- shall map to an element <conclude-ErrorPDU> including an element <errorClass/> (see 10.2.2.3.2).

Table 27 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped globally on these request and response XML messages.

Table 27 – Mapping of ACSI Release service

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<conclude-RequestPDU/>	None	
AssociationId			
Response+	<conclude-ResponsePDU/>	None	
AssociationId			
Result			
Response-		See 10.2.2.3.2	
ServiceError	<conclude-ErrorPDU/>		

10.2.2.3.2 Response-

Table 28 details the mappings of ACSI errors to the possible values of <errorClass/> included in a <conclude-ErrorPDU/>. ACSI ServiceError values that do not appear in the table are not mapped.

Table 28 – Release service error mappings

ACSI service error	ServiceError	
	Error class	Error code
instance-in-use	conclude	further-communication-required
failed-due-to-server-constraint	conclude	other

11 Logical device model

11.1 General

The GetLogicalDeviceDirectory request maps to an XML element <confirmed-RequestPDU/> including an element <getNameList/> where the <objectClass/> element shall have a value of <namedVariable/> (NamedVariable is the MMS mapping for Logical Nodes, see Table 2). Figure 21 gives the structure of the XML message corresponding to the user data (i.e. without the security elements which will wrap the user message in the real message on the wire).

The GetLogicalDeviceDirectory response maps to an XML element <confirmed-ResponsePDU/> including an element <GetNameList/>. Figure 22 gives the structure of the XML message corresponding to the user data.

Table 29 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped within respectively these request and response structures.

The GetLogicalDeviceDirectory service may be constrained by the negotiated PDU size limitation. In that case, the response may return only a subset of references and a flag (moreFollows) indicating the need to request the next subset. The client shall then post

another request, this time specifying the position in the list at which the server should continue with name retrieval. Due to this constraint, the mapping of this ACSI service is actually a sequence of requests/responses, until moreFollows=FALSE in the response.

Table 29 – Mapping of ACSI GetLogicalDeviceDirectory

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNameList/>		See structure of XML request in Figure 21
	<objectClass/>	<namedVariable/>	Fixed value
LDName	<domainSpecific/>	<LDName>	
	<continueAfter/>	<LNReferenceX>	Optional: name of the last reference returned in the previous response in case it contains moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNameList/>		See structure of XML response in Figure 22
LNReference[0..n]	<listOfIdentifier/>	<Identifier>Reference1</Identifier> <Identifier>Reference2</Identifier> ...	The returned identifiers shall be the flattened named variables making up the LNs of the logical device, whose <objectClass/> is <namedVariable/>. 8.2.2 explains how to construct the flattened named variables of a LN. In the present clause the list of identifiers shall be lexicographically ordered. In order to match the definitions of IEC 61850-7-2, this list shall be filtered by the client application in order to keep only the Logical Nodes (without '\$' character). The list may be eventually empty
	moreFollows	true false	
Response-	<confirmed-ErrorPDU/>	See 11.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2329013</invokeID>
  <ConfirmedServiceRequest>
    <getNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL</domainSpecific>
      </objectScope>
    </getNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 21 – XML structure of GetLogicalDeviceDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>2329013</invokeID>
  <ConfirmedServiceResponse>
    <getNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>LLN0</Identifier>
        <Identifier>LLN0$CF</Identifier>
        <Identifier>LLN0$CF$Mod</Identifier>
        <Identifier>LLN0$CF$Mod$ctlModel</Identifier>
        <Identifier>LLN0$DC</Identifier>
        <Identifier>LLN0$DC$NamPlt</Identifier>
        <Identifier>LLN0$DC$NamPlt$vendor</Identifier>
        <Identifier>LLN0$DC$NamPlt$swRev</Identifier>
        <Identifier>LLN0$DC$NamPlt$d</Identifier>
        .....
        <Identifier>CSWI1$ST$Pos$stVal</Identifier>
        <Identifier>CSWI1$ST$Pos$st</Identifier>
        <Identifier>CSWI1$ST$Pos$st</Identifier>
      </listOfIdentifier>
    </getNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 22 – XML structure of GetLogicalDeviceDirectory-Response

If the requested LDName exists but there are no Logical Node inside, then a GetLogicalDeviceDirectory-Response shall be sent with MoreFollows=FALSE and a NULL <listOfIdentifier/>.

The <listOfIdentifier/> returned in the response may contain more references than only Logical Nodes. The identifiers will need to be filtered by the client application in order to keep only the LN references, i.e. the references without '\$' character.

11.2 Response-

A GetLogicalDeviceDirectory response- is mapped on a <confirmed-ErrorPDU/> including a <serviceError/>:

- If the requested LDName contained in the <domainSpecific/> element of the corresponding request does not exist the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the client does not have access privileges for the requested LDName contained in the <domainSpecific/> element of the corresponding request, the server shall return a <serviceError/> with <errorClass/><access/> and errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.

- See also 8.3.3.4.8

11.3 Extended behaviour

The behaviour described in this subclause shall be negotiated during the association process. The goal is that the server application should return only the necessary information strictly required by the client application and not the complete information returned in the <getNameList/> element shown above, thus avoiding filtering at application level.

This specific behaviour is indicated in the request by using the choice <extendedGetNameList/> instead of the <getNameList/> element (see Figure 23). Then the corresponding response message contains only the list of LNName (see Figure 24).

Table 30 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped in case the extended behaviour is selected.

Table 30 – Extended mapping of ACSI GeLogicalDeviceDirectory

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <extendedGetNameList/>		See structure of XML request in Figure 23
	<objectClass/>	<namedVariable/>	Fixed value
LDName	<domainSpecific/>	<LDName>	
	<continueAfter/>	<LNReferenceX>	Optional: name of the last reference returned in the previous response in case it contains moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <extendedGetNameList//>		See structure of XML response in Figure 24
LNReference[0..n]	<listOfIdentifier/>	<Identifier>LNName1</Identifier> <Identifier>LNName2</Identifier> ...	The returned identifiers shall be the names all the LNs. The list may be eventually empty
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Similar to 11.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>5440259</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 23 – XML structure of extended GetLogicalDeviceDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>5440259</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>LLN0</Identifier>
        <Identifier>LPHD1</Identifier>
        <Identifier>GGIO1</Identifier>
        <Identifier>CSWI1</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 24 – XML structure of extended GetLogicalDeviceDirectory-Response

12 Logical Node model

12.1 General

A logical Node is composed of various attributes of different ACSI classes (see Table 31). All these kinds of objects can be retrieved by the service GetLogicalNodeDirectory.

Table 31 – Objectclasses for GetLogicalNodeDirectory service

ACSI class being requested in GetLogicalNodeDirectory	Value of objectClass
DataObject	NamedVariable
DataSet	NamedVariableList
BufferedReportControlBlock	NamedVariable
UnbufferedReportControlBlock	NamedVariable
LogControlBlock	NamedVariable
SettingGroupControlBlock	NamedVariable
Log	Journal
GooseControlBlock	NamedVariable
GSSEControlBlock (deprecated)	NamedVariable
MulticastSampledValueControlBlock	NamedVariable
UnicastSampledValueControlBlock (deprecated)	NamedVariable

12.2 GetLogicalNodeDirectory

12.2.1 General

The GetLogicalNodeDirectory request maps to an XML element <confirmed-RequestPDU/> including an element <getNameList/> where the <objectClass/> element shall have a value as specified in Table 31. Figure 25 gives the structure of the XML message corresponding to the user data (i.e. without the security elements which will wrap the user message in the real message on the wire).

The GetLogicalNodeDirectory response maps to an XML element <confirmed-ResponsePDU/> including an element <GetNameList/>. Figure 26 gives the structure of the XML message corresponding to the user data.

Table 32 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped within respectively these request and response structures.

The GetLogicalNodeDirectory service may be constrained by the negotiated PDU size limitation. In that case, the response may return only a subset of references and a flag (moreFollows) indicating the need to request the next subset. The client shall then post another request, this time specifying the position in the list at which the server should continue with name retrieval. Due to this constraint, the mapping of this ACSI service is actually a sequence of requests/responses, until moreFollows=FALSE in the response.

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

Table 32 – Mapping of ACSI GetLogicalNodeDirectory

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNameList/>		See structure of XML request in Figure 25
LNReference	<domainSpecific/>	<LDName>	Name of parent logical device
ACSIClass	<objectClass/>	See Table 31	
	<continueAfter/>	<ReferenceX>	Optional: name of the last reference returned in the previous response in case it contains moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNameList/>		See structure of XML response in Figure 26
InstanceName[0..n]	<listOfIdentifier/>	<Identifier>Reference1</Identifier> <Identifier>Reference2</Identifier> ...	The returned identifiers shall be the flattened named variables making up the LNs of the logical device, whose <objectClass/> is the one specified in input. 8.2.2 explains how to construct the flattened named variables of a LN. In the present clause the list of identifiers shall be lexicographically ordered. In order to match the definitions of IEC 61850-7-2, this list shall be filtered by the client application in order to keep only the highest level objects of the only one requested logical node. The list may be eventually empty.
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	See 12.2.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceRequest>
    <getNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL</domainSpecific>
      </objectScope>
    </getNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 25 – XML structure of GetLogicalNodeDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceResponse>
    <getNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>LLN0</Identifier>
        <Identifier>LLN0$CF</Identifier>
        <Identifier>LLN0$CF$Mod</Identifier>
        <Identifier>LLN0$CF$Mod$ctlModel</Identifier>
        <Identifier>LLN0$DC</Identifier>
        <Identifier>LLN0$DC$NamPlt</Identifier>
        <Identifier>LLN0$DC$NamPlt$vendor</Identifier>
        <Identifier>LLN0$DC$NamPlt$swRev</Identifier>
        <Identifier>LLN0$DC$NamPlt$d</Identifier>
        .....
        <Identifier>CSWI1$ST$Pos$stVal</Identifier>
        <Identifier>CSWI1$ST$Pos$q</Identifier>
        <Identifier>CSWI1$ST$Pos$st</Identifier>
      </listOfIdentifier>
    </getNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 26 – XML structure of GetLogicalNodeDirectory-Response

The <listOfIdentifier/> returned in the response may contain more references than only the requested objects of the requested logical node. The identifiers will need to be filtered by the client application in order to keep only the objects of the requested logical node. If the requested ACSI class is DataObject, only the highest level names must be kept.

When there is no object matching the GetLogicalNodeDirectory request, a Response+ shall be returned with the <listOfIdentifier/> being NULL and moreFollows set to FALSE.

12.2.2 Response-

A GetLogicalNodeDirectory response- is mapped on a <confirmed-ErrorPDU/> including a <serviceError/>:

- If the requested LDName contained in the <domainSpecific/> element of the corresponding request does not exist the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the client does not have access privileges for the requested LDName contained in the <domainSpecific/> element of the corresponding request, the server shall return a <serviceError/> with <errorClass/><access/> and errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.
- See also 8.3.3.4.8

12.2.3 Extended behaviour

The behaviour described in this subclause shall be negotiated during the association process. The goal is that the server application should return only the necessary information strictly required by the client application and not the complete information returned in the <getNameList/> element shown above, thus avoiding filtering at application level.

This specific behaviour is indicated in the request by using the choice <extendedGetNameList/> instead of the <getNameList/> element.

The mapping of this ACSI service requires a two step process. In a first request (step 1), the parameter LNReference shall be <LDName/LNName> (see Figure 27) and the response shall return the name of all the Functional Constraints present in the specified LN (see Figure 28).

Then for each returned FC a second request (step 2) shall be issued with LNReference=<LDName/LNName\$FC> (see Figure 29) and the response message shall contain the names of the first level objects located in the specified LN and having at least one sub-component of the specified FC (see Figure 30).

Table 33 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped in case the extended behaviour is selected.

Table 33 – Extended mapping of ACSI GeLogicalNodeDirectory

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <extendedGetNameList/>		See structure of XML request in Figure 27 for step 1 and Figure 29 for step 2
LNReference	<domainSpecific/>	<LDName/LNName> or <LDName/LNName\$FC>	Step 1: one request with LNReference Step 2: one request for each FC
ACSIClass	<objectClass/>	See Table 31	
	<continueAfter/>	<ReferenceX>	Optional: name of the last reference returned in the previous response in case it contains moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <extendedGetNameList//>		See structure of XML response in Figure 28 for step 1 and Figure 30 for step 2
InstanceName [0..n]	<listOfIdentifier/>	<Identifier>Name1</Identifier> <Identifier>Name2</Identifier> ...	The returned identifiers shall be: Step 1: the names of the present FCs for the specified objectClass Step 2: the names of the first level objects whose objectClass is the one specified in input and having at least one sub-component of the specified FC The list is lexicographically ordered. It may be eventually empty
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Similar to 12.2.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL/LLN0</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 27 – XML structure of extended GetLogicalNodeDirectory-Request (step 1)

```

<confirmed-ResponsePDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>CF</Identifier>
        <Identifier>DC</Identifier>
        <Identifier>EX</Identifier>
        <Identifier>ST</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 28 – XML structure of extended GetLogicalNodeDirectory-Response (step 1)

```

<confirmed-RequestPDU>
  <invokeID>2329058</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass>
          <namedVariable/>
        </objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL/LLN0$ST</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 29 – XML structure of extended GetLogicalNodeDirectory-Request (step 2)

```

<confirmed-ResponsePDU>
  <invokeID>2329058</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>Beh</Identifier>
        <Identifier>Health</Identifier>
        <Identifier>Loc</Identifier>
        <Identifier>Mod</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 30 – XML structure of extended GetLogicalNodeDirectory-Response (step 2)

12.3 GetAllDataValues

12.3.1 General

The ACSI request shall map to an XML element <confirmed-RequestPDU/> including a <read/> element. In any case, the element <variableAccessSpecification/> shall include one <variableSpecification/> for the mapping of the ACSI LNReference parameter as per 8.2.3.

However, it is recommended that the ACSI request contains a functional constraint parameter (see example in Figure 31); when a functional constraint is specified, the element <variableAccessSpecification/> in the request shall be a <listOfVariable/> constrained to a single <variableSpecification/> and shall also specify an <alternateAccess/> using the <unnamed/> choice. Within element <unnamed/>, the <accessSelection/> element shall include the <component/> choice with the value of the functional constraint being specified. Any other access mechanism is out of scope of this subclause.

NOTE The access to all values of a particular (without functional constraint SV, CO, and without control blocks) logical node can also be obtained through the use of a single named variable, with no alternate access. The name of the variable would be <LogicalNodeName>.

The ACSI GetAllDataValues response+ maps to a <confirmed-ResponsePDU/> including an element <read/> with one <success> element returning data as a <structure/> element containing one <structure/> per DataObject and possibly nested <structure/> elements for nested sub-DO or sub-DA (see 8.4).

The ACSI GetAllDataValues response+ parameter of DataAttributeReference is not mapped nor supported within this SCSM.

Table 34 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped within respectively the request and response structures.

Table 34 – Mapping of ACSI GetAllDataValues

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		See structure of XML request in Figure 31
LNReference	<variableSpecification/>	If domain-specific, domainId=<LDName> and itemId=<LNName>	See other cases in 8.2.3
FunctionalConstraint[0..1]	<alternateAccess/>	see example in Figure 31	Optional. If exists, the alternateAccess element shall be present within the variableAccessSpecification
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		See structure of XML response in Figure 32
LNReference	<variableSpecification/>		Idem Request but optional in Response
DataAttributeReference[1..n]	N/A	N/A	
DataAttributeValue[1..n]	<listOfAccessResult/>	Structure of nested structures	One nested level per additional sub-DO or sub-DA (see 8.4)
Response–	<confirmed-RequestPDU/> <ConfirmedServiceResponse/> <read/> or <confirmed-ErrorPDU/>	See 8.3.3.4.9	

```

<confirmed-RequestPDU>
  <invokeID>121013</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
            <alternateAccess>
              <CHOICE>
                <unnamed>
                  <selectAlternateAccess>
                    <accessSelection>
                      <component>ST</component>
                    </accessSelection>
                    <alternateAccess/>
                  </selectAlternateAccess>
                </unnamed>
              </CHOICE>
            </alternateAccess>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 31 – XML structure of GetAllDataValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>121013</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <structure>
                  <Data>
                    <integer>1</integer>
                  </Data>
                  <Data>
                    <bit-string>0000000000000</bit-string>
                  </Data>
                  <Data>
                    <utc-time>017215E30000001F</utc-time>
                  </Data>
                </structure>
              </Data>
            </Data>
          </Data>
          [...]
          <Data>
            <structure>
              <Data>
                <bit-string>10</bit-string>
              </Data>
              <Data>
                <bit-string>0000000000000</bit-string>
              </Data>
              <Data>
                <utc-time>017215E30000001F</utc-time>
              </Data>
            </structure>
          </Data>
        </structure>
      </AccessResult>
    </listOfAccessResult>
  </read>
</ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 32 – XML structure of GetAllDataValues-Response

13 DataObject, DataAttribute, SubDataAttribute model

13.1 General

See 8.2.4 and 8.2.5 for information about how to map FCD and FCDA (i.e. respectively Functionally Constrained Data and Functionally Constrained Data Attribute) to <variableAccessSpecification/> elements.

13.2 GetDataValues

The ACSI GetDataValues request shall map to an XML element <confirmed-RequestPDU/> including a <read/> element where the <variableAccessSpecification/> element includes one <VariableSpecification/> as defined in 8.2.4 and 8.2.5.

The ACSI GetDataValues response+ maps to a <confirmed-ResponsePDU/> including an element <read/> with one <success/> element returning data as one basic type element if the Reference parameter is a final DA (e.g. see Figure 34), or one <structure/> element including possibly and recursively nested <structure/> elements in case of nested DO or DA (see 8.4).

The mapping of the ACSI service parameters shall be as defined in Table 35.

Table 35 – Mapping of GetDataValues service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		See structure of XML request in Figure 33
Reference	<variableSpecification/>	See 8.2.4 and 8.2.5	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		See structure of XML response in Figure 34
DataAttributeValue[1..n]	<listOfAccessResult/>	One basic type value or one <structure/> with possibly recursively nested <structure/> inside (see 8.4)	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/> or <confirmed-ErrorPDU/>	See 8.3.3.4.9	

```

<confirmed-RequestPDU>
  <invokeID>2676121</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>IEDNameLDINst</domainId>
                  <itemId>LLN0$ST$Beh$stVal</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 33 – XML structure of GetDataValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>2676121</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <integer>1</integer>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 34 – XML structure of GetDataValues-Response

13.3 SetDataValues

The ACSI SetDataValues request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <write/> where the <variableAccessSpecification/> element includes one <VariableSpecification/> as defined in 8.2.4 and 8.2.5, and the <listOfData/> element includes one element <Data/> including itself either one basic type element (e.g. see Figure 35) if the Reference parameter is a final DA or one <structure/> element including possibly and recursively nested <structure/> elements in case the Reference has nested DO or DA (see 8.4).

The ACSI GetDataValues response+ maps to a <confirmed-ResponsePDU/> including an element <write/> with one <success/> empty element.

The mapping of the ACSI service parameters shall be as defined in Table 36.

Table 36 – Mapping of SetDataValues service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		See structure of XML request in Figure 35
Reference	<variableSpecification/>	See 8.2.4 and 8.2.5	
DataAttributeValue[1..n]	<listOfData/><Data/>	One basic type value or one <structure/> with possibly recursively nested <structure/> inside (see 8.4)	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/>		See structure of XML response in Figure 36
	<success/>	Empty	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/> or <rejectPDU/>	See 8.3.3.4.10	

```

<confirmed-RequestPDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>TEST</domainId>
                  <itemId>GGIO1$CF$Mod$ctlModel</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <integer>1</integer>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 35 – XML structure of SetDataValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 36 – XML structure of SetDataValues-Response

13.4 GetDataDirectory

13.4.1 General

The ACSI GetDataDirectory shall map to a sequence of requests and responses.

The requests shall map to an XML element <confirmed-RequestPDU/> including an element <getVariableAccessAttributes/> where the Reference parameter is mapped on a <name/> element including a <domain-specific/> element (see example in Figure 37). Within this sequence of requests, the <itemId/> element shall go through the list of Functional Constraints established by the client.

Each ACSI GetDataDirectory response+ maps to a <confirmed-ResponsePDU/> including an element <getVariableAccessAttributes/> including itself an element <typeSpecification> which gives recursively the name and type of all the components, level by level, of all nested DO or DA.

At the IEC 61850 application level the information returned in the response shall be filtered in order to keep only the names of the components on the level below the reference, without any type.

The mapping of the ACSI service parameters shall be as defined in Table 37.

Table 37 – Mapping of GetDataDirectory service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getVariableAccessAttributes/>		One request and one response for each FC. See structure of XML request in Figure 37
DataObjectReference	<name/>	Same structure than element <name/> in 8.2.4	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getVariableAccessAttributes/>		See structure of XML response in Figure 38 NOTE mmsDeletable is always false
SubDataObjectName[0..n] or DataAttributeName[1..n]	<typeSpecification>	See example in Figure 38	Filtered at the application level to keep only names of components one level below
Response–	<confirmed-ErrorPDU/>	See 13.4.2	

```

<confirmed-RequestPDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceRequest>
    <getVariableAccessAttributes>
      <name>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId>LLN0$STSMod</itemId>
        </domain-specific>
      </getVariableAccessAttributes>
    </ConfirmedServiceRequest>
  </confirmed-RequestPDU>

```

Figure 37 – XML structure of GetDataDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceResponse>
    <getVariableAccessAttributes>
      <mmsDeletable>
        false
      </mmsDeletable>
      <typeSpecification>
        <structure packed="true">
          <components>
            <SEQUENCE>
              <componentName>stVal</componentName>
              <componentType>
                <integer>1</integer>
              </componentType>
            </SEQUENCE>
            <SEQUENCE>
              <componentName>q</componentName>
              <componentType>
                <bit-string>13</bit-string>
              </componentType>
            </SEQUENCE>
            <SEQUENCE>
              <componentName>t</componentName>
              <componentType>
                <utc-time/>
              </componentType>
            </SEQUENCE>
          </components>
        </structure>
      </typeSpecification>
    </getVariableAccessAttributes>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 38 – XML structure of GetDataDirectory-Response

13.4.2 Response-

The ACSI GetDataDirectory Response- maps on a <confirmed-ErrorPDU/> including a <serviceError/>.

- If the <domainId/> or the <idemId/> specified in the Request does not exist, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the client has no access privilege to the <domainId/> specified in the Request, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the request would cause a Response+ that exceeds the negotiated PDU size, a <confirmed-ErrorPDU/> shall be returned and the <serviceError/> shall be <errorClass/><service/> with an errorCode=<pdu-size/>. This shall be mapped to the ACSI ServiceError of failed-due-to-communications-constraint.

See Table 38 for GetDataDirectory service error mappings.

Table 38 – GetDataDirectory service error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	service	pdu-size	

13.4.3 Extended behaviour

13.4.3.1 General

The behaviour described in this subclause shall be negotiated during the association process. The goal is that the server application should return only the necessary information strictly required by the client application and not the complete information returned in the <getVariableAccessAttributes/> element shown above, thus avoiding filtering at application level.

For this extended behaviour, the ACSI GetDataDirectory request shall map also to a sequence of requests. Each request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <extendedGetNameList/> (see example in Figure 39) and each response+ shall be mapped to an XML element <confirmed-ResponsePDU/> including an element <extendedGetNameList/> (see example in Figure 40). Within this sequence of requests, the <domainSpecific/> element shall go through the list of Functional Constraints established by the client.

Note that the choice <extendedGetNameList/> shall be used instead of the <getNameList/> element within the elements <confirmed-RequestPDU/> and <confirmed-ResponsePDU/>. Then the corresponding response message contains only the list of components one level below the requested reference (see Figure 40).

With this extended behaviour, the GetDataDirectory service may be constrained by the negotiated PDU size limitation. In that case, the response may return only a subset of names and a flag (moreFollows) indicating the need to request the next subset. The client shall then post another request, this time specifying the position in the list at which the server should continue with name retrieval. Due to this constraint, the mapping of this ACSI service is actually a sequence of requests/responses, until moreFollows=FALSE in the response.

Table 39 explains how the request and response parameters defined in IEC 61850-7-2 shall be mapped in case the extended behaviour is selected.

Table 39 – Extended mapping of ACSI GetDataDirectory

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <extendedGetNameList/>		One request and one response for each FC. See structure of XML request in Figure 39
	<objectClass/>	<namedVariable/>	
DataObjectReference	<domainSpecific/>	<LDName/Reference>	Reference is the name of a flattened named variable as described in 8.2.2
	<continueAfter/>	<LNReferenceX>	Optional: name of the last reference returned in the previous response in case it contains moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <extendedGetNameList/>		See structure of XML response in Figure 40
SubDataObjectName[0..n] or DataAttributeName[1..n]	<listOfIdentifier/>	<Identifier>Name1</Identifier> <Identifier>Name2</Identifier> ...	Names of components one level below the requested reference. The list may be eventually empty
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	See 13.4.3.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL/LLN0$ST$Mod</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 39 – XML structure of extended GetDataDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>stVal</Identifier>
        <Identifier>q</Identifier>
        <Identifier>t</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 40 – XML structure of extended GetDataDirectory-Response

13.4.3.2 Response-

The extended GeDataDirectory response- is mapped on a <confirmed-ErrorPDU/> including a <serviceError/>:

- If the requested LDName contained in the <domainSpecific/> element of the corresponding request does not exist the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the client does not have access privileges for the requested LDName contained in the <domainSpecific/> element of the corresponding request, the server shall return a <serviceError/> with <errorClass/><access/> and errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.
- See also 8.3.3.4.8.

13.5 GetDataDefinition

The mapping of this service shall be the same as the default mapping of GetDataDirectory at the XML message level (see 13.4.1 and 13.4.2). However, no filtering shall be applied at the IEC 61850 application level so that names and types of all the components are returned whatever is their level.

14 Data set class model

14.1 General

There are several types of data sets specified in IEC 61850-7-2, either persistent or non-persistent. In the following services, the parameters containing a data set reference are all mapped on a <variableAccessSpecification/> including a <variableListName/>element. However, depending on the type of data sets, different choices shall be used within this <variableListName/>element:

- Persistent data set within a Logical Device: choice= <domain-specific/> (see Figure 41)
- Persistent data set outside any Logical Device: choice=<vmd-specific/> (see Figure 42)
- Non persistent data set: choice=<aa-specific/> (see Figure 43)

```

<domain-specific>
  <domainId>CONTROL</domainId>
  <itemId>GGIO1$myDS</itemId>
</domain-specific>

```

Figure 41 – Mapping of reference to persistent data set within logical device

<vmd-specific>myPersistentDS</vmd-specific>

Figure 42 – Mapping of reference to persistent data set outside logical device

<aa-specific>@myNonPersistentDS</aa-specific>

Figure 43 – Mapping of reference to non-persistent data set

14.2 GetDataSetValues

14.2.1 General

The ACSI GetDataSetValues request shall be mapped to an XML element <confirmed-RequestPDU/> including a <read/> element (see example in Figure 44).

The ACSI GetDataSetValues response+ shall be mapped to an XML element <confirmed-ResponsePDU/> including an element <read/> (see example in Figure 45).

The mapping of the ACSI service parameters shall be as defined in Table 40.

Table 40 – Mapping of GetDataSetValues service parameters

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		See structure of XML request in Figure 44
	specificationWithResult	true or false	
DataSetReference	<DomainId/>	<LDName>	Case of a persistent data set within a Logical Device. For other cases see 14.1
	<ItemId/>	<LNName>\${DSName}	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		See structure of XML response in Figure 45
DataSetReference	<variableAccessSpecification/>		Optional. Shall be constrained by the user request
DataAttributeValue[1..n]	<listOfAccessResult/>		Includes a sequence of <success/> and <failure/> elements
	<success/>	One basic type value or one <structure/> with possibly recursively nested <structure/> inside (see 8.4)	
	<failure/>	One enumerated value as defined by XML complexType DataAccessError	

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/> or <confirmed-ErrorPDU/>	See 14.2.2	

The Response+ shall return a <listOfAccessResults/>. The values contained within this list shall indicate the success or failure. This means that some AccessResults may return <success/> while others return <failure/>.

```

<confirmed-RequestPDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <variableListName>
          <domain-specific>
            <domainId>CONTROL</domainId>
            <itemId>GGIO1$myNewDS</itemId>
          </domain-specific>
        </variableListName>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 44 – XML structure of GetDataSetValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <bit-string>0000000000000</bit-string>
              </Data>
              <Data>
                <utc-time>017215E00000001F</utc-time>
              </Data>
            </structure>
          </success>
        </AccessResult>
        <AccessResult>
          <failure>
            <temporarily-unavailable/>
          </failure>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 45 – XML structure of GetDataSetValues-Response

14.2.2 Response–

Errors may be returned as a Response+ or as a <confirmed-ErrorPDU/> including a <serviceError/>.

This clause, together with 8.3.3.4.9, details the errors that shall be returned.

The following processing shall occur prior to processing per 8.3.3.4.9.

- If the request specifies a Logical Device (i.e. <domainId>) for which the client does not have access privileges, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the data set does not exist, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- A request whose response encoding exceeds the negotiated PDU size, shall return a <serviceError/> with an <errorClass/>=<service/> with an errorCode=<pdu-size/>.

All other errors/processing shall be per 8.3.3.4.9.

See Table 41 for GetDataSetValues error mappings.

Table 41 – GetDataSetValues error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	service	pdu-size	
NOTE For other error codes, see 8.3.3.4.9.			

14.3 SetDataSetValues

14.3.1 General

The ACSI SetDataSetValues request shall be mapped to an XML element <confirmed-RequestPDU/> including a <write/> element (see example in Figure 46). The <listOfData/> element shall include one <Data/> element for each DataSet member reference. Each <Data/> element includes either one basic type element if the DataSet member reference is a final DA or one <structure/> element including possibly and recursively nested <structure/> elements in case the DataSet member reference has nested DO or DA (see 8.4).

The ACSI SetDataSetValues response+ shall be mapped to an XML element <confirmed-ResponsePDU/> including an element <write/> (see example in Figure 47).

The mapping of the ACSI service parameters shall be as defined in Table 42.

Table 42 – Mapping of SetDataSetValues service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest> <write>		See structure of XML request in Figure 46
DataSetReference	<DomainId/>	<LDName>	Case of a persistent data set within a Logical Device. For other cases see 14.1
	<IdemId/>	<LNName>\${DSName}	
DataAttributeValue[1..n]	<listOfData/>	List of </Data> elements including one basic type value or one <structure/> with possibly recursively nested <structure/> inside (see 8.4)	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse> <write>		See structure of XML response in Figure 47
Result	<success/>		

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse> <write> or <confirmed-ErrorPDU/> or <rejectPDU/>	See 14.3.2	

```

<confirmed-RequestPDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <variableListName>
          <domain-specific>
            <domainId>CONTROL</domainId>
            <itemId>GGIO1$myDS</itemId>
          </domain-specific>
        </variableListName>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <integer>1</integer>
        </Data>
        <Data>
          <integer>2</integer>
        </Data>
        <Data>
          <integer>3</integer>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>
    
```

Figure 46 – XML structure of SetDataSetValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>
    
```

Figure 47 – XML structure of SetDataSetValues-Response

14.3.2 Response–

Errors may be returned as a Response+, a <rejectPDU/> element or as a <confirmed-ErrorPDU/> including a <serviceError/>.

This clause, together with 8.3.3.4.10, details the errors that shall be returned.

The following processing shall occur prior to processing per 8.3.3.4.10.

- A SetDataSetValues request that exceeds the negotiated PDU size shall cause a <rejectPDU/> with a rejectPDUType=<confirmed-requestPDU/> and a rejectCode=<value-out-of-range/>. This shall be mapped to the ACSI ServiceError of “failed-due-to-communications-constraint”.
- If the request specifies a Logical Device (i.e. <domainId>) for which the client does not have access privileges, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the data set does not exist, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.

All other errors/processing shall be per 8.3.3.4.10.

See Table 43 for SetDataSetValues error mappings.

Table 43 – SetDataSetValues error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
ACSI service error	Reject		Comments
	Reject PDU Type	Reject Code	
failed-due-to-communications-constraint	Confirmed-requestPDU	Value-out-of-range	
NOTE For other error codes, see 8.3.3.4.10.			

14.4 CreateDataSet

14.4.1 General

The ACSI CreateDataSet request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <defineNamedVariableList/> (see example in Figure 48).

The ACSI CreateDataSet response+ shall be mapped to an XML element <confirmed-ResponsePDU/> including an element <defineNamedVariableList/> (see example in Figure 49).

The mapping of the ACSI service parameters shall be as defined in Table 44 and the service error mapping shall be as defined in 14.4.2.

Table 44 – Mapping of CreateDataSet service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <defineNamedVariableList/>		See structure of XML request in Figure 48
DataSetReference	<domainId/>	<LDName>	Case of a persistent data set within a Logical Device. For other cases see 14.1
	<idemId/>	<LNName>\${DSName}	
DSMemberRef[1..n]	<listOfVariable/>		List of FCD or FCDA. Each FCD or FCDA is mapped as described in 8.2.4 and 8.2.5
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <defineNamedVariableList/>		Only one empty element: <defineNamedVariableList/>. See structure of XML response in Figure 49
Response–	<confirmed-ErrorPDU/>	See 14.4.2	

```

<confirmed-RequestPDU>
  <invokeID>36113013</invokeID>
  <ConfirmedServiceRequest>
    <DefineNamedVariableList>
      <variableListName>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId>GGIO1$myNewDS</itemId>
        </domain-specific>
      </variableListName>
      <listOfVariable>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$ST$Ind1</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$ST$Ind3</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
      </listOfVariable>
    </DefineNamedVariableList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 48 – XML structure of CreateDataSet-Request

```

<confirmed-ResponsePDU>
  <invokeID>36113013</invokeID>
  <ConfirmedServiceResponse>
    <DefineNamedVariableList/>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 49 – XML structure of CreateDataSet-Response

14.4.2 Response–

The CreateDataSet response- is mapped on a <confirmed-ErrorPDU/> including a <serviceError/> element:

- If the CreateDataSet request specifies a Logical Device (i.e. <domainId>) which does not exist, the <serviceError/> within the <confirmed-ErrorPDU/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the request specifies a Logical Device (i.e. <domainId>) for which the client does not have access privileges, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If one of the <variableSpecification/> included in the element <listOfVariable/> of the request specifies a non-existent object, the <serviceError/> within the <confirmed-ErrorPDU/> shall be <errorClass/><definition/> with an errorCode=<object-undefined/>.
- If the object name contained in the element <variableListName/> already exists, the <serviceError/> within the <confirmed-ErrorPDU/> shall be <errorClass/><definition/> with an errorCode=<object-exists/>.
- If the number of <variableSpecification/> in <listOfVariable/> exceeds the maximum allowed, the <serviceError/> within the <confirmed-ErrorPDU/> shall be <errorClass/><resource/> with an errorCode=<capability-unavailable/>. This shall be mapped to the ACSI ServiceError of “failed-due-to-server-constraint”.
- A CreateDataSet request that exceeds the negotiated PDU size shall cause a <rejectPDU/> with a rejectPDUType=<confirmed-requestPDU/> and a rejectCode=<value-out-of-range/>. This shall be mapped to the ACSI ServiceError of “failed-due-to-communications-constraint”.
- A CreateDataSet request that specifies a Logical Device (i.e. <domainId>) and specifies a <variableListName/> which does not contain the name of an existing Logical Node, shall cause a <confirmed-ErrorPDU/> with a <serviceError/><errorClass/><access/> and an errorCode=<object-non-existent/>. This shall be mapped to the ACSI ServiceError of “parameter-value-inconsistent”.

See Table 45 for CreateDataSet service error mappings.

Table 45 – CreateDataSet service error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
instance-in-use	definition	object-exists	
parameter-value-inconsistent	definition	object-undefined	
failed-due-to-server-constraint	resource	capability-unavailable	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	
ACSI service error	Reject		Comments
	Reject PDU Type	Reject Code	
failed-due-to-communications-constraint	Confirmed-requestPDU	Value-out-of-range	

14.5 DeleteDataSet

14.5.1 General

The ACSI DeleteDataSet request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <deleteNamedVariableList/> (see example in Figure 50).

The ACSI DeleteDataSet response+ shall be mapped to an XML element <confirmed-ResponsePDU/> including an element <deleteNamedVariableList/> (see example in Figure 51).

The mapping of the ACSI service parameters shall be as defined in Table 46.

Table 46 – Mapping of DeleteDataSet service parameters

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest /> <deleteNamedVariableList />		See structure of XML request in Figure 50
	scopeOfDelete	specific	Attribute with fixed value
DataSetReference	<domainId/>	<LDName>	Case of a persistent data set within a Logical Device. For other cases see 14.1
	<itemId/>	<LNName>\${DSName}	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse /> <deleteNamedVariableList />		See structure of XML response in Figure 51
	<numberMatched/>	1 (if success)	For values in case of error, see 14.5.2
	<numberDeleted/>	1 (if success)	

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Response–	<code><confirmed-ResponsePDU/></code> <code><ConfirmedServiceResponse/></code> <code><deleteNamedVariableList/></code> or <code><confirmed-ErrorPDU/></code>	See 14.5.2	

```

<confirmed-RequestPDU>
  <invokeID>1322432</invokeID>
  <ConfirmedServiceRequest>
    <deleteNamedVariableList scopeOfDelete="specific">
      <listOfVariableListName>
        <ObjectName>
          <domain-specific>
            <domainId>CONTROL</domainId>
            <itemId>GGIO1$myDS</itemId>
          </domain-specific>
        </ObjectName>
      </listOfVariableListName>
    </deleteNamedVariableList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 50 – XML structure of DeleteDataSet-Request

```

<confirmed-ResponsePDU>
  <invokeID>1322432</invokeID>
  <ConfirmedServiceResponse>
    <deleteNamedVariableList>
      <numberMatched>1</numberMatched>
      <numberDeleted>1</numberDeleted>
    </deleteNamedVariableList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 51 – XML structure of DeleteDataSet-Response

14.5.2 Errors

The ACSI DeleteDataSet service may return errors as a Response+ or as a <confirmed-ErrorPDU/> including a <serviceError/> or as a <confirmed-ErrorPDU/> including a <serviceError/> with a <serviceSpecificInformation/> element.

- Requests specifying an element <scopeOfDelete/> with values <aa-specific/>, <domain/> or <vmd/> are out of scope of this SCSM.
- If the specified data set does not exist a Response+ with <numberMatched/>=0 shall be returned
- If the specified data set exists but is not deletable, a Response+ with <numberMatched/>=1 and <numberDeleted/>=0 shall be returned.
- If the specified data set is deletable but cannot be deleted, because for example it is used in a Report Control Block, a Response– (i.e. <confirmed-ErrorPDU/> including a <serviceSpecificInformation/>) with the number of <deleteNamedVariableList/> set to 0 shall be returned.

See Table 47 for DeleteDataSet service error mappings.

Table 47 – DeleteDataSet service error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	
ACSI service error	Response+		Comments
	Parameter	value	
instance-not-available	numberMatched	0	
	numberDeleted	0	
failed-due-to-server-constraint	numberMatched	1	
	numberDeleted	0	
ACSI service error	Response–		Comments
	Parameter	value	
failed-due-to-server-constraint	numberDeleted	0	Constraint to Error Type: Error class = Service error code = object- constraint-conflict

14.6 GetDataSetDirectory

14.6.1 General

The ACSI GetDataSetDirectory request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <getNamedVariableListAttributes/> (see example in Figure 52).

The ACSI GetDataSetDirectory response+ shall be mapped to an XML element <confirmed-ResponsePDU/> including an element <getNamedVariableListAttributes/> (see example in Figure 53).

The mapping of the ACSI service parameters shall be as defined in Table 48.

Table 48 – Mapping of GetDataSetDirectory service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNamedVariableListAttributes/>		See structure of XML request in Figure 52
DataSetReference	<domainId/>	<LDName>	Case of a persistent data set within a Logical Device. For other cases see 14.1
	<itemId/>	<LNName>\$<DSName>	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNamedVariableListAttributes/>		See structure of XML response in Figure 53
DSMemberRef[1..n]	<listOfVariable/>		List of FCD or FCDA. Each FCD or FCDA is mapped as described in 8.2.4 and 8.2.5
Response–	<confirmed-ErrorPDU/>	See 14.6.2	

```

<confirmed-RequestPDU>
  <invokeID>1322763</invokeID>
  <ConfirmedServiceRequest>
    <getNamedVariableListAttributes>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$myDS</itemId>
      </domain-specific>
    </getNamedVariableListAttributes>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 52 – XML structure of GetDataSetDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>1322763</invokeID>
  <ConfirmedServiceResponse>
    <getNamedVariableListAttributes>
      <mmsDeletable>true</mmsDeletable>
      <listOfVariable>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CF$SPCSO1$ctlModel</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CF$SPCSO2$ctlModel</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CF$DPCSO1$ctlModel</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
      </listOfVariable>
    </getNamedVariableListAttributes>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 53 – XML structure of GetDataSetDirectory-Response

14.6.2 Response-

The GetDataSetDirectory response- is mapped on a <confirmed-ErrorPDU/> including a <serviceError/>.

- If the GetDataSetDirectory request specifies a Logical Device (i.e. <domainId>) which does not exist, the <serviceError/> within the <confirmed-ErrorPDU/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the data set does not exist, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-non-existent/>.
- If the request specifies a Logical Device (i.e. <domainId>) for which the client does not have access privileges, the <serviceError/> shall be <errorClass/><access/> with an errorCode=<object-access-denied/> or <errorClass/><access/> with an errorCode=<object-non-existent/>.
- A request whose response encoding exceeds the negotiated PDU size shall return <serviceError/><errorClass/><service/> with an errorCode=<pdu-size/>. This shall be mapped to the ACSI ServiceError of failed-due-to-communications-constraint.

See Table 49 for GetDataSetDirectory service error mappings.

Table 49 – GetDataSetDirectory service error mappings

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	service	pdu-size	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	

15 ServiceTracking model

The service tracking model is based on specific common data classes (CDC) defined in IEC 61850-7-3. The role of these CDCs is to store the parameters used by the ACSI services in order to log or report them each time a service is called. The attributes of these CDCs are using data types already mapped in 8.3.1 so that this clause will not describe the detailed mapping for all the CDCs. The only information presented here is:

- The possible values for attributes defined as ENUMERATED by the common data class CST: serviceType (see Table 50) and errorCode (Table 51).
- The mapping of the CDCs whose definition is modified in this SCSM compared to IEC 61850-7-3: class LTS (see Table 52) and GTS (see Table 53).

Table 50 – Mapping of ACSI ServiceType values

ACSI Service	Value
Unknown	0
Associate	1
Abort	2
Release	3
GetServerDirectory	4
GetLogicalDeviceDirectory	5
GetAllDataValues	6
GetDataValues	7
SetDataValues	8
GetDataDirectory	9
GetDataDefinition	10
GetDataSetValues	11
SetDataSetValues	12
CreateDataSet	13
DeleteDataSet	14
GetDataSetDirectory	15
SelectActiveSG	16
SelectEditSG	17
SetEditSGValue	18
ConfirmEditSGValues	19

ACSI Service	Value
GetEditSGValue	20
GetSGCBValues	21
Report	22
GetBRCBValues	23
SetBRCBValues	24
GetURCBValues	25
SetURCBValues	26
GetLCBValues	27
SetLCBValues	28
QueryLogByTime	29
QueryLogAfter	30
GetLogStatusValues	31
SendGOOSEMessage	32 ^a
GetGoCBValues	33
SetGoCBValues	34
GetGoReference	35 ^a
GetGOOSEElementNumber	36 ^a
SendMSVMessage	37 ^a
GetMSVCBValues	38
SetMSVCBValues	39
SendUSVMessage	40 ^a
GetUSVCBValues	41
SetUSVCBValues	42
Select	43
SelectWithValue	44
Cancel	45
Operate	46
CommandTermination	47
TimeActivatedOperate	48
GetFile	49
SetFile	50
DeleteFile	51
GetFileAttributeValues	52
TimeSynchronization	53
InternalChangeUnknown	54
GetLogicalNodeDirectory	55
^a These services are not mapped within IEC 61850-8-2. However their execution in the context of another SCSM may be logged or reported through IEC 61850-8-2.	

Table 51 – Mapping of ACSI errorCode values

ACSI ServiceError	Value
no-error	0
instance-not-available	1
instance-in-use	2
access-violation	3
access-not-allowed-in-current-state	4
parameter-value-inappropriate	5
parameter-value-inconsistent	6
class-not-supported	7
instance-locked-by-other-client	8
control-must-be-selected	9
type-conflict	10
failed-due-to-communications-constraint	11
failed-due-to-server-constraint	12

Table 52 – Mapping of CDC LTS

ACSI attribute name	Component name	ACSI type	Comment
Shall inherit all the MMS components of the CST CDC			
Specific to the LTS			
logEna	logEna	BOOLEAN	see 8.3.1
logRef	logRef	ObjectReference	see 8.3.3.2.3
datSet	datSet	ObjectReference	see 8.3.3.2.3
	oldEnrTm	EntryTime	see 8.3.3.8. Introduced with the mapping of the LCB in IEC 61850-8-2
	newEnrTm	EntryTime	see 8.3.3.8. Introduced with the mapping of the LCB in IEC 61850-8-2
	oldEnt	EntryID	see 8.3.3.5. Introduced with the mapping of the LCB in IEC 61850-8-2
	newEnt	EntryID	see 8.3.3.5. Introduced with the mapping of the LCB in IEC 61850-8-2
optFlds			Not mapped. This Clause forces the reasonCodes to always appear in the Log.
trgOps	trgOps	TriggerConditions	see 8.3.3.9
intgPd	intgPd	INT32U	see 8.3.1

Table 53 – Mapping of CDC GTS

ACSI attribute name	component name	ACSI type	Comment
Shall inherit all the MMS components of the CST CDC			
Specific to the GTS			
goEna	goEna	BOOLEAN	see 8.3.1
goID	goID	VISIBLE STRING	see 8.3.2.5
datSet	datSet	ObjectReference	The value of this component shall be of the format of ObjectReference and shall be limited to VMD or domain scoped NamedVariableLists
confRev	confRev	INT32U	see 8.3.1
ndsCom	ndsCom	BOOLEAN	see 8.3.1
dstAddress	dstAddress	PhyComAddr	Constraint to the transport used. See 8.3.3.3.1 and 8.3.3.3.2
	minTime	INT32U	see 8.3.1. Introduced with the mapping of the GoCB in IEC 61850-8-2
	maxTime	INT32U	see 8.3.1. Introduced with the mapping of the GoCB in IEC 61850-8-2
	fixedOffs	BOOLEAN	see 8.3.1. Introduced with the mapping of the GoCB in IEC 61850-8-2

16 Setting group control class model

16.1 Setting group control block definition

If it exists within a logical device, the SGCB shall be considered as a specific object of the logical node LLN0, under the functional constraint SP.

The structure representing the SGCB shall be of the type defined in Table 54. The presence conditions of the ACSI components do not differ from IEC 61850-7-2 specifications.

Table 54 – Mapping of SGCB

ACSI component name	ACSI Type	Comment
NumOfSG	INT8U	Mapped according to 8.3.1
ActSG	INT8U	Mapped according to 8.3.1
EditSG	INT8U	Mapped according to 8.3.1
CnfEdit	BOOLEAN	Mapped according to 8.3.1
LActTm	Timestamp	Mapped according to 8.3.1
ResvTms	INT16U	Mapped according to 8.3.1

16.2 Setting group control class services

16.2.1 SelectActiveSG

This service is a specialization of the SetDataValues service (see 13.3) where the value of ActSG shall be set. See in Figure 54 an example of SelectActiveSG request for activating the first setting group of a logical device named CONTROL.

Figure 55 and Figure 56 show examples of respectively a positive response and a negative response in case the referenced SGCB doesn't exist. Note that these examples of positive and negative responses may apply to the other services of the present class, as far as they are mapped on an XML element <confirmed-RequestPDU/> including an element <write/> (i.e. SelectEditSG, SetEditSGValue and ConfirmEditSGValues). For the different possible errors included in the negative response see 8.3.3.4.10.

```
<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB$ActSG</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <unsigned>1</unsigned>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>
```

Figure 54 – XML structure of SelectActiveSG-Request

```
<confirmed-ResponsePDU>
  <invokeID>2018020</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>
```

Figure 55 – XML structure of SelectActiveSG-Response+

```
<confirmed-ResponsePDU>
  <invokeID>2018020</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <failure>
          <object-non-existent/>
        </failure>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>
```

Figure 56 – XML structure of SelectActiveSG-Response-

16.2.2 SelectEditSG

This service is a specialization of the SetDataValues service (see 13.3) where the value of EditSG shall be set. See in Figure 57 an example of SelectEditSG request for selecting the second setting group of a logical device, for edition.

```

<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB$EditSG</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <unsigned>2</unsigned>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 57 – XML structure of SelectEditSG-Request

16.2.3 SetEditSGValue

This service is a specialization of the SetDataValues service (see 13.3) where the functional constraint of the FCD or FCDA shall be SE. See in Figure 58 an example of SetEditSGValue request.

Prior to execution of this service, the appropriate edit group needs to be selected through the use of SelectEditSG service.

```

<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>PVOC1$SE$MinOpTmms$setVal</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <integer>21</integer>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 58 – XML structure of SetEditSGValue-Request

16.2.4 ConfirmEditSGValues

This service is a specialization of the SetDataValues service (see 13.3). The value of CnfEdit shall be set to true. See Figure 59 for an example of ConfirmEditSGValues request.

The effect of when and how the non-volatile storage is updated is a local issue. The methodology shall be documented in the PIXIT. The value of CnfEdit shall return to FALSE once the storage is completed.

```

<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB$CnfEdit</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <boolean>true</boolean>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 59 – XML structure of ConfirmEditSGValues

16.2.5 GetEditSGValue

This service is a specialization of the GetDataValues service (see 13.2) where the functional constraint of the FCD or FCDA shall be SE or SG. See in Figure 60 and Figure 61 examples of GetEditSGValue request and response.

Prior to execution of this service, the appropriate edit group and/or active group need to be selected through the use of SelectEditSG or SelectActiveSG service.

```

<confirmed-RequestPDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>PVOC1$SE$MinOpTmms$setVal</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 60 – XML structure of GetEditSGValue-Request

```

<confirmed-ResponsePDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <integer>123</integer>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 61 – XML structure of GetEditSGValue-Response

16.2.6 GetSGCBValues

This service is a specialization of the GetDataValues service (see 13.2) where the reference to read is the SGCB object (with FC=SP). See Figure 62 and Figure 63 for examples of respectively GetSGCBValues request and response.

```

<confirmed-RequestPDU>
  <invokeID>2637821</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 62 – XML structure of GetSGCBValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>2637821</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <unsigned>3</unsigned>
              </Data>
              <Data>
                <unsigned>1</unsigned>
              </Data>
              <Data>
                <unsigned>2</unsigned>
              </Data>
              <Data>
                <boolean>>false</boolean>
              </Data>
              <Data>
                <utc-time>017215E00000001F</utc-time>
              </Data>
              <Data>
                <unsigned>600</unsigned>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 63 – XML structure of GetSGCBValues-Response

17 Reporting and logging class model

17.1 Report model – Report control blocks

17.1.1 Buffered report control block

A BRCB shall be a specific object within an instance of a logical node, under the functional constraint BR. The structure of the BRCB shall be as defined in Table 55. The types of the BRCB attributes shall be mapped as defined in 8.3.

Table 55 – BRCB structure

BRCB attributes	ACSI type	Comment
RptID	VISIBLE STRING	May not be written when RptEna=TRUE
RptEna	BOOLEAN	
DatSet	ObjectReference	May not be written when RptEna=TRUE
ConfRev	INT32U	May not be written
OptFlds	RCBReportOptions	May not be written when RptEna=TRUE
BufTm	INT32U	May not be written when RptEna=TRUE
SqNum	INT16U	May not be written
TrgOps	TriggerConditions	May not be written when RptEna=TRUE
IntgPd	INT32U	May not be written when RptEna=TRUE
GI	BOOLEAN	
PurgeBuf	BOOLEAN	May not be written when RptEna=TRUE
EntryID	EntryID	May not be written when RptEna=TRUE
TimeOfEntry	EntryTime	May not be written
ResvTms	INT16	May not be written when RptEna=TRUE
Owner	OCTET STRING 64	

A write of a change of value to RptID, DatSet, BufTm, TrgOps, IntgPd, or PurgeBuf shall execute the PurgeBuf procedure described in IEC 61850-7-2.

The naming of instances of a BRCB shall be controlled by the IEC 61850-6 RCB attribute "indexed":

- If the IEC 61850-6 RCB attribute "indexed" is TRUE, the BRCB instance numbering shall start with 01 and progress through 99. Two digits of instance numbering is mandatory. For instance, the SCL <RCB name="myRcb" indexed="true"> lead to MMS Named Components myRCB01, myRCB02, ...myRcbXX where XX represents the attribute max of the SCL subElement RptEnabled of the given RCB.
- If the IEC 61850-6 RCB attribute "indexed" is FALSE, the BRCB instance shall be without additional instance number. For instance, the SCL <RCB name="myRcb" indexed="false"> lead to an MMS Named Components myRCB.

RptEna

An attempt to set the value of RptEna shall fail if the value of DatSet is NULL.

DatSet

An attempt to set a reference to a DataSet that does not exist shall fail unless the value is NULL. The value of NULL shall indicate that there is no DataSet.

OptFlds

OptFlds shall be mapped to a BITSTRING according to 8.3.3.12.

The segmentation bit is reserved in order to keep OptFlds and ReportedOptFlds in alignment.

Owner

Reading the Owner shall return the JID of the application entity that reserved the report control block. The encoding of the ip address shall follow: exactly 4 octets for an IPv4 address (for example: C0A80017 are the 4 octets to use to encode the IPv4 address "192.168.0.23") and 16 octets for an IPv6 address. There are two kinds of report control block owners: the preconfigured owner and the online owner. In case the report control block has been preconfigured within the SCL station file and associated to a dedicated ClientLN SCL element,

then reading the Owner shall return the JID address of the client for which it has been configured, even if the ClientLN is not in the same XMPP domain and will connect to the Server through several XMPP servers (federation).

In case the report control block has been preconfigured without an association to a dedicated ClientLN SCL element, or the BRCB has been instantiated for a dynamic reporting client, then reading the Owner returns an empty-string.

The Owner attribute is for diagnosis only, and shall not be used to refuse a reservation request to a client. IEC 62351-6, which allows authentication of a user and its rights, defines the value that provides an authentication mechanism to deal with denial of reservation.

NOTE IEC 61850-7-2 defines an association of a report control block with a set of cooperating clients. However, the IEC 61850-6 SCL does not support up to now the concept of a set of cooperating client, therefore the static preconfigured ownership is limited to the ClientLN SCL element.

17.1.2 Unbuffered report control block

A URCB shall be a specific object within an instance of a logical node, under the functional constraint RP. The structure of the URCB shall be as defined in Table 56. The types of the URCB attributes shall be mapped as defined in 8.3.

Table 56 – URCB structure

URCB attributes	ACSI type	Comment
RptID	VISIBLE STRING	May not be written when RptEna=TRUE
RptEna	BOOLEAN	
Resv	BOOLEAN	May not be written when RptEna=TRUE
DatSet	ObjectReference	May not be written when RptEna=TRUE
ConfRev	INT32U	May not be written
OptFlds	RCBReportOptions	May not be written when RptEna=TRUE, The values of buffer-overflow and entryID shall be ignored
BufTm	INT32U	May not be written when RptEna=TRUE
SqNum	INT16U	May not be written
TrgOps	TriggerConditions	May not be written when RptEna=TRUE
IntgPd	INT32U	May not be written when RptEna=TRUE
GI	BOOLEAN	
Owner	OCTET STRING	See 17.1.1

OptFlds

The value returned for OptFlds.buffer-overflow and OptFlds.entryID shall always be false.

The values of OptFlds shall reflect the default value(s) if no writing of OptFlds has occurred while the the URCB is reserved (e.g. Resv=TRUE).

If a writing occurs while Resv=FALSE, the value returned when reading OptFlds is a local issue.

A transition of RptEna from FALSE to TRUE shall cause the value of SqNum to be set to zero (0).

The naming of instances of a URCB shall be controlled by the IEC 61850-6 RCB attribute "indexed":

- If the IEC 61850-6 RCB attribute "indexed" is TRUE, the URCB instance numbering shall start with 01 and progress through 99. Two digits of instance numbering is

mandatory. For instance, the SCL <RCB name="myRcb" indexed="true"> lead to MMS Named Components myRCB01, myRCB02, ...myRcbXX where XX represents the attribute max of the SCL subElement RptEnabled of the given RCB.

- If the IEC 61850-6 RCB attribute "indexed" is FALSE, the URCB instance shall be without instance number. For instance, the SCL <RCB name="myRcb" indexed="false"> lead to an MMS Named Components myRCB.

RptEna

An attempt to set the value of RptEna shall fail if the value of DataSet is NULL.

DatSet

An attempt to set a reference to a DataSet that does not exist shall fail unless the value is NULL. The value of NULL shall indicate that there is no DataSet.

17.2 Reporting services

17.2.1 Report service

A Report shall be mapped to an XML element <unconfirmed-PDU/> including an element <informationReport/>. Figure 64 shows an example of Report message.

The <variableAccessSpecification/> element included in this <informationReport/> shall use the choice <variableListName/> and <vmd-specific/> with the value "RPT".

In addition to the <variableAccessSpecification/>, the <informationReport/> shall consist of a sequence of AccessResult. The AccessResult values shall occur as described in Table 57.

For the AccessResult(s) which correspond to an attribute in the control block, their value shall be mapped like the corresponding attribute in the control block: see the ACSI types in Table 55 and Table 56 for respectively the attributes of BRCB and URCB, and refer to 8.3 for knowing how to map the values for the present SCSM.

For the AccessResult(s) which do not correspond to an attribute in the control blocks, the mapping is explained below.

Table 57 – Order of AccessResults for Report

IEC 61850-7-2 report format parameter name	Condition
RptID	Shall always be present
Reported OptFlds	Shall always be present
SeqNum	Shall be present if OptFlds.sequence-number is TRUE
TimeOfEntry	Shall be present if OptFlds.report-time-stamp is TRUE
DatSet	Shall be present if OptFlds.data-set-name is TRUE
BufOvfl	Shall be present if OptFlds.buffer-overflow is TRUE
EntryID	Shall be present if OptFlds.entryID is TRUE
ConfRev	Shall be present if OptFlds.conf-rev is TRUE
SubSeqNum	Shall be present if OptFlds.segmentation is TRUE
MoreSegmentsFollow	Shall be present if OptFlds.segmentation is TRUE
Inclusion-bitstring	Shall be present
data-reference(s)	Shall be present if OptFlds.data-reference is TRUE
value(s)	See AccessResult for value(s)
ReasonCode(s)	Shall be present if OptFlds.reason-for-inclusion is TRUE

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <variableListName>
          <vmd-specific>RPT</vmd-specific>
        </variableListName>
      </variableAccessSpecification>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$BR$simpleBReport05
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <bit-string>0111110010</bit-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <unsigned>2312</unsigned>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <binary-time>031DC5B02DD2</binary-time>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$myNewDS
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <unsigned>1</unsigned>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <bit-string>11</bit-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$ST$Ind1
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$ST$Ind3
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <bit-string>000000000000</bit-string>
              </Data>
              <Data>
                <utc-time>017215E00000001F</utc-time>
              </Data>
            </structure>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>

```

← RptID

← OptFlds

← SeqNum

← TimeOfEntry

← DataSet

← ConfRev

← Inclusion-bitstring

← data-reference(s)

← value(s)

← CONTROL/GGIO1\$ST\$Ind1\$stVal

← CONTROL/GGIO1\$ST\$Ind1\$q (= 0)

← CONTROL/GGIO1\$ST\$Ind1\$t

<boolean>>false</boolean>	← CONTROL/GGIO1\$ST\$Ind3\$stVal
</Data>	
<Data>	
<bit-string>000000000000</bit-string>	← CONTROL/GGIO1\$ST\$Ind3\$q (= 0)
</Data>	
<Data>	
<utc-time>017215E30000001F</utc-time>	← CONTROL/GGIO1\$ST\$Ind3\$t
</Data>	
</structure>	
</success>	
</AccessResult>	
<AccessResult>	
<success>	← ReasonCode(s)
<bit-string>000010</bit-string>	
</success>	
</AccessResult>	
<AccessResult>	
<success>	
<bit-string>000010</bit-string>	
</success>	
</AccessResult>	
</listOfAccessResult>	
</informationReport>	
</UnconfirmedService>	
</unconfirmed-PDU>	

Figure 64 – XML structure of Report

AccessResult for RptID

If the control block RptID value is NULL, then the AccessResult shall contain the object reference specifying the control block which caused the report to be generated.

AccessResult for Reported OptFlds

The values for this result are defined in 8.3.3.12.

The segmentation bit shall be used to indicate the presence or absence of the SubSeqNum and the MoreSegmentsFollow access-result.

If the segmentation bit is TRUE, then the SubSeqNum and MoreSegmentsFollow AccessResults shall be present. If the segmentation bit is FALSE, the SubSeqNum and MoreSegmentsFollow AccessResults shall not be present.

AccessResult for BufOvfl

The AccessResult, containing the value of BufOvfl shall be mapped as an ACSI type BOOLEAN (see mapping in 8.3). The value of FALSE indicates that there is no buffer overflow.

AccessResult for SubSeqNum

The value conveyed in this AccessResult shall be mapped as an ACSI Type INT16U (see mapping in 8.3).

AccessResult for MoreSegmentsFollow

The AccessResult, containing the value of MoreSegmentsFollow shall be mapped as an ACSI type BOOLEAN (see mapping in 8.3). The value of TRUE indicates that there are more report sub-sequences required to complete the entire report.

AccessResult for inclusion-bitstring

The AccessResult, containing the value of the inclusion bitstring shall be mapped as a BITSTRING (see mapping in 8.3.2.1). The number of used bits, within the bitstring shall be equal to the number of members of the DataSet. If a bit value is TRUE, then a value for the corresponding data set member will be present. If OptFld.data-reference and/or OptFlds.reason-for-inclusion are TRUE, then those AccessResults shall also be present. The order of the bits, in the bitstring, shall be the order of the members of the data set.

AccessResult for data-reference(s)

The AccessResult, containing the value of the data-reference shall be mapped as an ACSI type ObjectReference (see mapping in 8.3.3.2.4). The value of the AccessResult shall be the ObjectReference for the data set member whose value is being included. The order of the AccessResult(s) shall be the order of the members of the data set.

AccessResult for value(s)

This list of AccessResult shall contain <success/> elements with the data corresponding to the members of the DataSet that are being reported. Each <success/> element shall be one basic type value or one <structure/> including several basic types, with possibly recursively nested <structure/> inside (see 8.4). The order of the <success/> elements shall be the order of the members of the data set.

AccessResult for ReasonCode(s)

The Reason-for-inclusion shall be encoded as multiple AccessResults. Each AccessResult shall be mapped as an ACSI type ReasonForInclusionInReport. The size of the AccessResult and bit definition shall follow the definition of ReasonForInclusionInReport – see 8.3.3.10. The general-interrogation bit (Bit 5) shall be used to indicate that the reason is due to a general interrogation. This number of AccessResults shall be the same as the number of AccessResults of the value(s) attribute, within the InformationReport. The order of the AccessResult(s) shall be the order of the members of the data set.

17.2.2 GetBRCBValues

The ACSI GetBRCBValues request shall map to an XML element <confirmed-RequestPDU/> including a <read/> element where the <variableAccessSpecification/> element includes one <variableSpecification/> as defined in 8.2.4, and where the reference to read is the BRCB (with FC=BR, see Figure 65).

The ACSI GetBRCBValues response+ maps to a <confirmed-ResponsePDU/> including an element <read/> with one <success/> element returning one <structure/> element including all the attributes of the BRCB as shown in Figure 66.

The mapping of the ACSI service parameters shall be as defined in Table 58

Table 58 – Mapping of GetBRCBValues service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		See structure of XML request in Figure 65
BRCBReference	<variableSpecification/>	See 8.2.4	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		See structure of XML response in Figure 66

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
ReportIdentifier	<listOfAccessResult/>	One <structure/> including a basic type value for each attribute of the BRCB (see 8.4 and Figure 66)	
ReportEnable			
DataSetReference			
ConfigurationRevision			
OptionalFields			
BufferTime			
SequenceNumber			
TriggerOptionsEnabled			
IntegrityPeriod			
GeneralInterrogation			
PurgeBuf			
EntryIdentifier			
TimeOfEntry			
ReserveTimeSecond			
Owner			
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/> or <confirmed-ErrorPDU/>	See 8.3.3.4.9	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>195015</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1$BR$simpleBReport05</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 65 – XML structure of GetBRCBValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>195015</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <visible-string>
                  CONTROL/GGIO1$BR$simpleBReport05
                </visible-string>
              </Data>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <visible-string>
                  CONTROL/GGIO1$myNewDS
                </visible-string>
              </Data>
              <Data>
                <unsigned>1</unsigned>
              </Data>
              <Data>
                <bit-string>0111110010</bit-string>
              </Data>
              <Data>
                <unsigned>0</unsigned>
              </Data>
              <Data>
                <unsigned>0</unsigned>
              </Data>
              <Data>
                <bit-string>000010</bit-string>
              </Data>
              <Data>
                <unsigned>0</unsigned>
              </Data>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <boolean>>false</boolean>
              </Data>
              <Data>
                <octet-string>
                  0000000000000F40
                </octet-string>
              </Data>
              <Data>
                <binary-time>
                  031DC5B02DD2
                </binary-time>
              </Data>
              <Data>
                <integer>0</integer>
              </Data>
              <Data>
                <octet-string/>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</invokeID>
</confirmed-ResponsePDU>

```

←RptID
 ←RptEna
 ←DatSet
 ←ConfRev
 ←OptFlds
 ←BufTm
 ←SqNum
 ←TrgOps
 ←IntgPd
 ←GI
 ←PurgeBuf
 ←EntryID
 ←TimeOfEntry
 ←ResvTms
 ←Owner

```

        </success>
      </AccessResult>
    </listOfAccessResult>
  </read>
</ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 66 – XML structure of GetBRCBValues-Response

17.2.3 SetBRCBValues

The ACSI SetBRCBValues request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <write/> where the <variableAccessSpecification/> element includes one or several <variableSpecification/> elements as defined in 8.2.5. There shall be one <variableSpecification/> element per attribute to be set in the BRCB (with FC=BR). The <listOfData/> element shall include one or several basic type values, one for each attribute to be set in the BRCB. In case the request specifies several <variableSpecification/>, each is treated individually, processed in the order of the list, and may individually fail, whereas other succeed. The same request cannot be used for setting the same BRCB element several times.

The ACSI SetBRCBValues response+ maps to a <confirmed-ResponsePDU/> including an element <write/> with one or several <success/> or <failure/> elements, indicating that the writing of the different attributes has succeeded or failed. The failure in writing one attribute doesn't imply the failure of the other operations.

The Response– service error shall be mapped as specified in 8.3.3.4.10.

The mapping of the ACSI service parameters shall be as defined in Table 59.

An implementation may decide that only one Control Block attribute can be written per service call. This shall be mentioned in its PICS/PIXIT.

Table 59 – Mapping of SetBRCBValues service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		See structure of XML request in Figure 67
BRCBReference	<variableAccessSpecification/>	One <variableSpecification/> for each attribute to be set in the BRCB (see 8.2.5)	
ReserveTimeSecond [0..1]	<listOfData/><Data/>	One basic type value for each attribute described by a <variableSpecification/> above	
ReportIdentifier [0..1]			
DataSetReference [0..1]			
OptionalFields [0..1]			
BufferTime [0..1]			
TriggerOptionsEnabled [0..1]			
IntegrityPeriod [0..1]			
PurgeBuffer [0..1]			
EntryIdentifier [0..1]			
ReportEnable [0..1]			
GeneralInterrogation [0..1]			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/>		See structure of XML response in Figure 68
	<success/> or <failure/>	<success/> elements are empty. For <failure/> values, see 8.3.3.4.10	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/> or <rejectPDU/>	See 8.3.3.4.10	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>3726014</invokeID>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1$BR$simpleBReport05$ResvTms</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

```

</SEQUENCE>
<SEQUENCE>
  <variableSpecification>
    <name>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$BR$simpleBReport05$DatSet</itemId>
      </domain-specific>
    </name>
  </variableSpecification>
</SEQUENCE>
<SEQUENCE>
  <variableSpecification>
    <name>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$BR$simpleBReport05$OptFlds</itemId>
      </domain-specific>
    </name>
  </variableSpecification>
</SEQUENCE>
<SEQUENCE>
  <variableSpecification>
    <name>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$BR$simpleBReport05$TrgOps</itemId>
      </domain-specific>
    </name>
  </variableSpecification>
</SEQUENCE>
</listOfVariable>
</variableAccessSpecification>
<listOfData>
  <Data>
    <integer>600</integer>
  </Data>
  <Data>
    <visible-string>CONTROL/GGIO1$UnkownDS</visible-string>
  </Data>
  <Data>
    <bit-string>0111010000</bit-string>
  </Data>
  <Data>
    <bit-string>011100</bit-string>
  </Data>
</listOfData>
</write>
</ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 67 – XML structure of SetBRCBValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>3726014</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
      <CHOICE>
        <failure>
          <object-undefined/>
        </failure>
      </CHOICE>
    </CHOICE>
    <CHOICE>
      <success/>
    </CHOICE>
    <CHOICE>
      <success/>
    </CHOICE>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 68 – XML structure of SetBRCBValues-Response

17.2.4 GetURCBValues

Shall be mapped as per 17.2.2.

17.2.5 SetURCBValues

Shall be mapped as per 17.2.3.

17.3 Log model

17.3.1 Overview

IEC 61850-7-2 specifies two major components to the Log Model: a control class and the log class itself. However, for the present SCSCM, these two classes are merged into one Log Control Block (LCB) under the Functional Constraint “LG”.

The Log Control Block (LCB) combines the attributes from the IEC 61850-7-2 log control class and other attributes from the log class (see Figure 69).

NOTE Log entries defined in IEC 61850-7-2 as the attributes Entry[1..n] of the log control class can be accessed by the services QueryLogByTime and QueryLogAfter as <JournalEntry/> elements (see 17.3.3.4).

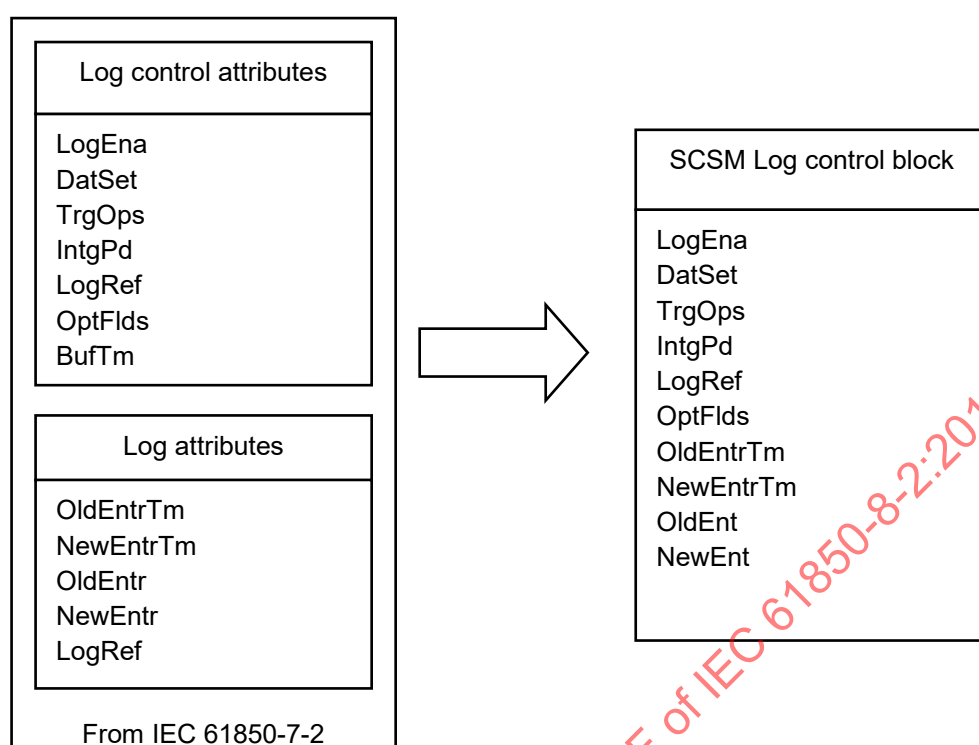


Figure 69 – Relationship of LCB attributes to IEC 61850-7-2 log definitions

The structure of the LCB shall be as defined in Table 60. The types of the LCB attributes shall be mapped as defined in 8.3.

Table 60 – LCB structure

LCB attributes	ACSI Type	r/w	m/o	Comments
LogEna	BOOLEAN	r/w	m	
LogRef	ObjectReference	r/w	m o	May not be written if LogEna=TRUE
DatSet	ObjectReference	r/w	m o	May not be written if LogEna=TRUE
OldEntrTm	EntryTime	r	m	See ^a
NewEntrTm	EntryTime	r	m	See ^a
OldEntr	EntryID	r	m	See ^a
NewEntr	EntryID	r	m	See ^a
OptFlds	LCBLogEntryOptions			Not mapped. This Clause forces the reasonCodes to always appear in the Log.
TrgOps	TriggerConditions	r/w	m	May not be written if LogEna=TRUE
IntgPd	INT32U	r/w	o	May not be written if LogEna=TRUE
bufTm				Not mapped. Every Entry is handled as if bufTm was 0, i.e. without an extra buffering of the events.

^a These attributes are defined as part of the Log Object in IEC 61850-7-2, but are mapped to additional attributes in the LCB within the context of this standard. The values of these attributes shall reflect the appropriate values for the Log specified by the value of LogRef.

17.3.2 Description of LCB attributes

17.3.2.1 LogEna

The default value of LogEna is a local issue, however the default value shall be FALSE for non-configured or improperly configured LCBs.

17.3.2.2 LogRef

Shall be of the data type specified in 8.3.3.2.2. The form of the value shall be that of an ObjectReference. The DEFAULT value shall be an ObjectReference specifying the logical device and the actual name of a journal (i.e LogName).

NOTE A journal is the place where the log entries are stored. This is a concept inherited from MMS, but all the information required to implement it in the context of this SCSM is provided here.

The name of the default journal shall be "GeneralLog". Thus the default value of LogRef shall be of the form: <Logical Device Name>/<LNName>\$GeneralLog.

Example For a LogRef within the Logical Node LLN0 of the Logical Device "LD1", the value of LogRef would be "LD1/LLN0\$GeneralLog".

Writing a LogRef value that specifies a non-existent journal shall fail.

A NULL value is reserved to indicate that no log/journal has been selected.

Multiple LCBs may reference the same log through having the same value of LogRef.

It is recommended that implementations claiming conformance to the log model implement at least one LCB as part of LLN0.

17.3.2.3 DataSet

Shall be of the data type specified in 8.3.3.2.2. Writing a DataSet value that specifies a non-existent data set shall fail.

A NULL value is reserved to indicate that no data set has been selected.

17.3.2.4 OldEntrTm and NewEntrTm

The values shall specify the Entry-Time.

If there are no journal entries within the journal, specified by LogRef, the value of these components shall be equal and the binary-time values shall be 0.

17.3.2.5 OldEntr and NewEntr

The values contained in OldEntr and NewEntr, shall be of type EntryID (see 8.3.3.5). These values shall represent the Journal entryIdentifier values for the oldest and newest entries.

If there are no journal entries within the journal, specified by LogRef, the value of these components shall be equal, and NULL.

17.3.3 Mapping of log and log control services

17.3.3.1 GetLCBValues

The mapping of GetLCBValues is similar to the mapping of GetBRCBValues (see 17.2.2) but the reference to read is the LCB (with FC=LG) and the Response+ shall return the values of the LCB attributes.

17.3.3.2 SetLCBValues

The mapping of the ACSI SetLCBValues is similar to the SetBRCBValues (see 17.2.3).

Writing a value, while the component LogEna has a value of TRUE, shall fail for the attributes LogRef, DatSet, TrgOps, and IntgPd.

17.3.3.3 GetLogStatusValues

As the attributes defined as part of the Log Object in IEC 61850-7-2, are mapped to additional attributes in the LCB, the GetLogStatusValues is the same than GetLCBValues.

17.3.3.4 QueryLogByTime

17.3.3.4.1 General

The ACSI QueryLogByTime request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <readJournal/> (see example in Figure 70).

The ACSI QueryLogByTime response+ maps to a <confirmed-ResponsePDU/> including an element <readJournal/> (see example in Figure 71).

The Response– service error shall be mapped as specified in 17.3.3.4.2.

The mapping of the ACSI service parameters shall be as defined in Table 61.

Table 61 – Mapping of QueryLogByTime service parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <readJournal/>		See structure of XML request in Figure 70
LogReference	<domainId/>	<LDName>	See 17.3.2.2
	<itemId/>	<LNName>\${<LogName>}	
RangeStartTime [0..1]	<startingTime/>	6 bytes xsd:hexbinary	See type TimeOfDay in 8.3.3.8
RangeStopTime [0..1]	<endingTime/>		
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <readJournal/>		See structure of XML request in Figure 71
ListOfLogEntries	<listOfJournalEntry/>		
TimeOfEntry EntryID EntryData [1..n] DataRef Value ReasonCode	<JournalEntry/>		See detail below. Several LogEntry may be included in one <JournalEntry/>
Response–	<confirmed-ErrorPDU/>	See 17.3.3.4.2	

```

<confirmed-RequestPDU>
  <invokeID>3722376</invokeID>
  <ConfirmedServiceRequest>
    <readJournal>
      <journalName>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId>LLN0$GeneralLog</itemId>
        </domain-specific>
      </journalName>
      <rangeStartSpecification>
        <startingTime>01D2EB402DB1</startingTime>
      </rangeStartSpecification>
      <rangeStopSpecification>
        <endingTime>01F3E0C02DB1</endingTime>
      </rangeStopSpecification>
    </readJournal>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 70 – XML structure of QueryLogByTime-Request

```

<confirmed-ResponsePDU>
  <invokeID>3722376</invokeID>
  <ConfirmedServiceResponse>
    <readJournal moreFollows="false">
      <listOfJournalEntry>
        <JournalEntry>
          <entryIdentifier>0000000000000009</entryIdentifier>
          <originatingApplication/>
          <entryContent>
            <occurrenceTime>01D377DF2DB1</occurrenceTime>
            <entryForm>
              <data>
                <event>
                  <eventConditionName>
                    <domain-specific>
                      <domainId>CONTROL</domainId>
                      <itemId>LLN0$LG$myLCB</itemId>
                    </domain-specific>
                  </eventConditionName>
                  <currentState>active</currentState>
                </event>
                <listOfVariables>
                  <SEQUENCE>
                    <variableTag>
                      CONTROL/GGIO1$ST$Ind1$StVal
                    </variableTag>
                    <valueSpecification>
                      <boolean>true</boolean>
                    </valueSpecification>
                  </SEQUENCE>
                  <SEQUENCE>
                    <variableTag>ReasonCode</variableTag>
                    <valueSpecification>
                      <bit-string>000010</bit-string>
                    </valueSpecification>
                  </SEQUENCE>
                </listOfVariables>
              </data>
            </entryForm>
          </entryContent>

```

```

</JournalEntry>
<JournalEntry>
  <entryIdentifier>000000000000000A</entryIdentifier>
  <originatingApplication>
    <ap-invocation-id>101</ap-invocation-id>
  </originatingApplication>
  <entryContent>
    <occurrenceTime>01D982BF2DB1</occurrenceTime>
    <entryForm>
      <data>
        <listOfVariables>
          <SEQUENCE>
            <variableTag>
              CONTROL/GGIO1$ST$Ind1$StVal
            </variableTag>
            <valueSpecification>
              <boolean>>false</boolean>
            </valueSpecification>
          </SEQUENCE>
          <SEQUENCE>
            <variableTag>ReasonCode</variableTag>
            <valueSpecification>
              <bit-string>000010</bit-string>
            </valueSpecification>
          </SEQUENCE>
        </listOfVariables>
      </data>
    </entryForm>
  </entryContent>
</JournalEntry>
</listOfJournalEntry>
</readJournal>
</ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 71 – XML structure of QueryLogByTime-Response

TimeOfEntry

Maps to the element <occurrenceTime/> in <entryContent/>. The value shall be 6 bytes xsd:hexbinary (see TimeOfDay in 8.3.3.8).

EntryID

Maps to the element <entryIdentifier/> of the <JournalEntry/>. The value shall be a 8 bytes xsd:hexbinary. The content of the value is a local issue. An entry identifier whose octets all have a value of 0 shall not be allowed. This value is reserved to indicate that the log is empty.

EntryData

Maps to an element <entryForm/> in <entryContent/>. Within <entryForm/>, the choice <data/> shall be used to map the EntryData. The use of the <annotation/> choice is a local issue.

Within the <data/> element, the <event/> sub-element shall be used to indicate information entered into the log due to log control block control/processing. The use of this constraint is not restricted solely to indicating LCB control, but other uses are considered to be a local issue.

The <event/> element shall be used for journalling transitions of the LCD LogEna value as prescribed:

- The <eventConditionName/> element shall contain the NamedVariable ObjectName representing the log control block that caused the journal entry to be created.

- The event condition state (element `<currentState/>`) shall be the state of the LogEna component of the log control block that caused the journal entry to be created.

The value mapping between LogEna values and event condition state values shall be:

- A value of LogEna being FALSE shall be mapped to the event condition state of `<disabled/>`. The transition of the LogEna value from TRUE to FALSE shall cause a journal entry to be created.
- A value of LogEna being TRUE shall be mapped to either the event condition state of `<idle/>` or `<active/>`. The `<idle/>` state is reserved to indicate a transition of the value of LogEna from FALSE to TRUE. Only a journal entry with an event condition state value of `<active/>` shall have a list of journal variables.

On a power-up or reset the status of the LCB's LogEna should be journaled with a value `<idle/>`.

Then the `<listOfVariables/>` element shall be used for logging of values from DataSets specified by an LCB.

Each LogEntry is mapped within the same `<listOfVariables/>` as two `<SEQUENCE/>` respectively for encoding the dataRef/Value and the reasonCode.

NOTE Several LogEntry can be included within the same `<listOfVariables/>`. Each LogEntry is mapped as two `<SEQUENCE/>` elements as described in this clause.

Each `<SEQUENCE/>` consists of a `<variableTag/>` and a `<valueSpecification/>`. The `<variableTag/>` shall have a minimum-maximum size of 129 octets.

The `<variableTag/>` of the first `<SEQUENCE/>` (i.e. encoding the DataRef/Value) shall have a value of the form ObjectReference – see 8.3.3.2.2. The `<valueSpecification/>` shall include one basic type value or one `<structure/>` with possibly recursively nested `<structure/>` inside (see 8.4).

The `<variableTag/>` of the second `<SEQUENCE/>` (i.e. encoding the ReasonCode) shall have the value of "ReasonCode". The `<valueSpecification/>` shall be a BITSTRING as defined in 8.3.3.11 ReasonForInclusionInLog.

Example The value of a variableTag for a reasonCode relating to a dataRef, whose ObjectReference is LD1/MMXU\$MX\$V, would be: "ReasonCode".

17.3.3.4.2 Response–

The Response– is mapped on a `<confirmed-ErrorPDU/>` including a `<serviceError/>`.

- If the request specifies a Logical Device Name which does not exist, a `<confirmed-ErrorPDU/>` shall be returned and the `<serviceError/>` shall be `<errorClass/><access/>` with an `errorCode=<object-non-existent/>`.
- If the request specifies a LogName for which the Client does not have access privileges, a `<confirmed-ErrorPDU/>` shall be returned and the `<serviceError/>` shall be `<errorClass/><access/>` with an `errorCode=<object-access-denied/>` or `<errorClass/><access/>` with an `errorCode=<object-non-existent/>`.

See Table 62 for ServiceError mappings for Log services.

Table 62 – ServiceError mappings for Log services

ACSI service error	ServiceError		Comments
	Error class	Error code	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-server-constraint	Any class	Any unmapped error codes	

17.3.3.5 QueryLogAfter

The ACSI QueryLogAfter request shall be mapped to an XML element <confirmed-RequestPDU/> including an element <readJournal/> (see example in Figure 72).

The ACSI QueryLogAfter response+ is similar to the response+ of QueryLogByTime (see 17.3.3.4.1)

The ACSI QueryLogAfter response- is similar to the response- of QueryLogByTime (see 17.3.3.4.2)

The mapping of the request parameters shall be as defined in Table 63.

Table 63 – Mapping of QueryLogAfter-Request parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <readJournal/>		See structure of XML request in Figure 72
LogReference	<domainId/>	<LDName>	See 17.3.2.2
	<itemId/>	<LNName>\$<LogName>	
RangeStartTime	<timeSpecification/>	6 bytes xsd:hexbinary	See type TimeOfDay in 8.3.3.8
Entry	<entrySpecification/>	8 bytes xsd:hexbinary	See explanation for EntryID in 17.3.3.4.1

```

<confirmed-RequestPDU>
  <invokeID>3722380</invokeID>
  <ConfirmedServiceRequest>
    <readJournal>
      <journalName>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId>LLN0$GeneralLog</itemId>
        </domain-specific>
      </journalName>
      <entryToStartAfter>
        <timeSpecification>01D2EB402DB1</timeSpecification>
        <entrySpecification>0000000000000009</entrySpecification>
      </entryToStartAfter>
    </readJournal>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 72 – XML structure of QueryLogAfter-Request

17.3.4 Conformance

An implementation that claims conformance to the ACSI log model shall support the services as described in Table 64.

Table 64 – Log conformance requirements

ACSI Service	Client	Server
GetLCBValue	m	m
SetLCBValue	o	m
QueryLogByTime	c1	m
QueryLogAfter	c1	m
GetLogStatusValues	o	m
c1 One of either QueryLogByTime or QueryLogAfter shall be supported		

18 Mapping of the generic substation event model (GSE) – Generic object oriented substation event (GOOSE)

18.1 GOOSE control definition

The GOOSE Control, as defined in IEC 61850-7-2, shall be mapped to a GOOSE Control Block (GoCB) as defined in Table 65. The types of the GoCB attributes shall be mapped as defined in 8.3.

Table 65 –TypeDescription definition for GoCB structure

GoCB Attributes	ACSI Type	r/w	m/o	Comments
GoEna	BOOLEAN	r/w	m	
GoID	VISIBLE STRING	r	m	
DatSet	ObjectReference	r	m	Shall be limited to persistent data set (within a Logical Device or outside any Logical Device)
ConfRev	INT32U	r	m	
NdsCom	BOOLEAN	r	m	
DstAddress	PhyComAddr	r	m	PhyComAddr structure depends on the mapping used for GOOSE communication (see clause 18.2 and 18.3)
MinTime	INT32U	r	o	As specified in the SCD file for the GoCB
MaxTime	INT32U	r	o	As specified in the SCD file for the GoCB
FixedOffs	BOOLEAN	r	o	As specified in the SCD file for the GoCB

GoEna

Shall be as defined in IEC 61850-7-2.

GoID

Shall be as defined in IEC 61850-7-2. The default value of this attribute shall be the GOOSE Control Block reference.

DatSet

The value shall be constrained to the set of existing data set. Writing a value indicating a non-existent data set shall fail.

MinTime

Shall be as defined in IEC 61850-6 the sending delay on a data change between the first immediate sending of the change and the first repetition in ms.

MaxTime

Shall be as defined in IEC 61850-6 the source supervision time in ms (supervision heartbeat cycle time).

Within this time, a failed message from the source shall be detected by the client.

18.2 Specialization for Layer 2 GoCB

All Layer 2 GoCBs shall be of the functional constraint “GO”.

The attribute DstAddress shall be a structured type whose components are defined as described in Table 4 in the PhyComAddr for Layer 2 communication definition.

18.3 Specialization for Routable GOOSE

All Routable GoCBs shall be of the functional constraint “RG”.

The attribute DstAddress shall be a structured type whose components are defined as described in Table 5 in the PhyComAddr for UDP/IP communication definition.

18.4 GOOSE services**18.4.1 General**

The ACSI services GetGoReference, GetGOOSEElementNumber and SendGOOSEMessage are not mapped by the present SCSM because an XMPP mapping would not fulfill the required performances. For these services, implementations shall use the SCSMs as defined in IEC 61850-8-1.

18.4.2 GetGoCBValues

This service is similar to the GetBRCBValues (see 17.2.2) but the reference to read is the GoCB (with FC=GO or RG) and the Response+ shall return the values of the GoCB attributes.

18.4.3 SetGoCBValues

The mapping of the ACSI SetGoCBValues is similar to the SetBRCBValues (see 17.2.3), limited to the GoEna attribute.

19 Transmission of sampled values class model**19.1 Sampled value control block****19.1.1 General**

The Multicast and Unicast control blocks shall be as specified in IEC 61850-9-2 with the exception of the Functional Constraint and DstAddress.

19.1.2 Specialization for Layer 2 Sampled value**19.1.2.1 Functional Constraint**

All Sampled value control block for Layer 2 Sampled value shall be of the functional constraint “MS”.

19.1.2.2 DstAddress

This shall be a structured type whose components are defined as described in the definition of PhyComAddr for Layer 2 communication – see 8.3.3.3.1.

19.1.3 Specialization for Routable Sampled value

19.1.3.1 Functional constraint

All Sampled value control block for routable Sampled value shall be of the functional constraint “RS”.

19.1.3.2 DstAddress

This shall be a structured type whose components are defined as described in the definition of PhyComAddr for UDP/IP communication (see 8.3.3.3.2).

19.1.4 Specialization for Unicast Sampled value

19.1.4.1 Functional constraint

All Sampled value control block for routable Sampled value shall be of the functional constraint “US”.

19.1.4.2 DstAddress

This shall be a structured type whose components are defined as described in the definition of PhyComAddr for Layer 2 communication (see 8.3.3.3.1).

19.2 Sampled value services

19.2.1 General

Only the ACSI services for getting and setting the attributes of the related Control Blocks are mapped within the present SCSM (see Table 1). The services SendMSVMessage, GetMsvReference and GetMSVElementNumber are not mapped within the present document because an XMPP mapping would not fulfill the required performances.

19.2.2 GetMSVCBValues

This service is similar to the GetBRCBValues (see 17.2.2) but the reference to read is the MSVCB (with FC=MS or RS) and the Response+ shall return the values of the MSVCB attributes.

19.2.3 SetMSVCBValues

The mapping of the ACSI SetMSVCBValues is similar to the SetBRCBValues (see 17.2.3).

20 Control class model

20.1 General

The principle is to map the parameters of the ACSI control services (as defined in IEC 61850-7-2) to some dedicated Data Attributes (control component) within the Data Object under control. More specifically, each control service consists in writing (or reading) the sub-attributes of one of these dedicated Data Attributes, each sub-attribute corresponding to one of the service parameters defined in IEC 61850-7-2.

NOTE Only controllable data objects whose data attribute ctlModel is not set to “status-only” are concerned by this clause.

The functional constraint for these data attributes shall be “CO”. So the references to be used for accessing the control parameters for a given DO shall start with <LDname>/<LNname>\$CO\$<DOname>. For convenience such a reference is called CO_CtrlObjectRef in the following clauses.

Depending on the services, one of the following control components (within a CO_CtrlObjectRef) will be written or read: OPER, CANCEL, SBO, SBOw. Each of them may include the sub-attributes described in Table 66 for mapping the needed parameters.

Table 67 shows which of these control components are used by the control services depending on the kind of control model.

In order to be generic enough, the type of the parameter ctrlVal is not defined in IEC 61850-7-2. The type used for the sub-attribute ctrlVal within the control component shall correspond to the type of the ctrlVal attribute defined in the CDC of the object under control.

Table 66 – Controllable service parameters

IEC 61850-7-2 parameter	Semantic	ACSI Type	Comment
ctrlVal	Control value	Depend on the CDC of the control object – see IEC 61850-7-3	
operTm	Timestamp	Timestamp	See 8.3.3.7
origin	Origin	ConstructedAttribute Originator Structure – 8.3.3.16	
ctrlNum	Control number	INT8U	See 8.3.1
T	Control time stamp	Timestamp	See 8.3.3.7
Test	Test status	BOOLEAN	See 8.3.1
Check	Check condition	CheckConditions	See 8.3.3.6
AddCause	Additional cause diagnostic	ENUMERATED	See 8.3.2.2

Table 67 – Mapping of IEC 61850-7-2 control model to control components

Service choice	DataAttributes	M/O/C
Direct with normal security		
	Oper	M
	Cancel	O
SBO with normal security		
	SBO	M
	Oper	M
	Cancel	M
SBO with enhanced security		
	SBOw	M
	Oper	M
	Cancel	M
Direct with enhanced security		
	Oper	M
	Cancel	O

20.2 Overview of control services mapping

Table 68 gives an overview of the mapping of the control services.

Table 68 – Mapping of control services

ACSI service		Type of XML element	Variable specification	Access result
Select	Request	Read-Request	SBO	
	Response +	Read-Response	CO_CtrlObjectRef	Success
	Response –	Read-Response	Empty string	Success
SelectWithValue	Request	Write-Request	SBOw	
	Response +	Write-Response		Success
	Response –	InformationReport (ListOfVariable)	LastApplError	
		Write-Response		Failure
Cancel	Request	Write-Request	Cancel	
	Response +	Write-Response		Success
	Response –	InformationReport ^b (ListOfVariable)	LastApplError	
		Write-Response		Failure
Operate or TimeActivatedOperate (operTm = 0)	Request	Write-Request	Oper	
	Response +	Write-Response		Success
	Response –	InformationReport ^b (ListOfVariable)	LastApplError	
		Write-Response		Failure
TimeActivatedOperate (operTm != 0)	Request	Write-Request	Oper	
	Response +	Write-Response		Success
	Response –	InformationReport ^b (ListOfVariable)	LastApplError	
		Write-Response		Failure
TimeActivatedOperateTermination (operTm != 0)	Request +	InformationReport (ListOfVariable)	Oper	
	Request –	InformationReport (ListOfVariable)	LastApplError	
CommandTermination	Request + ^a	InformationReport (ListOfVariable)	Oper	
	Request – ^a	InformationReport (ListOfVariable)	LastApplError Oper	
^a When available, the value of operTm shall be zero(0) within the CommandTermination message.				
^b Optional for control with normal security				

20.3 Select

The ACSI Select request shall map to an XML element <confirmed-RequestPDU/> including a <read/> element where the <variableAccessSpecification/> element includes one <variableSpecification/> referring to the SBO component of the <CO_CtrlObjectRef >.

The ACSI Select response+ shall map to a <confirmed-ResponsePDU/> including an element <read/> with one <success/> element returning the value <CO_CtrlObjectRef > as a <visible-string/> element.

The ACSI Select response- shall map to a <confirmed-ResponsePDU/> including an element <read/> with one <success/> element returning an empty <visible-string/>.

The mapping of parameters is as specified in Table 69.

Table 69 – Mapping of Select parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		See structure of XML request in Figure 73
ControlObjectReference	<variableSpecification/>	See 8.2.5	Constraint to <CO_ CtrlObjectRef> >\$SBO
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		See structure of XML request in Figure 74
ControlObjectReference	<listOfAccessResult/> with one <success/> element	<CO_ CtrlObjectRef>	Mapped as a <visible-string/>
Response-	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		
ControlObjectReference	<listOfAccessResult/> with one <success/> element	Empty visible-string	

```

<confirmed-RequestPDU>
  <invokeID>1874014</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <!-- Select = Read $SBO -->
                  <itemId>GGIO1$CO$DPCSO1$SBO</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 73 – XML structure of Select-Request


```

<confirmed-ResponsePDU>
  <invokeID>1874014</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <visible-string>CONTROL/GGIO1$CO$DPCSO1</visible-string>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 74 – XML structure of Select-Response

20.4 SelectWithValue

20.4.1 SelectWithValue service parameter mapping

The SelectWithValue service shall be performed by writing the SBOw attribute of the object under control. Therefore the control parameters are mapped to sub-attributes of SBOw as specified in Table 70.

Table 70 – SelectWithValue service parameter mapping

IEC 61850-7-2 parameter	IEC 61850-8-2 component name	FC	TypeDefinition	m/o
ctlVal	<CO_CtrlObjectRef>\$SBOw\$ctlVal		See Table 66	m
operTm	<CO_CtrlObjectRef>\$SBOw\$operTm		See Table 66	c
origin	<CO_CtrlObjectRef>\$SBOw\$origin		See Table 66	m
ctlNum	<CO_CtrlObjectRef>\$SBOw\$ctlNum		See Table 66	m
T	<CO_CtrlObjectRef>\$SBOw\$T		See Table 66	m
Test	<CO_CtrlObjectRef>\$SBOw\$Test		See Table 66	m
Check	<CO_CtrlObjectRef>\$SBOw\$Check		See Table 66	m
AddCause	/LastAppIError		See 20.9	m
NOTE 1 ctlVal type is dependant of the CDC of the object (e.g. ctlVal or ctlVal\$f).				
NOTE 2 The service parameter operTm shall only be present if the control object support TimeActivatedOperate.				

20.4.2 General mapping of the SelectWithValue service

The ACSI SelectWithValue request shall map to an XML element <confirmed-RequestPDU/> including an element <write/> where the <variableAccessSpecification/> element includes one <variableSpecification/> referring to the SBOw component of the <CO_CtrlObjectRef>. The <listOfData/> element shall include a value for all the sub-attributes of SBOw as listed in Table 70.

The ACSI SelectWithValue response+ shall map to a <confirmed-ResponsePDU/> including an element <write/> with one <success/> element.

The ACSI SelectWithValue response– shall map to two messages as specified in 20.4.3.

The mapping of parameters is as specified in Table 71.

Table 71 – Mapping of SelectWithValue parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		See structure of XML request in Figure 75
ControlObjectReference	<variableSpecification/>	See 8.2.5	Constraint to <CO_ CtrlObjectRef >\$SBow
ctlVal	<listOfData/>	See example in Figure 75	Value of each sub-attributes of SBow (which map the parameters defined in 7-2)
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/>		See structure of XML response in Figure 76
ControlObjectReference	One <success/> element		Parameter values are optionally available as tracking of controllable service
ctlVal			
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
Response-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		For informationReport, see 20.4.3 and 20.9
ControlObjectReference	<variableAccessSpecification/>		/LastApplError (see 20.9)
ctlVal	and <confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/> <CHOICE/> <failure/>		For Write-Response, see 20.4.3
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
AddCause			
^a The service parameter operTm shall only be present if the control object support TimeActivatedOperate, and therefore mapped the service parameter operTm to <CO_ControlObjectReference>\$SBow\$OperTm.			

```

<confirmed-RequestPDU>
  <invokeID>89351</invokeID>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1$CO$DPCSO1$SBOW</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <structure>
            <Data>
              <boolean>true</boolean>
            </Data>
            <Data>
              <structure>
                <Data>
                  <integer>2</integer>
                </Data>
                <Data>
                  <octet-string>53544154494F4E</octet-string>
                </Data>
              </structure>
            </Data>
            <Data>
              <unsigned>25</unsigned>
            </Data>
            <Data>
              <utc-time>017215E00000001F</utc-time>
            </Data>
            <Data>
              <boolean>>false</boolean>
            </Data>
            <Data>
              <bit-string>00</bit-string>
            </Data>
          </structure>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 75 – XML structure of SelectWithValue-Request

```

<confirmed-ResponsePDU>
  <invokeID>89351</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 76 – XML structure of SelectWithValue-Response+

20.4.3 SelectWithValue response–

The IEC 61850-7-2 SelectWithValue response– maps onto two messages sent from the server to the client. The order of the messages shall be as follows:

- <unconfirmed-PDU/> including an <informationReport/> with AdditionalCauseDiagnostic (see 20.9).
- <confirmed-ResponsePDU/> corresponding to the SelectWithValue request, indicating <failure/> as indirect child of the <write/> element and values of DataAccessError as defined in Table 72 (see example in Figure 77).

Table 72 – SelectWithValue, Oper and Cancel DataAccessError specification

ACSI Error	DataAccessError	Error condition
TEMPORARILY-UNAVAILABLE	TEMPORARILY UNAVAILABLE	Control is already selected or being operated
HARDWARE-FAULT	HARDWARE-FAULT	Control may not be operated due to hardware fault
OBJECT-ACCESS-DENIED	OBJECT-ACCESS-DENIED	Access control failure
OBJECT-UNDEFINED	OBJECT-UNDEFINED	Control does not exist in this security view

```

<confirmed-ResponsePDU>
  <invokeID>89351</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <failure>
          <temporarily-unavailable/>
        </failure>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 77 – XML structure of SelectWithValue-Response–

20.5 Cancel

20.5.1 Cancel service parameter mapping

The Cancel service shall be performed by writing the Cancel attribute of the object under control. Therefore the control parameters are mapped to sub-attributes of Cancel as specified in Table 73.

Table 73 – Cancel service parameter mapping

IEC 61850-7-2 parameter	IEC 61850-8-1 component name	FC	TypeDefinition	m/o
	<CO_CtrlObjectRef>\$Cancel	CO		
ctlVal	<CO_CtrlObjectRef>\$Cancel\$ctlVal		See Table 66	m
operTm	<CO_CtrlObjectRef>\$Cancel\$operTm		See Table 66	c
origin	<CO_CtrlObjectRef>\$Cancel\$origin		See Table 66	m
ctlNum	<CO_CtrlObjectRef>\$Cancel\$ctlNum		See Table 66	m
T	<CO_CtrlObjectRef>\$Cancel\$T		See Table 66	m
Test	<CO_CtrlObjectRef>\$Cancel\$Test		See Table 66	m
AddCause	/LastAppLError		See 20.9	m
NOTE 1 ctlVal type is dependant of the CDC of the object (e.g. ctlVal or ctlVal\$f).				
NOTE 2 The service parameter operTm shall only be present if the control object support TimeActivatedOperate.				

20.5.2 General mapping of the Cancel service

The ACSI Cancel request shall map to an XML element <confirmed-RequestPDU/> including an element <write/> where the <variableAccessSpecification/> element includes one <variableSpecification/> referring to the Cancel component of the <CO_CtrlObjectRef>. The <listOfData/> element shall include a value for all the sub-attributes of Cancel as listed in Table 73.

The ACSI Cancel response+ shall map to a <confirmed-ResponsePDU/> including an element <write/> with one <success/> element.

The ACSI Cancel response– shall map to one or two messages as specified in 20.5.3.

The mapping of parameters is as specified in Table 74.

Table 74 – Mapping of the Cancel service

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		Similar to structure of XML request for SelectWithValue in Figure 75, but without value for the last parameter (Check) which doesn't exist for Cancel
ControlObjectReference	<variableSpecification/>	See 8.2.5	Constraint to <CO_ CtrObjectRef >\$Cancel
ctlVal	<listOfData/>	Similar to example in Figure 75, (but without last bit-string value)	Value of each sub-attributes of Cancel (which map the parameters defined in 7-2)
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/>		See structure of XML response in Figure 76
ControlObjectReference	One <success/> element		Parameter values are optionally available as tracking of controllable service
ctlVal			
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Response-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		For informationReport ^b , see 20.5.3 and 20.9.
ControlObjectReference	<variableAccessSpecification/>		/LastApplError (see 20.9)
ctlVal	and		For Write-Response, see 20.5.3
operTm[0..1] ^a	<confirmed-ResponsePDU/>		
origin	<ConfirmedServiceResponse/>		
ctlNum	<write/>		
T	<CHOICE/>		
Test	<failure/>		
AddCause			

^a The service parameter operTm shall only be present if the control object support TimeActivatedOperate, and therefore mapped the service parameter operTm to <CO_ControlObjectReference>\$Cancel\$OperTm.

^b Optional for control models with normal security; mandatory otherwise

20.5.3 Cancel response–

The IEC 61850-7-2 Cancel response- maps onto one or two messages sent from the server to the client. The order of the messages shall be as follows:

- <unconfirmed-PDU/> including an <informationReport/> with AdditionalCauseDiagnostic (optional for normal security and mandatory for enhanced security control models) (see 20.9).

- <confirmed-ResponsePDU/> corresponding to the Cancel request, indicating <failure/> as indirect child of the <write/> element and values of DataAccessError as defined in Table 72 (see example in Figure 77).

20.6 Operate

20.6.1 Operate service parameter mapping

The Operate service shall be performed by writing the Oper attribute of the object under control. Therefore the control parameters are mapped to sub-attributes of Oper as specified in Table 75.

Table 75 – Operate service parameter mapping

IEC 61850-7-2 parameter	IEC 61850-8-1 component name	FC	TypeDefinition	m/o
	<CO_CtrlObjectRef>\$Oper	CO		
ctlVal	<CO_CtrlObjectRef>\$Oper\$ctlVal		See Table 66	m
operTm	<CO_CtrlObjectRef>\$Oper\$operTm		See Table 66	c
origin	<CO_CtrlObjectRef>\$Oper\$origin		See Table 66	m
ctlNum	<CO_CtrlObjectRef>\$Oper\$ctlNum		See Table 66	m
T	<CO_CtrlObjectRef>\$Oper\$T		See Table 66	m
Test	<CO_CtrlObjectRef>\$Oper\$Test		See Table 66	m
Check	<CO_CtrlObjectRef>\$Oper\$Check		See Table 66	m
AddCause	/LastApplError		See 20.9	m
NOTE 1 ctlVal type is dependant of the CDC of the object (e.g. ctlVal or ctlVal\$f).				
NOTE 2 The service parameter operTm shall only be present if the control object support TimeActivatedOperate.				

20.6.2 General mapping of the Operate service

The ACSI Operate request shall map to an XML element <confirmed-RequestPDU/> including an element <write/> where the <variableAccessSpecification/> element includes one <variableSpecification/> referring to the Oper component of the <CO_CtrlObjectRef>. The <listOfData/> element shall include a value for all the sub-attributes of Oper as listed in Table 75.

The ACSI Operate response+ shall map to a <confirmed-ResponsePDU/> including an element <write/> with one <success/> element.

The ACSI Operate response– shall map to one or two messages as specified in 20.6.3 and 20.9.

The mapping of parameters is as specified in Table 76.

Table 76 – Mapping of the Operate service

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		Similar to structure of XML request for SelectWithValue in Figure 75
ControlObjectReference	<variableSpecification/>	See 8.2.5	Constraint to <CO_ CtrlObjectRef >\$Oper
ctlVal	<listOfData/>	Similar to example in Figure 75	Value of each sub-attributes of Oper (which map the parameters defined in 7-2)
operTm[0..1] ^a			
Origin			
ctlNum			
T			
Test			
Check			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/>		See structure of XML response in Figure 76
ControlObjectReference	One <success/> element		Parameter values are optionally available as tracking of controllable service
ctlVal			
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
Response-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		For informationReport ^b , see 20.6.3 and 20.9.
ControlObjectReference	<variableAccessSpecification/>		/LastApplError (see 20.9)
ctlVal	and <confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/> <CHOICE/> <failure/>		For Write-Response (see 20.6.3)
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
AddCause			
^a The service parameter operTm shall only be present if the control object support TimeActivatedOperate, and therefore mapped the service parameter operTm to <CO_ CtrlObjectRef>\$Oper\$OperTm. It shall have a value of zero (0) for the Operate service.			
^b Client implementations shall be aware that they may not receive the Information report in the context of gateways with other SCSMs. But servers compliant to the present SCSM shall implement it.			

20.6.3 Operate response–

The IEC 61850-7-2 Operate response– maps onto one or two messages sent from the server to the client. The order of the messages shall be as follows:

- <unconfirmed-PDU/> including an <informationReport/> with AdditionalCauseDiagnostic (optional for normal security and mandatory for enhanced security control models) (see 20.9).
- <confirmed-ResponsePDU/> corresponding to the Operate request, indicating <failure/> as indirect child of the <write/> element and values of DataAccessError as defined in Table 72 (see example in Figure 77).

CommandTermination

20.6.4 CommandTermination service parameter mapping

None of the IEC 61850-7-2 service parameters are needed additionally to the one specified for the Operate / TimeActivatedOperate service.

20.6.5 General mapping of the CommandTermination service

The CommandTermination request+ shall be mapped to an XML element <unconfirmed-PDU/> including an element <informationReport/> (see example in Figure 78).

The <variableAccessSpecification/> part of this <informationReport/> shall map to the reference to the Oper component of the object under control (i.e. <CO_CtrlObjectRef>\$Oper) and the <listOfAccessResult/> part shall map to the values of the same Oper component.

The CommandTermination-request+ of a Control with TimeActivation shall have an operTm attribute value of zero(0).

The CommandTermination request– shall be mapped to an XML element <unconfirmed-PDU/> including an element <informationReport/> containing two variables (see example in Figure 79). The first variable is the vmd specific variable “LastApplError” (see 20.9). The second variable is the Oper component of the object under control, like for request+.

The CommandTermination service shall be mapped as specified in Table 77.

Table 77 – Mapping of the CommandTermination service

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request+	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		See structure of XML request in Figure 78
ControlObjectReference	<variableSpecification/>	See 8.2.5	Constraint to <CO_ CtrlObjectRef >\$Oper
ctlVal	<listOfAccessResult/>	See example in Figure 78	Value of each sub-attributes of Oper (which map the parameters defined in 7-2)
operTm[0..1] ^a			
Origin			
ctlNum			
T			
Test			
Check			
Request-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		See structure of XML request in Figure 79
ControlObjectReference	<variableSpecification/>	See 8.2.3 and 8.2.5	/LastApplError + <CO_ CtrlObjectRef>\$Oper
ctlVal	<listOfAccessResult/>	See example in Figure 79	First <success/> element with values for LastApplError then second <success/> element with values for each sub-attributes of Oper (which map the parameters defined in 7-2)
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
AddCause			
^a The service parameter operTm shall only be present if the control object support TimeActivatedOperate, and therefore mapped the service parameter operTm to <CO_ControlObjectReference>\$Oper\$OperTm. It shall have a value of zero (0) for the Operate service.			

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1$CO$DPCSO1$Oper</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    <listOfAccessResult>
      <AccessResult>
        <success>
          <structure>
            <Data>
              <boolean>true</boolean>
            </Data>
            <Data>
              <structure>
                <Data>
                  <integer>2</integer>
                </Data>
                <Data>
                  <octet-string>53544154494F4E</octet-string>
                </Data>
              </structure>
            </Data>
            <Data>
              <unsigned>25</unsigned>
            </Data>
            <Data>
              <utc-time>017215E00000001F</utc-time>
            </Data>
            <Data>
              <boolean>>false</boolean>
            </Data>
            <Data>
              <bit-string>00</bit-string>
            </Data>
          </structure>
        </success>
      </AccessResult>
    </listOfAccessResult>
  </informationReport>
</UnconfirmedService>
</unconfirmed-PDU>

```

Figure 78 – XML structure of CommandTermination Request+

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <vmd-specific>LastApplError</vmd-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CO$DPCSO1$Oper</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
      </listOfVariable>
    </variableAccessSpecification>
  <listOfAccessResult>
    <AccessResult>
      <success>
        <structure>
          <Data>
            <visible-string>
              CONTROL/GGIO1$CO$DPCSO1$Oper
            </visible-string>
          </Data>
          <Data>
            <integer>3</integer>
          </Data>
          <Data>
            <structure>
              <Data>
                <integer>2</integer>
              </Data>
              <Data>
                <octet-string>53544154494F4E</octet-string>
              </Data>
            </structure>
          </Data>
          <Data>
            <unsigned>25</unsigned>
          </Data>
          <Data>
            <integer>3</integer>
          </Data>
        </structure>
      </success>
    </AccessResult>
  <AccessResult>
    <success>
      <structure>
        <Data>
          <boolean>true</boolean>
        </Data>
        <Data>

```

```

<structure>
  <Data>
    <integer>2</integer>
  </Data>
  <Data>
    <octet-string>53544154494F4E</octet-string>
  </Data>
</structure>
</Data>
<Data>
  <unsigned>25</unsigned>
</Data>
<Data>
  <utc-time>017215E00000001F</utc-time>
</Data>
<Data>
  <boolean>>false</boolean>
</Data>
<Data>
  <bit-string>00</bit-string>
</Data>
</structure>
</success>
</AccessResult>
</listOfAccessResult>
</informationReport>
</UnconfirmedService>
</unconfirmed-PDU>

```

Figure 79 – XML structure of CommandTermination Request-

20.7 TimeActivatedOperate

20.7.1 TimeActivatedOperate service parameter mapping

None of the IEC 61850-7-2 service parameters are needed additionally to the one specified for the Operate service.

20.7.2 Mapping of the TimeActivatedOperate service

The mapping of TimeActivatedOperate service shall be similar to the mapping of Operate, but the parameter OperTm shall be present and shall contain a valid timestamp. The value OperTm = 0 (i.e. with all octets set to 0) is equivalent to control with enhanced security (i.e. without time activation).

20.8 TimeActivatedOperateTermination service

The mapping of TimeActivatedOperateTermination service shall be similar to the mapping of CommandTermination, with the following differences:

- In the Request+, the value of the operTm attribute shall have the same value as written in the TimeActivatedOperate request. If the OperTm value is zero(0) (all octets set to 0), this shall indicate the CommandTermination.
- The Request- shall contain only the first variable: the vmd specific variable "LastApplError" (see 20.9).

20.9 AdditionalCauseDiagnosis in negative control service responses

Some abstract control services supply additional application specific information in their negative responses. This information is defined by the AdditionalCauseDiagnosis (AddCause) service parameter defined in IEC 61850-7-2. The transmission of this information shall map to an XML element <unconfirmed-PDU/> including an element <informationReport/> containing the vmd specific variable “LastApplError” of the structure defined in Table 78 (see example of resulting XML message in Figure 80). This variable shall be instantaneously created, reported, and then deleted.

Table 78 – Definition of LastApplError variable structure

Component name	ACSI TypeDescription	r/w	m/o	Comments
CntrlObj	VISIBLE STRING	r	m	
Error	ENUMERATED	r	m	
Origin	Originator	r	m	See IEC 61850-7-2
ctlNum	INT8U	r	m	See IEC 61850-7-2
AddCause	ENUMERATED	r	m	

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <vmd-specific>LastApplError</vmd-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <visible-string>
                  CONTROL/GGIO1$CO$DPCSO1$Oper
                </visible-string>
              </Data>
              <Data>
                <integer>3</integer>
              </Data>
              <Data>
                <structure>
                  <Data>
                    <integer>2</integer>
                  </Data>
                  <Data>
                    <octet-string>53544154494F4E</octet-string>
                  </Data>
                </structure>
              </Data>
              <Data>
                <unsigned>25</unsigned>
              </Data>
              <Data>
                <integer>3</integer>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </informationReport>
  </UnconfirmedService>
</unconfirmed-PDU>

```

Figure 80 – XML structure of InformationReport with AdditionalCauseDiagnosis

The value of the variable shall represent the last detected control service application error that required additional diagnostics. The value of LastApplError shall be reported as a single variable (e.g. the structured variable shall be reported). The value of this variable shall be volatile and shall assume the specified default values upon two-party association establishment.

CntrlObj

CntrlObj shall have a VISIBLE STRING data type of the structure

<LDeviceName>/<LNVariableName>\${FC}\${<LNDataName1>}\${<AttributeName1>
(see 8.3.3.2)

and shall have a maximum size of 129 octets. The value shall represent the control object that was operated upon that caused the additional cause to be generated.

The default value shall be NULL.

Error

Error shall have an ENUMERATED data type as defined in 8.3.2.2. The enumerated values for error shall be defined as:

- (0) No Error
- (1) Unknown
- (2) Timeout Test Not OK
- (3) Operator Test Not OK

The default value shall be “No Error” (0). The value “No Error” (0) shall be returned when the negative acknowledgment is not issued by negative test results within the control state machine for a given control model (typically for addCause values that encode application error: Time-limit-over, Position-reached, 1-of-n-control, ...).

“Unknown” shall be returned when the negative acknowledgement is not issued by negative test results within the control state machine for a given control model and an internal error occurred.

“TimeOut Test Not OK” shall be returned if a TimeActivated control command is issued, with TEST=TRUE, and the operation fails.

“Operator Test Not OK” shall be returned if an Operate control command is issued, with TEST=TRUE, and the operation fails.

origin

Used to designate the client who initiated the control action.

ctlNum

The control sequence number as specified by the client initiating the service.

NOTE Negative control responses mostly consist of a <confirmed-ResponsePDU/> including an element <write/> indicating failure of the Control service request (mapped to a Write-Request) and an additional InformationReport request defined above. The transmission of the origin and ctlNum in the InformationReport request enables the client to allocate this InformationReport request to the corresponding control service request.

The default value shall be 0.

AddCause

AddCause shall be an ENUMERATED data type as defined in 8.3.2.2. Table 79 defines the actual values assigned to AddCause.

Table 79 – Mapping of ACSI AddCause values

ACSI value	Value
Unknown	0
Not-supported	1
Blocked-by-switching-hierarchy	2
Select-failed	3
Invalid-position	4
Position-reached	5
Parameter-change-in-execution	6
Step-limit	7
Blocked-by-Mode	8
Blocked-by-process	9
Blocked-by-interlocking	10
Blocked-by-synchrocheck	11
Command-already-in-execution	12
Blocked-by-health	13
1-of-n-control	14
Abortion-by-cancel	15
Time-limit-over	16
Abortion-by-trip	17
Object-not-selected	18
Object-already-selected	19
No-access-authority	20
Ended-with-overshoot	21
Abortion-due-to-deviation	22
Abortion-by-communication-loss	23
Blocked-by-command	24
None	25
Inconsistent-parameters	26
Locked-by-other-client	27

20.10 Tracking of control services

20.10.1 General

The ServiceTracking model of the control services maps directly to a specialization of data.

It is important to note that the data attribute of type object reference are mapped to **VISIBLE STRING 129** and shall contain the ACSI object reference as specified in IEC 61850-7-2 and shall not contain the object address as specified in 8.3.3.2.3.

NOTE The mapping of the CDC CST is not defined in the present SCSM. It can be directly deduced from the general clause describing the mapping of the data attributes (see 8.3).

20.10.2 Mapping of the Control service tracking (CTS)

The mapping of the CDC CTS shall follow the definition specified in Table 80.

Table 80 – Mapping of CDC CTS

ACSI attribute name	component name	ACSI type	Comment
Shall inherit all the components of the CST CDC			
Specific to the CTS			
ctlVal	ctlVal	See clause 8.3	Depending of the CDC of the control object – see IEC 61850-7-3
operTm	operTm	Timestamp	See 8.3.3.7
origin	origin	Structure – see 8.3.3.16	ConstructedAttribute Originator Structure – see IEC 61850-7-2
ctlNum	ctlNum	INT8U	See 8.3.1
T	T	Timestamp	See 8.3.3.7
Test	Test	BOOLEAN	See 8.3.1
Check	Check	CheckConditions	See 8.3.3.15
respAddCause	respAddCause	ENUMERATED	See 8.3.2.2 and Table 79

21 Time and time synchronization model

Time synchronization shall be accomplished through the use of SNTP (see 5.3).

Hardware synchronization mechanisms (e.g. GPS or other) are beyond the scope of this standard.

22 Naming conventions

See clause 8.2.

23 File transfer

23.1 File transfer model

The principle is that the IEC 61850-7-2 file class shall be mapped to the MMS file object. However in the context of this SCSM, knowledge of the MMS file object is not a prerequisite. If an implementation declares support for file transfer, it shall be implemented according to this Clause. In addition, it is a local issue if the IETF File Transfer Protocol (FTP) or Secure File Transfer Propotocol (sFTP) is also implemented. The conformance to the FTP RFC is out-of-scope of this standard.

The mapping is specified in Table 81.

Table 81 – Mapping of ACSI file class to MMS file object

IEC 61850-7-2 file class attribute name		MMS file object attribute		PresCond
Attribute	Type	Attribute	Type	
FileName	VisString255	FileName	FileName	M
FileSize	INT32U	Size	Unsigned	M
LastModified	TimeStamp	LastModified	GeneralizedTime	M

FileName

The IEC 61850-7-2 FileName attribute shall be mapped to the MMS FileName attribute. The MMS FileName shall consist of a sequence of directory names and a name-of-a-file. IEC 61850-7-2 FileNames shall be constrained to be no larger than 255 octets.

The path specification consisting in a sequence of file directory names is mandatory in the File name. File directory names shall be constrained to be no larger than 32 octets. Directory names shall be separated by the PIXIT determined character (typically a “\” or “/”).

A server that contains files related to logical devices shall have as one of its root directories a directory whose name is “LD”. Below the LD root shall be a set of directory names that represent the logical devices within the server. These file directory names shall be the same value as the logical device names (e.g. the names of the domains), and shall be present if files exist. Files that are specific to a logical device shall be located within the directory that represents the logical device.

NOTE Not all files may be able to be associated with a particular LD. As an example, the image file of the server itself may not be directly associated with an LD. Therefore, for files not associated with a LD, it is a local issue in regards to the directory in which such files are stored.

The minimum-maximum name-of-a-file size shall be 12 octets. It is recommended that the file specification be a maximum size of 64 octets. The maximum file specification size shall be specified in the implementation’s PIXIT statement.

FileName suffixes should be used to differentiate the content format of files. The suffix should be a maximum of 3 octets. The suffixes in Table 82 are reserved.

It is recommended that a returned FileName, that is a directory on the server, should end with a PIXIT determined delimiter of either “\” or “/”.

FILE ServiceError Mappings to ACSI service errors

File services use specialized error codes returned within elements <serviceError/> with <errorClass/><file/>. Table 84 shows the appropriate mappings between the ACSI service error values and ServiceErrors for File operations.

Table 82 – Reserved file suffixes

Suffix	Contents
Bin	Shall indicate binary format
Dtd	eXtensible Markup Language Document Type Description file of format dtd:XML
Gif	Shall indicate graphics interchange format contents
Htm	Shall indicate HTML contents
Pqd	Power Quality Data Interchange Format – PQDIF
txt	Shall indicate ASCII contents
Xml	eXtensible Markup Language file of format table-xml:XML
Xsd	eXtensible Markup Language Schema Definition file of format xsd:XML
Zip	Shall indicate zip compressed file format

FileName case sensitivity shall be declared in the implementation’s PIXIT statement.

COMTRADE FILES

Either IEC 60255-24:2013 / IEEE C37.111-2013 or IEEE C37.111.1999 (COMTRADE) files shall be contained within a file directory whose name is “COMTRADE”. The file specifications shall be consistent with the naming conventions and suffixes specified in IEC 60255-24:2013 resp. IEEE C37.111:1999.

The IEC COMTRADE specification IEC 60255-24:2013 specifies the use of five different suffixes (e.g. cff, inf, hdr, cfg, and dat). In normal information/computational usage these suffixes may represent files other than COMTRADE.

The IEEE COMTRADE specification IEEE C37.111:1999 specifies the use of three different suffixes (e.g. hdr, cfg, and dat). In normal information/computational usage these suffixes may represent files other than COMTRADE.

If the directory contains a file with a suffix of “zip”, that file shall convey the compressed contents of the COMTRADE hdr, cfg, and dat files of the files of the same name. The file may optionally contain the associated hdr and/or inf files.

The COMTRADE directories shall be located in the appropriate directory path (e.g. within the LD directory or at the root level).

FileSize

A value of 0 may be used to indicate that the file has an unknown size or that the file contains 0 octets.

The size value shall be interpreted as an estimated file size and shall not be used to determine absolute size.

NOTE Transfers from one file system to another may increase or decrease the actual size of the file.

LastModified

Maps to an xsd:dateTime.Local or UTC time are allowed, whereas UTC time is the recommended format.

23.2 File services

23.2.1 GetFile

23.2.1.1 General

The ACSI GetFile service shall be mapped to a sequence of unitary services: one FileOpen, one or several FileRead, and one FileClose services. Each of these unitary services shall consist in an exchange of two XML messages, one for the request including an XML element <confirmed-RequestPDU/> with respectively elements of type FileOpen-Request, FileRead-Request or FileClose-Request inside, and a second for the confirmation including an XML element <confirmed-ResponsePDU/> with elements of type FileOpen-Response, FileRead-Response or FileClose-Response.

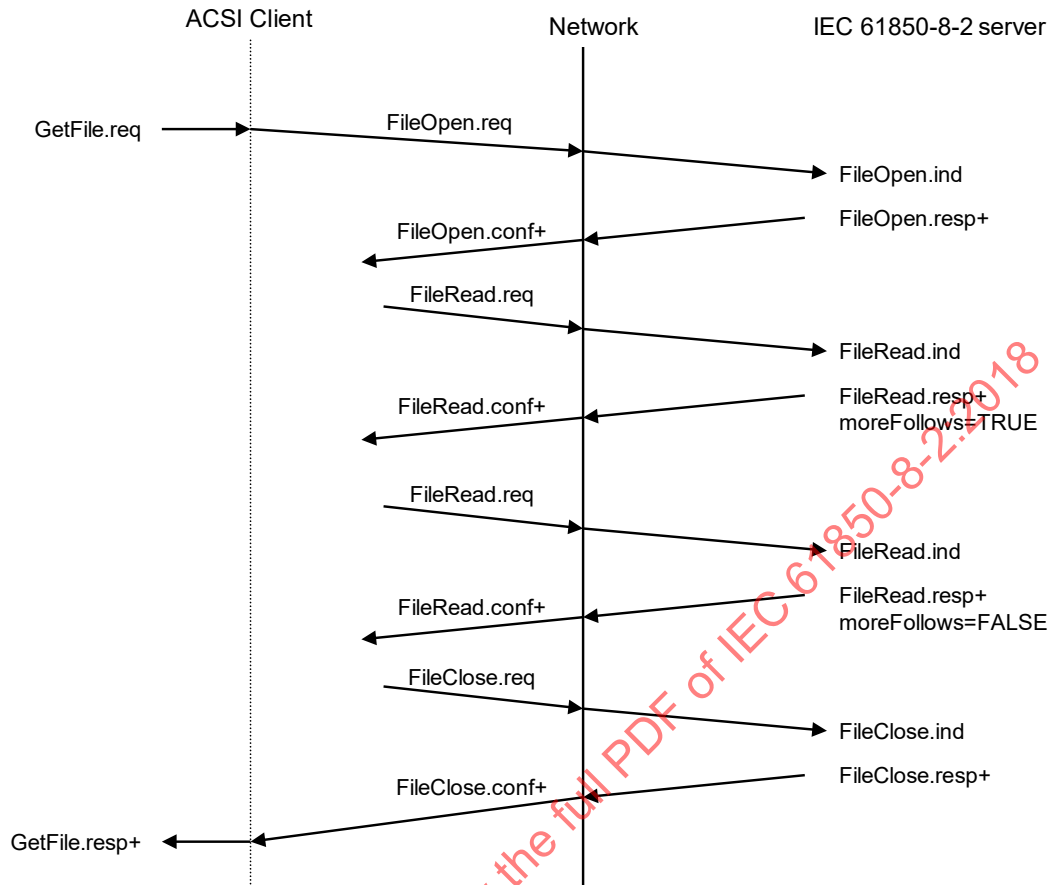


Figure 81 – Mapping of ACSI GetFile to FileOpen, FileRead, FileClose

Figure 81 shows the sequence of unitary services that result from an ACSI GetFile request. An ACSI GetFile request causes a FileOpen request to be generated. The initialPosition shall be 0. A FileOpen positive confirmation shall cause the first in a possible sequence of FileRead requests to be generated. A FileRead positive confirmation with moreFollows=FALSE shall cause a FileClose request to be issued. A FileRead positive confirmation with moreFollows=TRUE shall cause another in the sequence of FileRead requests to be issued. Upon receipt of a FileClose positive confirmation, a GetFile positive response indication shall be given.

If any unitary services return a negative confirmation, then a GetFile negative response indication shall be given as specified in Table 84 and 23.2.1.2.

Table 83 specifies the mapping of the ACSI GetFile service to a sequence of unitary file services. Figure 82 to Figure 89 show examples of XML messages resulting from this sequence.

Table 83 – Mapping of ACSI GetFile service

ACSI service or parameter	Unitary service or parameter	XML element	Constraint
GetFile request	FileOpen.Request		
FileName	FileName	Sequence of <GraphicString/>	FileName includes one <GraphicString/> element if its size is less than 64 characters. Otherwise additional <GraphicString/> elements with a maximum size of 64 characters are appended. Shall be present and shall have a non-NULL value.
	FileOpen.Resp+		
	ID	<frsmID/>	Unique ID of the file assigned by the server
	Size	<sizeOfFile/>	
	FileRead.Request		
	Unique ID	<fileRead/>	
	FileRead.Resp+		
	FileData	<fileData/>	Content of the file in xsd:hexbinary format
	moreFollows	moreFollows = true	
	FileRead.Request		
	Unique ID	<fileRead/>	
	FileRead.Resp+		
	FileData	<fileData/>	Content of the file in xsd:hexbinary format
	moreFollows	moreFollows = false	
	FileClose.Request		
	Unique ID	<fileClose/>	
	FileClose.Resp+		
GetFile response+			
File-Data			Concatenation of FileData returned by all the unitary FileRead
GetFile response–		See Table 84 and 23.2.1.2	
ServiceError		<serviceError/>	

See Table 84 for mappings of GetFile ACSI service errors.

Table 84 – GetFile service error mappings

ACSI service error	ServiceError	
	Error class	Error code
parameter-value-inappropriate	file	filename-ambiguous
Instance-locked-by-other-client	file	file-busy
parameter-value-inappropriate	file	filename-syntax-error
type-conflict	file	content-type-invalid
parameter-value-inconsistent	file	position-invalid
access-violation	file	file-access-denied
instance-not-available	file	file-non-existent
instance-in-use	file	duplicate-filename
failed-due-to-server-constraint	file	insufficient-space-in-filestore

```

<confirmed-RequestPDU>
  <invokeID>3819313</invokeID>
  <ConfirmedServiceRequest>
    <fileOpen>
      <fileName>
        <GraphicStringMMS>MyDir/MySubDir/MyOtherSubDir/MyVeryLongSubDirec
toryName/MyFileNa</GraphicStringMMS>
        <GraphicStringMMS>me</GraphicStringMMS>
      </fileName>
      <initialPosition>0</initialPosition>
    </fileOpen>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 82 – XML example of FileOpen Request

```

<confirmed-ResponsePDU>
  <invokeId>3819313</invokeId>
  <ConfirmedServiceResponse>
    <fileOpen>
      <frsmID>9</frsmID>
      <fileAttributes>
        <sizeOfFile>383</sizeOfFile>
      </fileAttributes>
    </fileOpen>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 83 – XML example of FileOpen Response+

```

<confirmed-RequestPDU>
  <invokeID>3819314</invokeID>
  <ConfirmedServiceRequest>
    <fileRead>9</fileRead>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 84 – XML example of FileRead Request (first)

```

<confirmed-ResponsePDU>
  <invokeID>3819314</invokeID>
  <ConfirmedServiceResponse>
    <fileRead moreFollows="true" >
      <fileData>
        54686520414353492047657446696c652073657276696365207368616c6c20626520
        6d617070656420746f20612073657175
      </fileData>
    </fileRead>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 85 – XML example of FileRead Response+ (first)

```

<confirmed-RequestPDU>
  <invokeID>3819315</invokeID>
  <ConfirmedServiceRequest>
    <fileRead>9</fileRead>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 86 – XML example of FileRead Request (second)

```

<confirmed-ResponsePDU>
  <invokeID>3819315</invokeID>
  <ConfirmedServiceResponse>
    <fileRead moreFollows="true" >
      <fileData>
        566669726d65642d526571756573745044552061207365636f
      </fileData>
    </fileRead>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 87 – XML example of FileRead Response+ (second)

```

<confirmed-RequestPDU>
  <invokeID>3819316</invokeID>
  <ConfirmedServiceRequest>
    <fileClose>9</fileClose>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 88 – XML example of FileClose Request

```

<confirmed-ResponsePDU>
  <invokeId>3819316</invokeId>
  <ConfirmedServiceResponse>
    <fileClose/>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 89 – XML example of FileClose Response

23.2.1.2 Response–

23.2.1.2.1 FileOpen

This clause details the mappings for the responses regarding errors that can occur when performing the FileOpen unitary service request:

- A FileOpen-Request that specifies a <fileName/> for which the Client does not have access privileges shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<file-access-denied/> or <errorClass/><file/> with an errorCode=<file-non-existent/>.

- A FileOpen-Request that specifies a <fileName/> that does not exist shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<file-non-existent/>.
- A FileOpen-Request that specifies a <fileName/> which is not available due to some local locking mechanism shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<file-busy/>.
- A FileOpen-Request that specifies a <fileName/> which is not syntactically correct shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with an errorCode=<filename-syntax-error/>.
- A FileOpen-Request that specifies an <initialPosition/> that is not correct shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with a errorCode=<position-invalid/>.

See Table 85 for mappings of ACSI ServiceErrors to FileOpen Service Errors.

Table 85 – Mappings of ACSI ServiceErrors to FileOpen Service Errors

ACSI ServiceError	ServiceError	
	Error class	Error code
instance-not-available	file	file-non-existent
Instance-locked-by-other-client	file	file-busy
parameter-value-inappropriate	file	filename-syntax-error
parameter-value-inconsistent	file	position-invalid
access-violation	file	file-access-denied
instance-not-available	file	file-non-existent

23.2.1.2.2 FileRead

This subclause details the mappings for the responses regarding errors that can occur when performing the FileRead unitary service request:

- A FileRead-Request that specifies an invalid FRSM shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><service/> with a errorCode=<object-state-conflict/>.

See Table 86 for mappings of ACSI ServiceErrors to FileRead Service Errors.

Table 86 – Mappings of ACSI ServiceErrors to FileRead Service Errors

ACSI ServiceError	ServiceError	
	Error class	Error code
parameter-value-inconsistent	service	object-state-conflict

23.2.1.2.3 FileClose

This subclause details the mappings for the responses regarding errors that can occur when performing the FileClose unitary service request:

- A FileClose-Request that specifies an invalid FRSM shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><service/> with a errorCode=<object-state-conflict/>.

See Table 87 for mappings of ACSI ServiceErrors to FileClose Service Errors.

Table 88 – Mapping of ACSI SetFile parameters

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <obtainFile/>		See example in Figure 91
FileName	<destinationFile/>		Shall be present and shall have a non-NULL value.
	<sourceFile/>		Name of the file from which FileData is to be obtained by the server through the ACSI GetFile service
File-Data	FileRead response FileData ^a		Shall be mapped to the sequence of FileRead response FileData returned
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <obtainFile/>		See example in Figure 92
Response-	<confirmed-ErrorPDU/>	See 23.2.2.2	

^a This mapping occurs as part of the execution of the ACSI GetFile service.

```

<confirmed-RequestPDU>
  <invokeID>3615231</invokeID>
  <ConfirmedServiceRequest>
    <obtainFile>
      <sourceFile>
        <GraphicStringMMS>MyDir/MySubDir/MyOtherSubDir/MyVeryLongSubDir
toryName/MyFileNa</GraphicStringMMS>
        <GraphicStringMMS>me</GraphicStringMMS>
      </sourceFile>
      <destinationFile>
        <GraphicStringMMS>destDir/destName</GraphicStringMMS>
      </destinationFile>
    </obtainFile>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 91 – XML example of ObtainFile Request

```

<confirmed-ResponsePDU>
  <invokeId>3615231</invokeId>
  <ConfirmedServiceResponse>
    <obtainFile/>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 92 – XML example of ObtainFile Response**23.2.2.2 Response-**

This subclause details the mappings for the responses regarding errors that can occur when performing the SetFile service request:

- An ObtainFile-Request that specifies a <sourceFile/> for which the resulting FileOpen-Request fails shall return a ObtainFileError PDU (in fact this is a <serviceError/> including a <serviceSpecificInformation/> element using the choice <obtainFile/>). The ObtainFileError PDU shall indicate a <source-file/> problem.

- An ObtainFile-Request that specifies in <destinationFile/> a directory for which the Client does not have access privileges shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><access/> with an errorCode=<object-access-denied/> or <errorClass/><file/> with an errorCode=<file-non-existent/>.
- An ObtainFile-Request that specifies a <destinationFile/> with a file that already exists, shall return a ObtainFileError PDU. The ObtainFileError PDU shall indicate a <destination-file/> problem.

See Table 89 for mappings of ACSI ServiceErrors to ObtainFile Service Errors.

Table 89 – Mappings of ACSI ServiceErrors to ObtainFile Service Errors

ACSI service error	ServiceError		Comments
	Error class	Error code	
access-violation	access	object-access-denied	
Instance-not-available	file	file-non-existent	
failed-due-to-communications-constraint	Any class	Any unmapped error codes	
ACSI service error	ObtainFileError		Comments
instance-not-available	source-file		
access-not-allowed-in-current-state	destination-file		

23.2.3 DeleteFile

23.2.3.1 General

The ACSI DeleteFile request and response+ map respectively to a <confirmed-RequestPDU/> including an element <fileDelete/> (see example in Figure 93) and a <confirmed-ResponsePDU/> including an element of type <fileDelete/> (see example in Figure 94).

The ACSI DeleteFile service parameters shall map as detailed in Table 90.

Table 90 – Mapping of ACSI DeleteFile service

Parameters as specified in IEC 61850-7-2	XML element	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <fileDelete/>		See example in Figure 93
FileName	Sequence of <GraphicString/>		Sequence of directory names (path) + name of the file. Shall be present and shall have a non-NULL value.
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <fileDelete/>		See example in Figure 94
Response-	<confirmed-ErrorPDU/>	See 23.2.3.2	

```

<confirmed-RequestPDU>
  <invokeID>3819320</invokeID>
  <ConfirmedServiceRequest>
    <fileDelete>
      <GraphicStringMMS>MyDir/MySubDir/MyOtherSubDir/MyVeryLongSubDirectory
Name/MyFileNa</GraphicStringMMS>
      <GraphicStringMMS>me</GraphicStringMMS>
    </fileDelete>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 93 – XML example of DeleteFile Request

```

<confirmed-ResponsePDU>
  <invokeId>3819320</invokeId>
  <ConfirmedServiceResponse>
    <fileDelete/>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 94 – XML example of DeleteFile Response

23.2.3.2 Response–

This subclause details the mappings for the responses regarding errors that can occur when performing the DeleteFile service request:

- A DeleteFile-Request that specifies a filespecification for which the Client does not have access privileges shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with a errorCode=<file-access-denied/> or <errorClass/><file/> with an errorCode=<file-non-existent/>.
- A DeleteFile-Request that specifies a filespecification that does not exist shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with a errorCode=<file-non-existent/>.
- A DeleteFile-Request that specifies a filespecification that is not available due to some local locking mechanism shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with a errorCode=<file-busy/>.
- A DeleteFile-Request that specifies a filespecification that is not syntactically correct shall return a <confirmed-ErrorPDU/>. The <serviceError/>, within the <confirmed-ErrorPDU/> shall be <errorClass/><file/> with a errorCode=<filename-syntax-error/>.

See Table 91 for mappings of ACSI ServiceErrors to DeleteFile Service Errors.

Table 91 – Mappings of ACSI ServiceErrors to DeleteFile Service Errors

ACSI service error	ServiceError	
	Error class	Error code
instance-not-available	file	file-non-existent
Instance-locked-by-other-client	file	file-busy
parameter-value-inappropriate	file	filename-syntax-error
access-violation	file	file-access-denied

23.2.4 GetFileAttributeValues

The ACSI GetFileAttributeValues request and response+ map respectively to a <confirmed-RequestPDU/> including an element <fileDirectory/> and a <confirmed-ResponsePDU/> including an element <fileDirectory/>.

An ACSI GetFileAttributeValues request shall include a single <fileSpecification/> element. The corresponding response+ shall indicate moreFollows=FALSE and only one element <directoryEntry/>.

An ACSI GetFileAttributeValues response- shall be indicated as in 8.3.3.4.11.

The parameters of the ACSI GetFileAttributeValues service shall be mapped as detailed in Table 92.

Table 92 – Mapping of ACSI GetFileAttributeValues parameters

Parameters as specified in IEC 61850-7-2	XML element / attribute	Value	Comment
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <fileDirectory/>		See example in Figure 95
FileName	<fileSpecification/>	Sequence of <GraphicString/>	Shall be mapped as specified in 23.1
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <fileDirectory/>		See example in Figure 96
FileName	<listOfDirectoryEntry/>	See example in Figure 96	Only one entry that matches with the <fileSpecification/>. <fileAttributes/> shall be mapped as specified in 23.1
FileSize[0..1]			
LastModified			
	moreFollows	false	Constrained to false
Response-	<confirmed-ErrorPDU/>	See 8.3.3.4.11	

The example below (Figure 95 and Figure 96) describes the GetFileAttributeValues request and response+ for a file located in nested folders, whose name is "Folder/SubFolder/MyFileName".

```

<confirmed-RequestPDU>
  <invokeID>3819321</invokeID>
  <ConfirmedServiceRequest>
    <fileDirectory>
      <fileSpecification>
        <GraphicStringMMS>Folder/SubFolder/MyFileName</GraphicStringMMS>
      </fileSpecification>
    </fileDirectory>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 95 – XML example of GetFileAttributeValues Request

```

<confirmed-ResponsePDU>
  <invokeID>3819321</invokeID>
  <ConfirmedServiceResponse>
    <fileDirectory moreFollows="true" >
      <listOfDirectoryEntry>
        <DirectoryEntry>
          <fileName>
            <GraphicStringMMS>Folder/SubFolder/MyFileName
          </fileName>
          <fileAttributes>
            <sizeOfFile>383</sizeOfFile>
            <lastModified>201512160532-0500</lastModified>
          </fileAttributes>
        </DirectoryEntry>
      </listOfDirectoryEntry>
    </fileDirectory>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 96 – XML example of GetFileAttributeValues Response

24 Conformance

24.1 Notation

For this clause, the following definitions apply.

- m: mandatory support. The item shall be implemented.
- c: conditional support. The item shall be implemented if the stated condition exists.
- o: optional support. The implementation may decide to implement the item.
- x: excluded. The implementation shall not implement this item.
- i: out-of-scope. The implementation of the item is not within the scope of this standard but the item may be supported by specific implementations.
- F/S: Functional Standard. Shall be applied in the context of IEC 61850-8-1.
- Base: Specifies the conformance from an underlying referenced specification (e.g MMS, IEC 61850-6, etc.).
- Declared: Specifies the support of a particular implementation.
- CR: Conformance requirement.

NOTE When conditional support depends on what is declared within the ACSI conformance statement, it is implicitly assumed that support of IEC 61850-8-2 is also declared in the ACSI basic conformance statement.

24.2 PICS

24.2.1 Profile conformance

Table 93, Table 94 and Table 95 define the basic conformance statement.

Table 93 – PICS for A-Profile support

A-Profile shortcut	Profile Description	Client		Server		Value/comment
		F/S	Declared	F/S	Declared	
A1	Client/server A-Profile	m		m		Refer to A.2.2.2
A2	GOOSE Service/ GSE management A-Profile	i		i		
A3	GSSE A-Profile	i		i		Deprecated
A4	TimeSync A-Profile	c		m		Refer to 5.3 and A.2.3.2 Server column specifies the IEC 61850 Server and not the TimeSync Server. Client column specifies the IEC 61850 Client and not the TimeSync Client.
A5	Security for Client/server A-Profile	m		m		Refer to IEC 62351-6
A6	Security for GOOSE Service /GSE management A-Profile	i		i		
A7	SV Service / SV management A-Profile	i		i		
A8	Security for SV Service / SV management A-Profile	i		i		

c Shall be 'm' if support for TimeActivatedControl or Logging is declared

Table 94 – PICS for Time Sync A-Profile support

A-Profile shortcut	Profile Description	Client		Server		Value/comment
		F/S	Declared	F/S	Declared	
Tm1	Simple Network Time Protocol	c		m		Refer to 5.3 and A.2.3.2
Tm2	PTP Profile for Power Utility Automation	i		i		

c Shall be 'm' if support for TimeActivatedControl or Logging is declared

Table 95 – PICS for T-Profile support

T-Profile	Profile Description	Client		Server		Value/Comment
		F/S	Declared	F/S	Declared	
T1	TCP/IP T-Profile	m		m		Refer to A.2.2.3
T2	OSI T-Profile	i		i		deprecated
T3	GOOSE Services/GSE management T-Profile	i		i		
T4	GSSE T-Profile	i		i		
T5	TimeSync T-Profile	c		m		Refer to A.2.3.3
T6	SV Services / SV management Profile T-Profile	i		i		
T7	Security for TCP/IP T-Profile	m		m		Already covered by T1

c Shall be 'm' if support for TimeActivatedControl or Logging is declared

24.2.2 XML Payload conformance

24.2.2.1 Environmental services

24.2.2.1.1 Initiate conformance

Table 96 and Table 97 specify the conformance of the initiate service.

Table 96 – MMS InitiateRequest general parameters

InitiateRequest	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
InitiateRequest						
localDetailCalling	m	m	Proposed MMS PDU size	m	m	Proposed MMS PDU size
proposedMaxServOutstandingCalling	m	m	1 or greater	m	m	1 or greater
proposedMaxServOutstandingCalled	m	m	1 or greater	m	m	1 or greater
proposedDataStructureNestingLevel	m	m	c2	m	m	c2
initRequestDetail	m	m		m	m	
InitiateRequestDetail						
proposedVersionNumber ¹	m	m	Shall be 1	m	m	Shall be 1
proposedParameterCBB	m	m	See 24.2.2.1.3	m	m	See 24.2.2.1.3
servicesSupportedCalling	m	m	See 24.2.2.1.2	m	m	See 24.2.2.1.2
additionalSupportedCalling	c1	x		c1	x	
additionalCbbSupportedCalling	c1	x		c1	x	
privilegeClassIdentityCalling	c1	x		c1	x	

c1 Conditional upon Parameter CBB CSPI – see Table 99.

c2 Shall be five(5) or greater if ACSI logical node model support is declared.

¹ The proposed version is of type integer and shall remain 1 for backward compatibility reasons.

Table 97 – MMS InitiateResponse general parameters

InitiateResponse	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
InitiateResponse						
localDetailCalled	m	m	Negotiated MMS PDU size	m	m	Negotiated MMS PDU size
negotiatedMaxServOutstandingCalling	m	m	1 or greater	m	m	1 or greater
negotiatedMaxServOutstandingCalled	m	m	1 or greater	m	m	1 or greater
negotiatedDataStructureNestingLevel	m	m	c2	m	m	c2
initResponseDetail	m	m		m	m	
InitiateResponseDetail						
negotiatedVersionNumber ¹	m	m	Shall be 1	m	m	Shall be 1
negotiatedParameterCBB	m	m	See 24.2.2.1.3	m	m	See 24.2.2.1.3
servicesSupportedCalled	m	m	See 24.2.2.1.2	m	m	See 24.2.2.1.2
additionalSupportedCalled	c1	x		c1	x	
additionalCbbSupportedCalled	c1	x		c1	x	
privilegeClassIdentityCalled	c1	x		c1	x	
c1 Conditional upon Parameter CBB CSPI – see Table 99.						
c2 Shall be five(5) or greater if ACSI logical node model support is declared.						
¹ The proposed version is of type integer and shall remain 1 for backward compatibility reasons.						

24.2.2.1.2 Services supported

Table 98 defines the service support requirement, and restrictions, for this standard.

Table 98 – MMS service supported conformance table

MMS service supported CBB	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Client Calling, Server Called						
initiate	m	m		m	m	
conclude	m	m		m	m	
Server Calling, Client Called						
initiate	o	c4		o	c4	
conclude	o	c4		o	c4	
Abort						
abort request	o	m		o	m	
abort indication	m	m		m	m	
status	o	o		o	o	
getNameList	o	o		o	c1	
identify	o	o		m	m	
rename	o	i		o	i	
read	o	o		o	c2	
write	o	o		o	c3	
getVariableAccessAttributes	o	o		o	c1	

MMS service supported CBB	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
defineNamedVariable	o	i		o	i	
defineScatteredAccess	o	x		o	x	
getScatteredAccessAttributes	o	x		o	x	
deleteVariableAccess	o	i		o	i	
defineNamedVariableList	o	o		o	c6	
getNamedVariableListAttributes	o	o		o	c5	
deleteNamedVariableList	o	o		o	c6	
defineNamedType	o	i		o	i	
getNamedTypeAttributes	o	i		o	i	
deleteNamedType	o	i		o	i	
input	o	i		o	i	
output	o	i		o	i	
takeControl	o	i		o	i	
relinquishControl	o	i		o	i	
defineSemaphore	o	i		o	i	
deleteSemaphore	o	i		o	i	
reportPoolSemaphoreStatus	o	i		o	i	
reportSemaphoreStatus	o	i		o	i	
initiateDownloadSequence	o	i		o	i	
downloadSegment	o	i		o	i	
terminateDownloadSequence	o	i		o	i	
initiateUploadSequence	o	i		o	i	
uploadSegment	o	i		o	i	
terminateUploadSequence	o	i		o	i	
requestDomainDownload	o	i		o	i	
requestDomainUpload	o	i		o	i	
loadDomainContent	o	i		o	i	
storeDomainContent	o	i		o	i	
deleteDomain	o	i		o	i	
getDomainAttributes	o	o		o	c14	
createProgramInvocation	o	i		o	i	
deleteProgramInvocation	o	i		o	i	
start	o	i		o	i	
stop	o	i		o	i	
resume	o	i		o	i	
reset	o	i		o	i	
kill	o	i		o	i	
getProgramInvocationAttributes	o	i		o	i	
obtainFile	o	c9		o	c9	
defineEventCondition	o	i		o	i	
deleteEventCondition	o	i		o	i	
getEventConditionAttributes	o	i		o	i	
reportEventConditionStatus	o	i		o	i	

MMS service supported CBB	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
alterEventConditionMonitoring	o	i		o	i	
triggerEvent	o	i		o	i	
defineEventAction	o	i		o	i	
deleteEventAction	o	i		o	i	
alterEventEnrollment	o	i		o	i	
reportEventEnrollmentStatus	o	i		o	i	
getEventEnrollmentAttributes	o	i		o	i	
acknowledgeEventNotification	o	i		o	i	
getAlarmSummary	o	i		o	i	
getAlarmEnrollmentSummary	o	i		o	i	
readJournal	o	c13		o	c13	
writeJournal	o	i		o	i	
initializeJournal	o	i		o	i	
reportJournalStatus	o	i		o	i	
createJournal	o	i		o	i	
deleteJournal	o	i		o	i	
getCapabilityList	o	o		o	o	
GetFile (Client sends requests, Server issues responses)						
fileOpen	o	c8		o	c8	
fileRead	o	c8		o	c8	
fileClose	o	c8		o	c8	
SetFile (Server sends requests, Client issues responses)						
fileOpen	o	c9		o	c9	
fileRead	o	c9		o	c9	
fileClose	o	c9		o	c9	
fileRename	o	i		o	i	
fileDelete	o	c9		o	c9	
fileDirectory	o	c11		o	c11	
unsolicitedStatus	o	i		o	i	
informationReport	o	c15		o	c7	
eventNotification	o	i		o	i	
attachToEventCondition	o	i		o	i	
attachToSemaphore	o	i		o	i	
cancel	o	o		o	o	
getDataExchangeAttributes	o	c10		o	c10	
exchangeData	o	c10		o	c10	
defineAccessControlList	o	c10		o	c10	
getAccessControlListAttributes	o	c10		o	c10	
reportAccessControlledObjects	o	c10		o	c10	
deleteAccessControlList	o	c10		o	c10	
alterAccessControl	o	c10		o	c10	
reconfigureProgramInvocation	o	c10		o	c10	

c1	Shall be 'm' if logical device or logical node model support is declared in ACSI basic conformance statement.
c2	Shall be 'm' if logical node model support is declared in ACSI basic conformance statement or if ACSI support for Select service or if support for the MMS write service is declared.
c3	Shall be 'm' if ACSI support for SetDataValues, SelectActiveSG, SelectEditSG, SetEditSGValue, ConfirmEditSGValues, SetURCBValues, SetBRCBValues, SetLCBValues, SetGoCBValues, SetMSVValues, SetUMVCBValues, SelectWithValue, Cancel, Operate, or TimeActivatedOperate service is declared or implied.
c4	May be required for communication involving distributed generation.
c5	Shall be 'm' if data set support is declared in the ACSI basic conformance statement.
c6	Shall be 'm' if support for defineNamedVariableList is declared.
c7	Shall be 'm' if support for ACSI Report or ControlModel direct control with normal security, direct control with enhanced security, SBO control with enhanced security is declared.
c8	Shall be 'm' if support for ACSI GetFile is declared.
c9	Shall be 'm' if support for ACSI SetFile is declared.
c10	Shall not be present since MMS minor version is declared to be 1.
c11	Shall be 'm' if support for ACSI GetFileAttributeValues is declared.
c12	
c13	Shall be 'm' if support for the ACSI QueryLogByTime or QueryLogAfter is declared.
c14	Shall be 'm' if support for the ACSI logical device model is declared.
c15	Shall be 'm' if support for receive Report or ControlModel direct control with normal security, direct control with enhanced security, SBO control with enhanced security is declared.

24.2.2.1.3 Parameter CBB

The specific settings for the MMS parameter CBB are specified in Table 99.

Table 99 – MMS Parameter CBB

MMS parameter CBB	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
STR1	o	c2		o	c3	
STR2	o	o		o	m	
NEST	1	c1		1	m	
VNAM	o	c1		o	m	
VADR	o	i		o	i	
VALT	o	c2		o	c3	
bit 5	x	x		x	x	
TPY	o	i		o	i	
VLIS	o	o		o	c2	
bit 8	x	x		x	x	
bit 9	x	x		x	x	
CEI	o	i		o	i	
ACO	o	i		o	i	
SEM	o	i		o	i	
CSR	o	i		o	i	
CSNC	o	i		o	i	
CSPLC	o	i		o	i	
CSPI	o	i		o	i	
c1	Shall be 'm' if ACSI logical node model support is declared.					
c2	Shall be 'm' if ACSI data set support is declared.					
c3	Shall be 'm' if there is a ACSI data object that has an Array implemented					

24.2.2.1.4 GetNameList conformance

Table 100 defines the conformance of the GetNameList service.

Table 100 – GetNameList conformance statement

GetNameList	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
ObjectClass	m	m		m	m	
ObjectScope	m	m		m	m	
DomainName	o	o		m	m	
ContinueAfter	o	m		m	m	
Response+						
List Of Identifier	m	m		m	m	
MoreFollows	m	m		m	m	
Response–						
Error Type	m	m		m	m	
NOTE Object class 'vmd' (formerly VMDSpecific in MMS V1.0) shall not appear. If a request contains this ObjectClass, an MMS Reject shall be issued.						

24.2.2.1.5 GetCapabilityList conformance

Table 101 defines the conformance of the GetCapabilityList service.

Table 101 – GetCapabilityList conformance statement

GetCapabilityList	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
ContinueAfter	o	m		m	m	
Response+						
List Of Capabilities	m	m		m	m	shall at least contain “IEC61850-8-1:2007B”
MoreFollows	m	m		m	m	
Response–						
Error Type	m	m		m	m	
NOTE GetCapabilityList service is not used within the scope of IEC 61850 mapping; however it is suggested to implement it for compatibility with MMS systems.						
For backward compatibility reasons, client implementations compliant with the current version of the standard shall not refuse connection with servers that do not return a “IEC61850-8-1:2007B” in the ListOfCapabilities.						

24.2.2.1.6 GetDomainAttributes conformance

Table 102 defines the conformance of the GetDomainAttributes service.

Table 102 – GetDomainAttributes conformance statement

GetDomainAttributes	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
DomainName	m	m		m	m	
Response+						
List Of Capabilities	m	m		m	m	
State	m	i		m	i	
MmsDeletable	m	i		m	i	
Sharable	m	i		m	i	
ListOfProgramInvoations	m	i		m	i	
UploadInProgress	m	i		m	i	
AccessControlList	c	i		c	i	
Response–						
Error Type	m	m		m	m	
NOTE GetDomainAttributes service is currently not used within the scope of IEC 61850 mapping; however it is suggested to implement it for compatibility with MMS systems.						

24.2.2.1.7 Status conformance

Table 103 defines the conformance of the Status service.

Table 103 – Status conformance statement

Status	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
ExtendedDerivation	m	m		m	m	
Response+						
StatusResponse	m	m		m	m	
Response–						
Error Type	m	m		m	m	
NOTE Status service is currently not used within the scope of IEC 61850 mapping; however it is suggested to implement it for compatibility with MMS systems.						

24.2.2.1.8 MMS Cancel conformance

Table 104 defines the conformance of the MMS Cancel service.

Table 104 – Cancel conformance statement

Cancel	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
OriginalInvokeID	m	m		m	m	
Response+						
OriginalInvokeID	m	m		m	m	
Response–						
OriginalInvokeID	m	m		m	m	
Error Type	m	m		m	m	
NOTE MMS Cancel Service is currently not used within the scope of IEC 61850 mapping; however it is suggested to implement it for compatibility with MMS systems. Cancel service can be used by an MMS client to cancel a request that has previously been issued, but has not yet completed.						

24.2.2.1.9 Identify conformance

Table 105 defines the conformance of the Identify service.

Table 105 – Identify conformance statement

Identify	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
Response+						
VendorName	m	m		m	m	
ModelName	m	m		m	m	
Revision	m	m		m	m	
ListOfAbstractSyntaxes	o	i		o	i	
NOTE Identify service is currently not used within the scope of IEC 61850 mapping; however its support is mandated by the MMS specification.						

24.2.2.2 Variable access conformance**24.2.2.2.1 Supporting productions****24.2.2.2.1.1 AlternateAccess**

AlternateAccess is the only way to get access to array elements, or sub-element of array elements. It is recommended to limit usage of AlternateAccess to those cases.

Identifier within the members “component” or “componentName” of the AlternateAccess shall not include the “\$” sign, in other word be named flattened variables.

Table 106 defines the conformance of the alternate access selection.

Table 106 – AlternateAccess conformance statement

AlternateAccess	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
unmaned	o	m		o	m	
named	o	m		o	m	
componentName	o	m		o	m	
access	o	m		o	m	

24.2.2.2.1.2 AlternateAccessSelection

Table 107 defines the conformance of the alternate access selection.

Table 107 – AlternateAccessSelection conformance statement

AlternateAccessSelection	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
selectAlternateAccess						
accessSelection	o	m		m	m	
component	o	m		m	m	
index	o	m		m	m	
indexRange	o	i		m	i	
lowIndex	o	i		m	i	
numberOfElements	o	i		m	i	
allElements	o	i		m	i	
alternateAccess	o	m		m	m	
selectAccess						
component	o	i		m	i	
index	o	i		m	i	
indexRange	o	i		m	i	
lowIndex	o	i		m	i	
numberOfIndex	o	i		m	i	
allElement	o	i		m	i	

24.2.2.2.1.3 AlternateAccess examples

The following clause exposes examples of the conformant usage and non conformant usage of AlternateAccess.

VariableSpecification for ACSI reference LDevice/MHA11.HA.phsAHar(7).cVal.mag.f is shown in Figure 97.

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDevice</domainId>
      <itemId>MHA11</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <component>MX</component>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>HA</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <component>phsAHar</component>
                        </accessSelection>
                        <alternateAccess>
                          <CHOICE>
                            <unnamed>
                              <selectAlternateAccess>
                                <accessSelection>
                                  <index>7</index>
                                </accessSelection>
                                <alternateAccess>
                                  <CHOICE>
                                    <unnamed>
                                      <selectAlternateAccess>
                                        <accessSelection>
                                          <component>cVal</component>
                                        </accessSelection>
                                        <alternateAccess>
                                          <CHOICE>
                                            <unnamed>
                                              <selectAlternateAccess>
                                                <accessSelection>
                                                  <component>mag</component>
                                                </accessSelection>
                                                <alternateAccess>
                                                  <CHOICE>
                                                    <unnamed>
                                                      <selectAlternateAccess>
                                                        <accessSelection>
                                                          <component>f</component>
                                                        </accessSelection>
                                                        <alternateAccess>
                                                          <selectAlternateAccess>
                                                            <unnamed>
                                                              <CHOICE>
                                                                </alternateAccess>
                                                              </CHOICE>
                                                            </alternateAccess>
                                                          </selectAlternateAccess>
                                                        </unnamed>
                                                      </CHOICE>
                                                    </alternateAccess>
                                                  </selectAlternateAccess>
                                                </unnamed>
                                              </CHOICE>
                                            </alternateAccess>
                                          </selectAlternateAccess>
                                        </unnamed>
                                      </CHOICE>
                                    </alternateAccess>
                                  </selectAlternateAccess>
                                </unnamed>
                              </CHOICE>
                            </alternateAccess>
                          </selectAlternateAccess>
                        </unnamed>
                      </CHOICE>
                    </alternateAccess>
                  </selectAlternateAccess>
                </unnamed>
              </CHOICE>
            </alternateAccess>
          </selectAlternateAccess>
        </accessSelection>
      </component>
    </accessSelection>
  </alternateAccess>
</CHOICE>
</alternateAccess>

```

```

</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>

```

Figure 97 – VariableSpecification for LDevice/MHA11.HA.phsAHar(7).cVal.mag.f

The resulting object reference to be used in a DataRef in a report is LDevice/MHA11\$MX\$HA\$phsAHar(7)\$cVal\$mag\$.

A shorter variableSpecification is illustrated in Figure 98.

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDevice</domainId>
      <itemId>MHAI1$MX$HA$phsAHar</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <index>7</index>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>cVal</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <component>mag</component>
                        </accessSelection>
                        <alternateAccess>
                          <CHOICE>
                            <unnamed>
                              <selectAlternateAccess>
                                <accessSelection>
                                  <component>f</component>
                                </accessSelection>
                                <alternateAccess/>
                              </selectAlternateAccess>
                            </unnamed>
                          </CHOICE>
                        </alternateAccess>
                      </selectAlternateAccess>
                    </unnamed>
                  </CHOICE>
                </alternateAccess>
              </selectAlternateAccess>
            </unnamed>
          </CHOICE>
        </alternateAccess>
      </selectAlternateAccess>
    </unnamed>
  </CHOICE>
</alternateAccess>

```

Figure 98 – Shorter VariableSpecification for LDevice/MHAI1.HA.phsAHar(7).cVal.mag.f

The resulting object reference to be used in a DataRef in a report is still LDevice/MHAI1\$MX\$HA\$phsAHar(7)\$cVal\$mag\$f.

However, the variableSpecification illustrated in Figure 99 is not conformant as it includes a componentName “mag\$f”, forbidden per 24.2.2.2.1.1.

```

<accessSelection>
  <component>phsAHar</component>
</accessSelection>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <index>7</index>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>cVal</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <component>mag$F</component>
                        </accessSelection>
                        <alternateAccess/>
                      </selectAlternateAccess>
                    </unnamed>
                  </CHOICE>
                </alternateAccess>
              </selectAlternateAccess>
            </unnamed>
          </CHOICE>
        </alternateAccess>
      </selectAlternateAccess>
    </unnamed>
  </CHOICE>
</alternateAccess>
</selectAlternateAccess>

```

Figure 99 – Non conformant VariableSpecification I

The variableSpecification illustrated in Figure 100 is not conformant as it includes a componentName “MX\$H\$phsAHar”, forbidden per 24.2.2.1.1.

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDevice</domainId>
      <itemId>MHAI1</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <component>MX$HA$phsAHar</component>
        </accessSelection>
      </selectAlternateAccess>
      <alternateAccess>
        <CHOICE>
          <unnamed>
            <selectAlternateAccess>
              <accessSelection>
                <index>7</index>
              </accessSelection>
            </selectAlternateAccess>
            <alternateAccess>
              <CHOICE>
                <unnamed>
                  <selectAlternateAccess>
                    <accessSelection>
                      <component>cVal</component>
                    </accessSelection>
                  </selectAlternateAccess>
                  <alternateAccess/>
                </CHOICE>
              </selectAlternateAccess>
            </alternateAccess>
          </CHOICE>
        </selectAlternateAccess>
      </unnamed>
    </CHOICE>
  </alternateAccess>
</CHOICE>
</alternateAccess>

```

Figure 100 – Non conformant VariableSpecification II

AlternateAccess specifications for ACSI reference LDevice/MHAI1.HA.phsAHar(7) [MX] are shown in Figure 101 and in Figure 102.

```

<accessSelection>
  <component>MX</component>
</accessSelection>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <component>HA</component>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>phsAHar</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <index>7</index>
                        </accessSelection>
                        <alternateAccess/>
                      </selectAlternateAccess>
                    </unnamed>
                  </CHOICE>
                </alternateAccess>
              </selectAlternateAccess>
            </unnamed>
          </CHOICE>
        </alternateAccess>
      </selectAlternateAccess>
    </unnamed>
  </CHOICE>
</alternateAccess>
</selectAlternateAccess>

```

Figure 101 – VariableSpecification for LDevice/MHA1.HA.phsAHar(7) [MX]

The resulting object reference to be used in a DataRef in a report is LDevice/MHAI1\$MX\$HA\$phsAHar(7).

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDevice</domainId>
      <itemId>MHAI1$MX$HA$phsAHar</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <index>7</index>
        </accessSelection>
        <alternateAccess/>
      </selectAlternateAccess>
    </unnamed>
  </CHOICE>
</alternateAccess>

```

Figure 102 – Shorter VariableSpecification for LDevice/MHAI1.HA.phsAHar(7) [MX]

The resulting object reference to be used in a DataRef in a report is still LDevice/MHAI1\$MX\$HA\$phsAHar(7).

24.2.2.2.1.4 VariableAccessSpecification

Table 108 defines the conformance of the variable access.

Table 108 – VariableAccessSpecification conformance statement

VariableAccessSpecification	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
listOfVariable	o	o		o	c1	
variableSpecification	o	o		o	c1	
alternateAccess	o	c3		o	c1	
variableListName	o	o		o	c2	
c1 Shall be 'm' if ACSI support for Logical Node Model is declared. c2 Shall be 'm' if ACSI support for ACSI DataSets, reporting, or logging is declared. c3 Shall be 'm' if ACSI support for ACSI DataSets, GetDataValue, Reporting, or logging is declared, as it depends on the type of the variable being accessed, however, alternateAccess is not mandatory to use in the SetDataValue request unless the variableAccess is an Array or an array element.						

24.2.2.2.1.5 VariableSpecification

Table 109 defines the conformance of the variable specification.

Table 109 – VariableSpecification conformance statement

VariableSpecification	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
name	o	o		o	m	
address	o	o		o	i	
variableDescription	o	o		o	i	
scatteredAccessDescription	o	x		o	x	
invalidated	o	x		o	x	

24.2.2.2.2 Read

Table 110 defines the conformance of the read service.

Table 110 – Read conformance statement

Read	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
specificationWithResult	o	m		o	m	
variableAccessSpecification	m	m	See 24.2.2.2.1.4	m	m	See 24.2.2.2.1.4
Response						
variableAccessSpecification	o	o		o	m	
listOfAccessResult	m	m		m	m	

24.2.2.2.3 Write

Table 111 defines the conformance of the write service.

Table 111 – Write conformance statement

Write	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
variableAccessSpecification	m	m	See 24.2.2.2.1.4	m	m	See 24.2.2.2.1.4
listOfData	m	m		m	m	
Response						
failure	m	m		m	m	
success	m	m		m	m	

24.2.2.2.4 InformationReport

Table 112 defines the conformance of the InformationReport service.

Table 112 – InformationReport conformance statement

InformationReport	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
variableAccessSpecification	m	m	See 24.2.2.2.1.4	m	m	See 24.2.2.2.1.4
listOfAccessResult	m	m		m	m	

24.2.2.2.5 GetVariableAccessAttributes

Table 113 defines the conformance of the GetVariableAccessAttributes service.

Table 113 – GetVariableAccessAttributes conformance statement

GetVariableAccessAttributes	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
name	o	o		m	m	
address	o	i		m	i	
Response						
mmsDeletable	m	m		m	m	
address	o	i		o	i	
typeDescription	m	m		m	m	

24.2.2.2.6 DefineNamedVariableList

Table 114 defines the conformance of the DefineNamedVariableList service.

Table 114 – DefineNamedVariableList conformance statement

DefineNamedVariableList	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
variableListName	m	m		m	m	
listOfVariable	m	m		m	m	
variableSpecification	m	m		m	m	
alternateAccess	o	o		o	m	
Response	m	m		m	m	

24.2.2.2.7 GetNamedVariableListAttributes

Table 115 defines the conformance of the GetNamedVariableListAttributes service.

Table 115 –GetNamedVariableListAttributes conformance statement

GetNamedVariableListAttributes	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
ObjectName	m	m		m	m	
Response						
mmsDeletable	m	m		m	m	
listOfVariable	m	m		m	m	
variableSpecification	m	m		m	m	
alternateAccess	o	m		o	c	
c Shall be 'm' if MMS variable of type Array are implemented in the object model of the server.						

24.2.2.2.8 DeleteNamedVariableList

Table 116 defines the conformance of the DeleteNamedVariableList service.

Table 116 – DeleteNamedVariableList conformance statement

DeleteNamedVariableList	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
Scope	m	m		m	m	
listOfVariableListName	m	m		m	m	
domainName	o	i		o	i	
Response						
numberMatched	m	m		m	m	
numberDeleted	m	m		m	m	
DeleteNamedVariableList-Error	m	m		m	m	

24.2.2.3 Journal management services – ReadJournal

Table 117 and Table 118 define the conformance of the ReadJournal service.

Table 117 – ReadJournal conformance statement

ReadJournal	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/ range
Request						
journalName	m	m		m	m	
rangeStartSpecification	o	m		o	m	
startingTime	o	c1		o	m	
startingEntry	o	c1		o	m	
rangeStopSpecification	o	m		o	m	

ReadJournal	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/ range
endingTime	o	c2		o	m	
numberOfEntries	o	c2		o	m	
ListOfVariable	o	i		o	i	
EntryToStartAfter	o	m		o	m	
TimeSpecification	m	m		m	m	
EntrySpecification	m	m		m	m	
Response						
listOfJournalEntry	m	m		m	m	
entryIdentifier	o	m		m	m	See explanation for EntryID in 17.3.3.4.1
originatingApplication	m	i		m	i	the value shall be ignored by the client
entryContent	m	m		m	m	
moreFollows	m	m		o	m	
c1 At least one shall be present. c2 At least one shall be present.						

Table 118 – EntryContent conformance statement

Ref	Parameter	Client-CR			Server-CR		
		Base	F/S	Value/range	Base	F/S	Value/range
1	occurrenceTime	m	m		m	m	
2	additionalDetail	x	x		x	x	
3	entryForm	m	m		m	m	
4	data	o	m		o	c	
5	event	o	m		o	o	
6	eventConditionName	o	m		o	m	
7	currentState	o	m		o	m	
8	listofVariable	o	m		o	m	
9	variableTag	o	m		o	m	
10	valueSpecification	o	m		o	m	
11	annotation	o	m		o	c	
c Either data or annotation shall be present.							

24.2.2.4 File management services

24.2.2.4.1 FileDirectory

Table 119 defines the conformance of the FileDirectory service.

Table 119 – FileDirectory conformance statement

FileDirectory	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
filespecification	o	o		m	m	
continueAfter	o	o		m	m	
Response+						
listOfDirectoryEntry	m	m		m	m	
MoreFollows	m	m		m	m	

24.2.2.4.2 FileOpen

Table 120 defines the conformance of the FileOpen service.

Table 120 – FileOpen conformance statement

FileOpen	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
filename	m	m		m	m	
initialPosition	o	o		m	m	
Response+						
frsmID	m	m		m	m	
fileAttributes	m	m		m	m	

24.2.2.4.3 FileRead

Table 121 defines the conformance of the file read service.

Table 121 – FileRead conformance statement

FileRead	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
frsmID	m	m		m	m	
Response+						
fileData	m	m		m	m	
moreFollows	m	m		m	m	

24.2.2.4.4 FileClose

Table 122 defines the conformance of the FileClose service.

Table 122 – FileClose conformance statement

FileClose	Client-CR			Server-CR		
	Base	F/S	Value/range	Base	F/S	Value/range
Request						
frsmID	m	m		m	m	
Response+	m	m		m	m	

24.3 PICS Statement

24.3.1 General

This subclause describes the Protocol Implementation Conformation Statement Proforma (PICS). Every implementor shall complete the entire PICS. Refer to IEC 61850-7-2 ACSI basic conformance statement.

The PICS, in Subclause 25.3.2, shall also be completed.

24.3.2 Substation configuration language

Conforming implementations shall support the substation configuration language as defined in IEC 61850-6 for exchange between engineering tools. Implementors that wish to provide online access and management for SCL configuration should refer to informative Annex F for guidance.

25 Substation Configuration Language (SCL)

For the purpose of this SCSM, the SCL ConnectedAP element shall be extended. The XML Schema definition of IEC 61850-6 shall be used. Table 123 defines the P-Types that are allowed for the xs:element of "Address".

Table 123 – Allowed P-Type definitions for client/server addressing

P-Type designation	Description	ICD m/o	SCD m/o	Restrictions/comments
JID	Jabber ID	o	o	See RFC 6122. As the resource part is constrained by the present SCSM, it shall be either compliant or not present

Annex A (normative)

Communication stack

A.1 Overview

A.1.1 General

This annex describes a layered approach of the communication stack for people familiar with IEC 61850-8-1, MMS and the OSI Reference Model (ISO/IEC 7498-1). However reading this annex is not required to understand the other clauses of IEC 61850-8-2, except Clause 24.

The purpose of IEC 61850-8-2 is to provide detailed instructions/specifications as to the mechanisms and rules required to implement the core ACSI services specified in IEC 61850-7-2, while making use of XMPP, as well as the usage of SNTP for time synchronization (see Figure A.1).

The communication requirements for utilities (listed in IEC 61850-5) are met by the profiles shown in Figure A.1. The message types listed below (specified in IEC 61850-5) are mapped as shown in Figure A.1, except Type 1, 1A and 4 that are out of scope of this SCSM.

- Type 1 (Fast messages)
- Type 1A (Trip)
- Type 2 (Medium speed messages)
- Type 3 (Low speed messages)
- Type 4 (Raw data messages)
- Type 5 (File transfer functions)
- Type 6 (Time synchronisation messages)

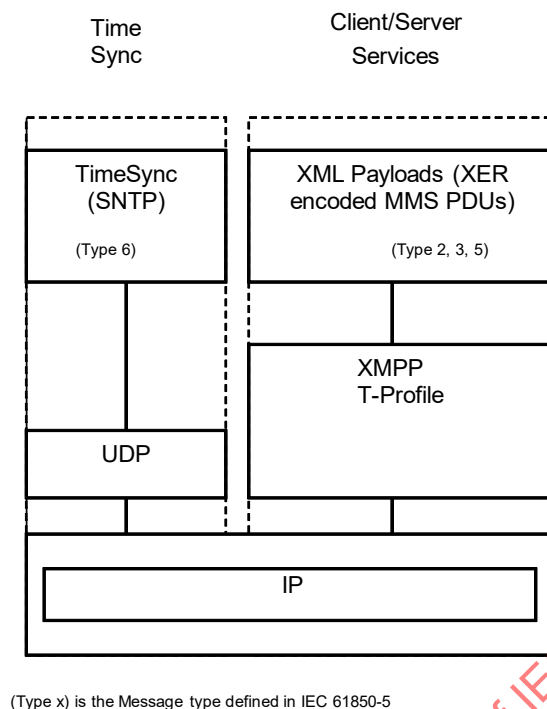


Figure A.1 – Overview of functionality and profiles

Further details pertaining to the use and actual specification of each profile may be found in Clause 6.

NOTE The stack profiles described here start at the IP layer. Which specifications must be used for the layers below IP is out of the scope of this SCSM.

A.1.2 XMPP communication profiles

ISO has recognized the existence of Application Profiles (A-Profiles) and Transport Profiles (T-Profiles). A-Profiles represent the protocols and agreements in regards to the upper 3 layers of the OSI Reference Model (ISO/IEC 7498-1). The T-Profiles represent the protocols and agreements in regards to the lower 4 layers of the OSI reference model.

For consistency with the other parts of IEC 61850 which already refer to A-Profiles and T-Profiles, this document splits also the description of the communication stack into two parts. However, for the purpose of this document, each communication profile includes only one possible A-Profile and one possible T-Profile which must be used together.

The profile for the Client/Server ACSI Services consists in a A-Profile based on MMS (ISO 9506), like the IEC 61850-8-1, but with an XML encoding called XER, and a T-Profile based on XMPP.

A.1.3 Non-XMPP communication profiles

The time synchronization services specified in IEC 61850-7-2, have been intentionally mapped to application protocols and communication profiles that make use neither of ISO 9506 nor of XMPP. These services are based on the IETF Simple Network Time Protocol (SNTP).

A.2 Communication stack

A.2.1 Overview of the protocol usage

The OSI Reference Model (ISO/IEC 7498-1) details a model based upon the concept of layering of communication functionality. The model details 7 layers and details the functional requirements, for each layer, in order to achieve a robust communication system. The model does not specify the protocols to be used to achieve the functionality, nor does it restrict the solution to be a single set of protocols.

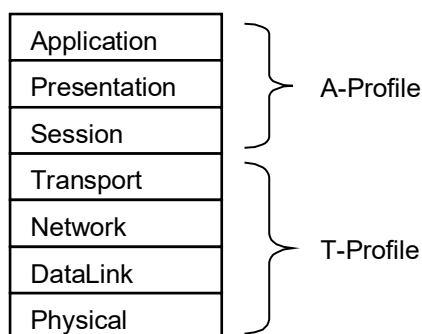


Figure A.2 – OSI reference model and profiles

The use of ISO Application (A-Profile) and Transport (T-Profile) Profiles (see Figure A.2) describes the various stack profiles. An ISO A-Profile is the set of specifications and agreements relating to the upper 3 layers of the ISO OSI reference model (e.g. the layers of application, presentation, and session). An ISO T-Profile is the set of specifications and agreements relating to the lower 4 layers of the ISO OSI reference model (e.g. the layers of transport, network, DataLink and physical).

Various combinations of A-Profiles and T-Profiles can be generally combined in order to allow certain types of information/services to be exchanged. In the context of this SCSM, the services, as specified in IEC 61850-7-2, are mapped into only 2 different combinations of A- and T- profiles. The two combinations are used for:

- Client/server services (see Figure A.1).
- Time Sync.

A.2.2 Client/server services and communication profiles

A.2.2.1 Client/server services

The client/server communication profile shall be used for any implementation claiming conformance to this standard and declaring support for one of the IEC 61850-7-2 services shown in Table 1.

A.2.2.2 A-Profile

Services and protocols of the A-Profile client/server shall be as shown in Table A.1.

Table A.1 – Service and protocols for client/server communication A-Profile

OSI model layer	Specification			m/o
	Name	Service specification	Protocol specification	
Application	Manufacturing Message Specification	ISO 9506-1:2003	ISO 9506-2:2003	m
	End-to-end security	IEC 62351-6		m
Presentation	Abstract Syntax	ISO/IEC 8824-1:2008	ITU X.693 (XER)	m
	Mapping of M-Services to E2E ^a		See Annex D	m
Session	Association context	IEC 62351-4		m
^a Association Context provides an alternative mapping for M-Services of ISO 9506-2 to end-to-end security and XMPP				

As described in Table A.1, the application layer of the A-Profile uses MMS, which is de facto a shared specification with the SCSM 8-1. More specifically, the principle of mapping the ACSI objects and services over the corresponding concepts of MMS described in IEC 61850-8-1 have been reused for the present SCSM 8-2.

End-to-end security is also a shared mechanism with the SCSM 8-1 which is described in IEC 62351-4. Application for the present SCSM is described in Clause 7.

For the presentation layer, note in particular that the encoding used for ASN.1 is not the same as in SCSM 8-1: the SCSM 8-2 will use the extended XML Encoding Rules defined by ITU X.693 XER rather than the binary encoding used by the SCSM 8-1 and defined by ISO 8825 BER. The normative XML Schema Definition for the XER encoded MMS PDUs is provided in Annex G.

Finally, the session layer manages the two party associations between two communicating applications. During the Associate service, both the calling and called entities shall generate an AssociationID that is locally unique within the pairing of JIDs. This AssociationID shall then be part of all exchanged messages between the two associated entities.

A.2.2.3 XMPP T-Profile

Table A.2 shows services and protocols of the XMPP T-Profile client/server.

Table A.2 – Service and protocols for client/server XMPP T-Profile

OSI Model Layer	Specification			m/o
	Name	Service specification	Protocol specification	
Transport	XMPP	RFC 6120 , RFC 6121, RFC 6122, XEP-0198, XEP-0199		m
	SASL	RFC 4422		m
	TLS	RFC 5246, IEC 62351-6		m
	Transmission Control Protocol (TCP)	RFC 793		m
Network	Internet Control Message Protocol (ICMP)	RFC 792		m
	Internet Protocol	RFC 791		c1,c2
	IPv6	RFC 2460		c1,c2
c1 shall be one or both for XMPP clients				
c2 mandatory for XMPP servers				

NOTE 1 Layers below the network layer are out of the scope of this SCSM.

NOTE 2 For guidelines about the usage of IPv6 see IEC TR 62357-200.

The details of how XMPP, SASL and TLS are used in the context of the present SCSM are explained in Clause 6 for XMPP and more specifically in 6.2.2 for SASL and TLS.

A.2.3 Time sync

A.2.3.1 Conformance statement

This communication profile shall be used for any implementation claiming conformance to this standard and declaring support for objects containing an attribute of type Timestamp.

A.2.3.2 A-Profile

A.2.3.2.1 General

The A-Profile for Time Sync Services shall be as shown in Table A.3.

Table A.3 – Time sync A-Profile

OSI model layer	Specification			m/o
	Name	Service specification	Protocol specification	
Application	Simple Network Time Protocol	RFC 5905		m
Presentation				
Session				

A.2.3.2.2 Implementation agreements

This A-Profile shall conform to the agreements specified in RFC 6864 and RFC 7766.

The A-Profile conformity to RFC 5905 is limited to the SNTP subset of NTP.

Mode 3 of SNTP time synchronization shall be supported for SNTP Clients.

Mode 4 of SNTP time synchronization shall be supported for SNTP Servers.

Mode 3, 4 of SNTP time synchronization shall be supported for SNTP Clients and Servers.

A.2.3.3 T-Profile

A.2.3.3.1 General

The T-Profile for Time Sync Services shall be as shown in Table A.4.

Table A.4 – Time sync T-Profile

OSI model layer	Specification			m/o
	Name	Service specification	Protocol specification	
Transport	Internet Control Message Protocol (ICMP)	RFC 792		m
	User Datagram Protocol (UDP)	RFC 768		m
Network	Internet Protocol	RFC 791 – IPv4		m
		RFC 2460 – IPv6		o
	An Ethernet Address Resolution Protocol (ARP)	RFC 826		m
	Broadcasting Internet Datagrams	RFC 922 RFC 950 RFC 919		m
	Host Extensions for IP Multicasting	RFC 1112		m

NOTE Layers below the network layer are out of the scope of this SCSM.

A.2.3.3.2 Implementation agreement for SNTP

The following implementation agreement applies to implementations claiming conformance with this standard.

The UDP port number assigned by the IANA to NTP is 123. The SNTP client shall use this value in the UDP Destination Port field for client request messages.

Annex B (informative)

Deployment of XMPP infrastructure

B.1 General

There are several ways to deploy XMPP in a communication environment. For instance, multi domains can prefer the use of federation or not. Outage of the communication path can also be taken care of when deploying the infrastructure. This informative annex gathers different aspects of XMPP infrastructure deployment to be considered. Both public and private XMPP infrastructures are allowed. However, depending on the requirements the deployment on a private infrastructure may be preferred.

B.2 Deployment of XMPP within one XMPP domain

B.2.1 Use case facility

The facility domain can be deployed using an XMPP communication environment as displayed in Figure B.1. The facility domain is responsible for maintaining the XMPP server of facil.org; the XMPP server can be located in a LAN or within a WAN depending on the communication network implementation.

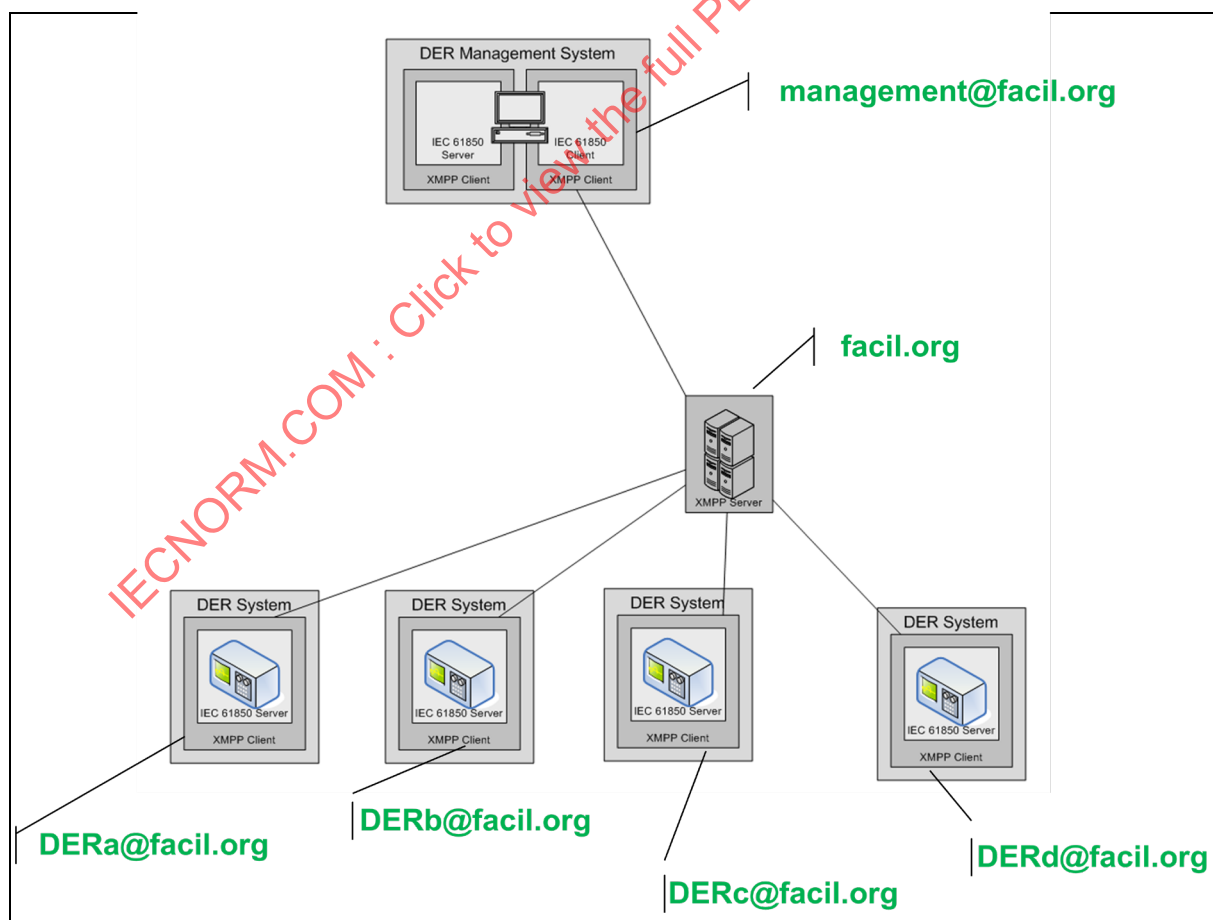


Figure B.1 – Facility domain

Figure B.1 also illustrates the deployment of JIDs for the devices members of a facility.

All facility members have a JID belonging to the domain “facil.org”.

B.2.2 Use case hierarchy within a facility

Within a facility, a DER management system can be configured to group different resources (for instance to group all resources located in a building). The resulting single virtual resource (for instance the building) will be then integrated at another DER management system located at a higher hierarchical level of the facility.

Figure B.2 illustrates the use case of the hierarchical aggregation where DERA and DERb are being aggregated within the facility to DERab. The IEC 61850 Client of the DER MS (“building1@facil.org”) connects to the resources DERA, DERb while the IEC 61850 Server of the DER MS (“DERab@facil.org”) virtualizes both resources.

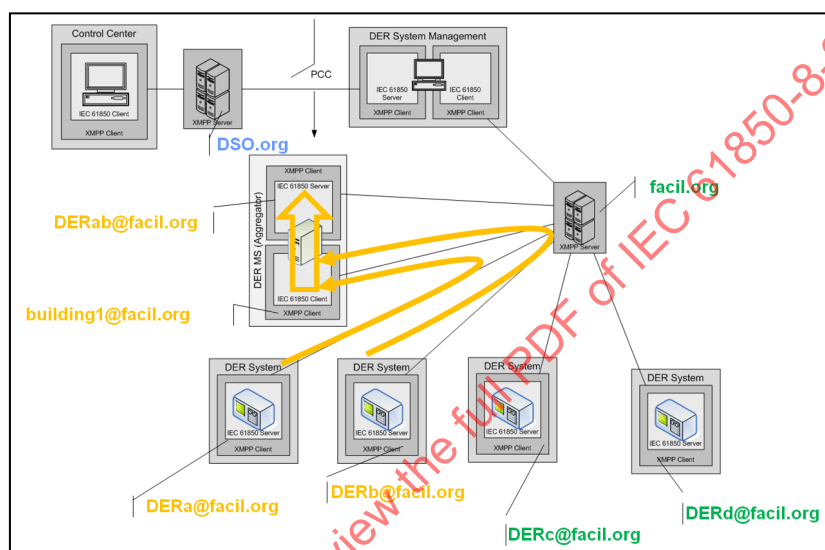


Figure B.2 – Hierarchical Aggregation at facility

Finally all resources are being aggregated at the DER management system located at the highest level of the facility. (“management@facil.org”) shown in Figure B.3.

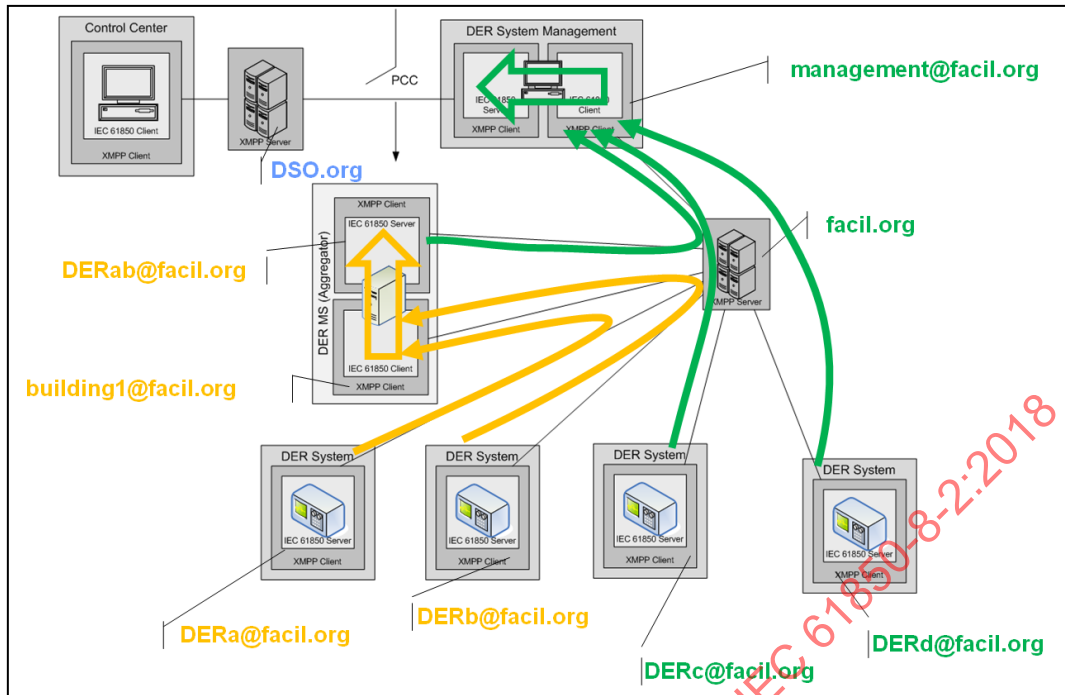


Figure B.3 – DER Management System at facility

B.3 Deployment of XMPP while interconnecting more than one XMPP domain

B.3.1 Interconnection of XMPP Domain

B.3.1.1 Use case Facility at DSO

Figure B.4 illustrates the connection of the facility using the domain “facil.org” to the DSO: using the domain DSO.org. The facility resource is virtualized at the IEC 61850 Server with the JID “management_facil@DSO.org”, i.e. located in the domain of the DSO.

The IEC 61850 Client located at the Control Center (“management@DSO.org”) can:

- neither see,
- nor connect to,
- nor monitor,
- nor subscribe the presence of

the individual resources DERa,b,c,d of the facility (neither the intermediate aggregated DERab), but operate and monitor the IEC 61850 Server representing the whole facility available at the (“management_facil@DSO.org”).

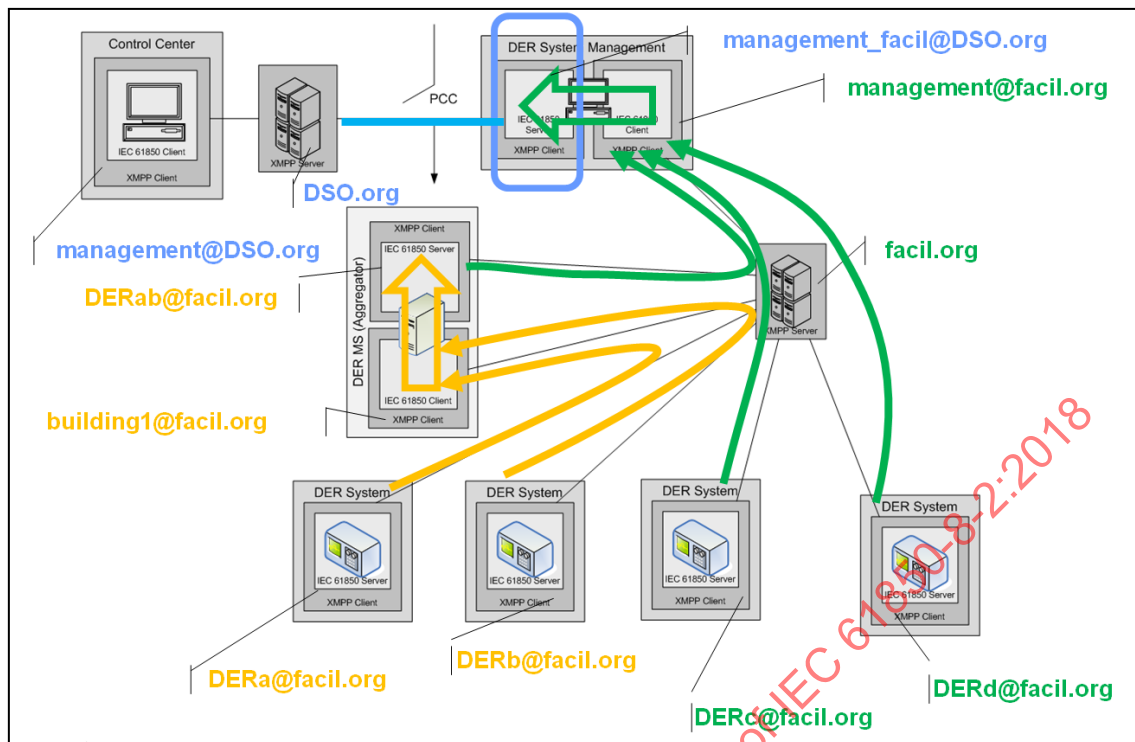


Figure B.4 – Facility Management Integration at DSO

B.3.1.2 Use case Multiple facilities at DSO

Figure B.5 illustrates that any hierarchical aggregation is possible. Colors delimit the domain identifiers used for the JID deployments. The Control Center of the DSO can get access to DER management Systems, which aggregates resources, and so on. The red arrow show the path from the DSO Control Center to a facility resource over multiple DER Management system. The Control Center cannot see the resource, but see the virtualized resource over multiple intermediate DER Management systems.

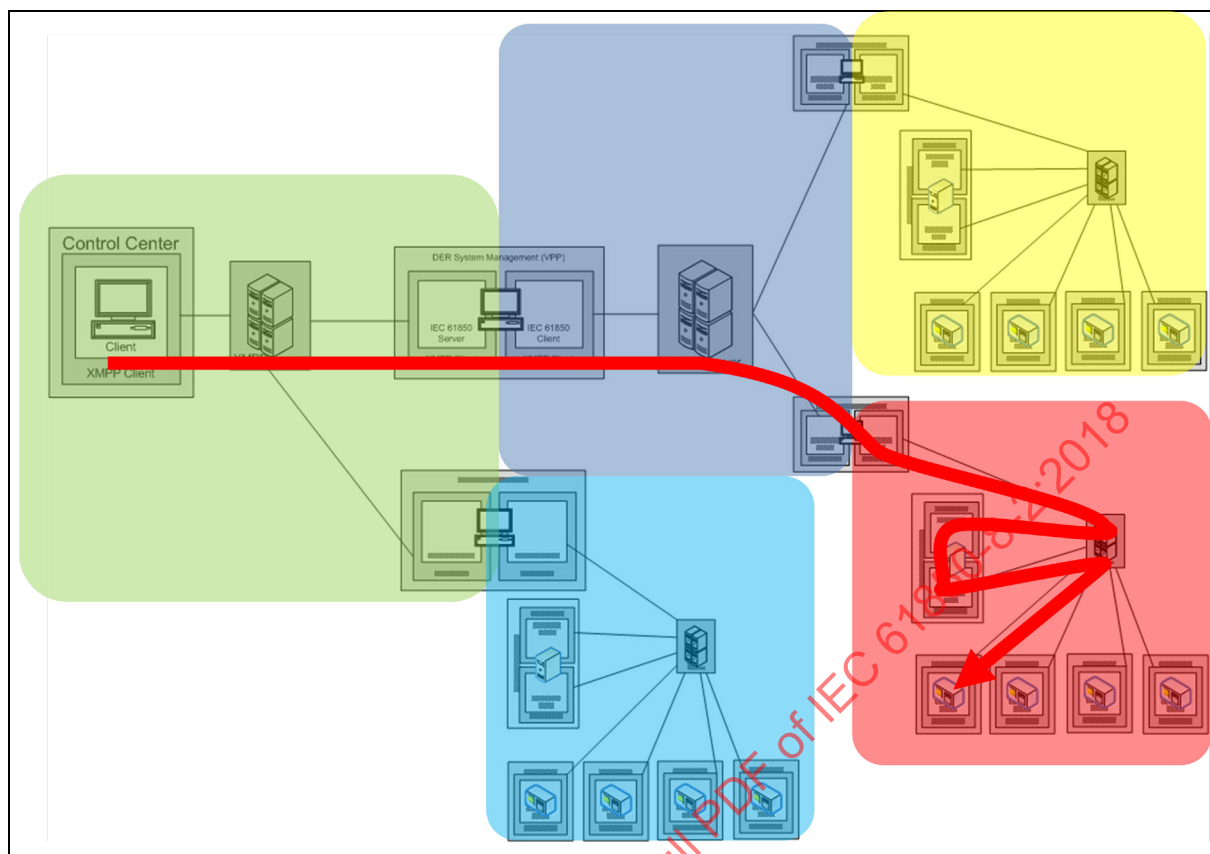


Figure B.5 – Multiple facilities at DSO

B.3.1.3 Use case VPP communications

The domains involved in the use case displayed in Figure B.6 are:

- DSO.org – the domain of a DSO.
- VPP.net – the domain of the virtual power plan having contracts with resources.

The Control Center of the DSO is being deployed with as many domain specific JIDs as domains it needs to interact with.

The Control Center of the VPP is being deployed with as many domain specific JIDs as domains it needs to interact with.

The blue lines delimit the DERs that are contracted by the VPP.net. The DERs can have either one JID (belonging to VPP.net) or 2 JIDs (belonging to VPP.net and to DSO.org).

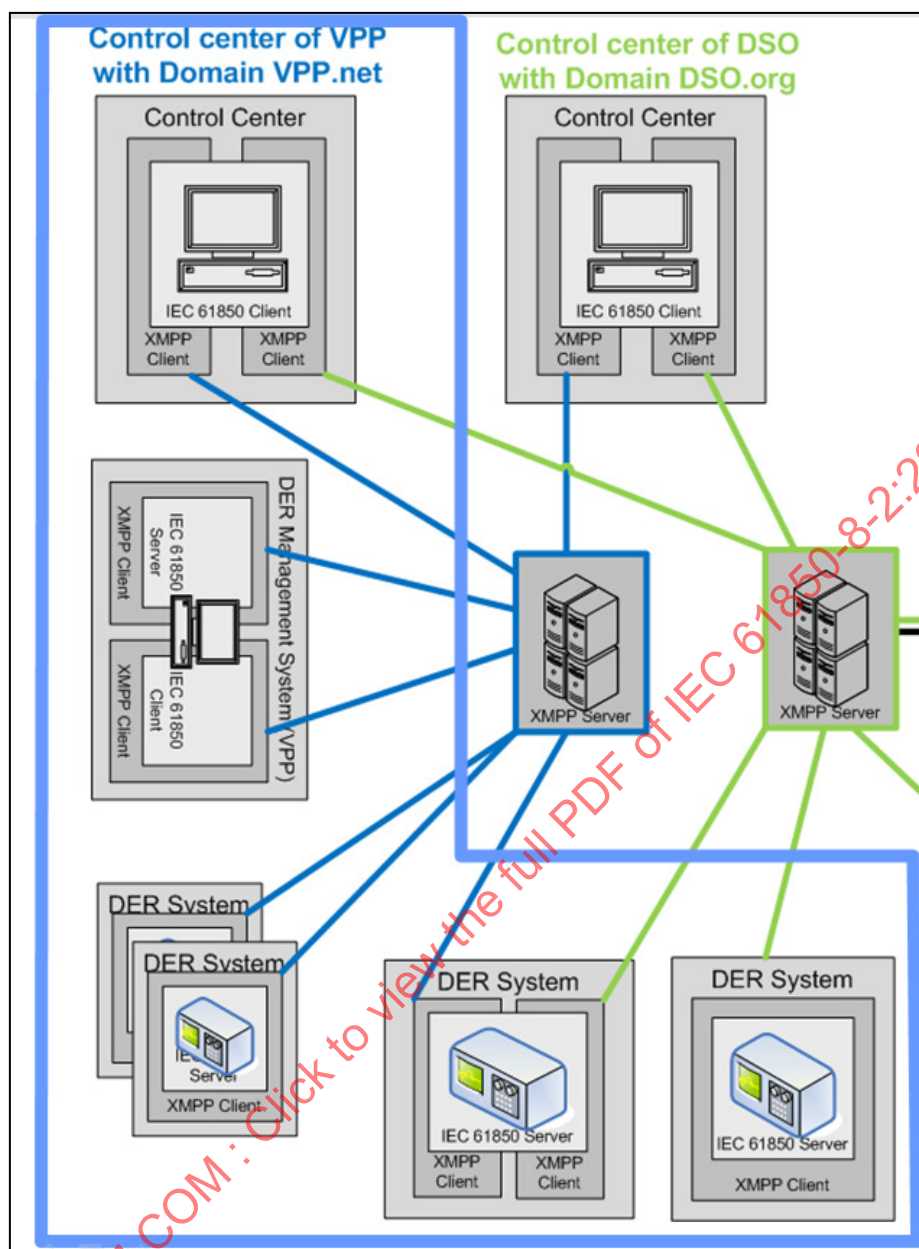


Figure B.6 – VPP and contracted DERs

B.3.1.4 Use case VPP communication to DERs@VPP

Figure B.7 and Figure B.8 illustrate the communication logical paths involved when the VPP Control Center communicates with VPP.net DERs.

Indirect Control illustrates the path from the Control Center down to the resource over the intermediate DER Management System.

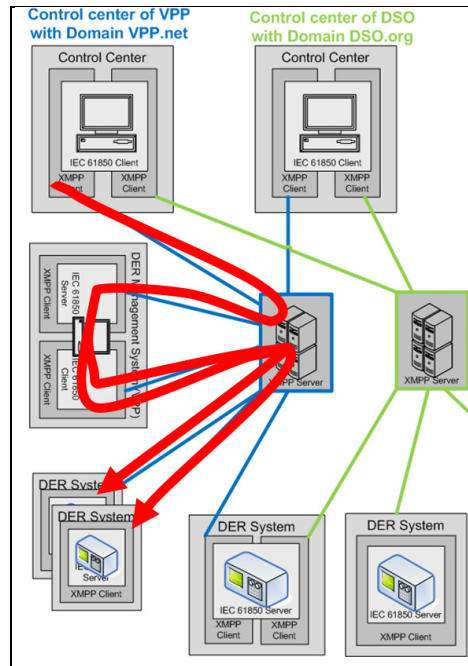


Figure B.7 – indirect control using VPP JIDs

Control illustrates the path from the Control Center down to the resource not managed by an intermediate DER Management System.

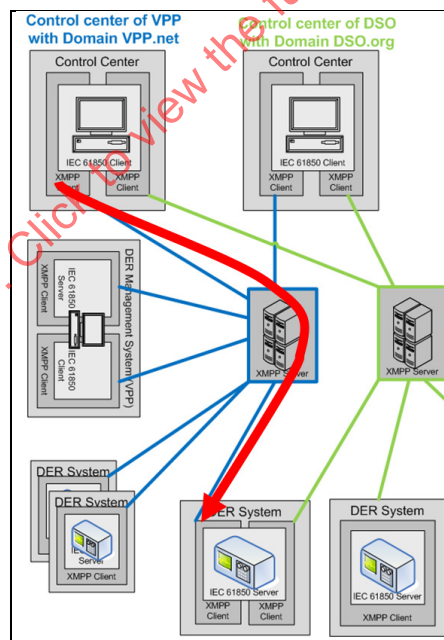


Figure B.8 – VPP direct control using VPP JIDs

B.3.1.5 Use case VPP communication to DERs@DSO

Figure B.9 illustrates the communication logical paths involved when the VPP Control Center communicates with DSO.org members. The path can only be established if VPP Control Center has been configured with a JID belonging to a DSO.org domain, additionally to its JID belonging to VPP.net.

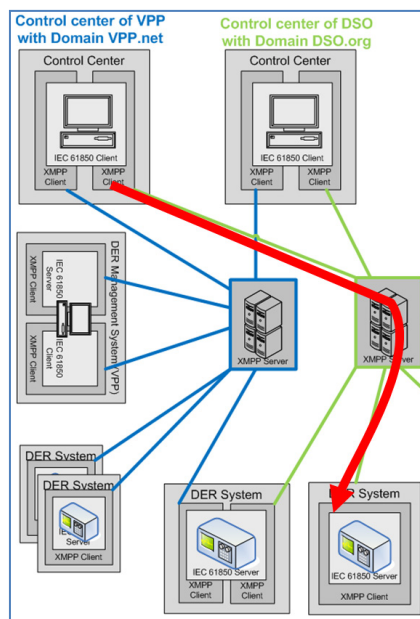


Figure B.9 – VPP direct control using DSO JIDs

B.3.1.6 Use case DSO communication to DERs@VPP

Figure B.10 illustrates the communication logical paths involved when the Control Center of DSO communicates with VPP.net members. The path can only be established if DSO Control Center has been configured with a JID belonging to a VPP.net domain, additionally to its JID belonging to DSO.org.

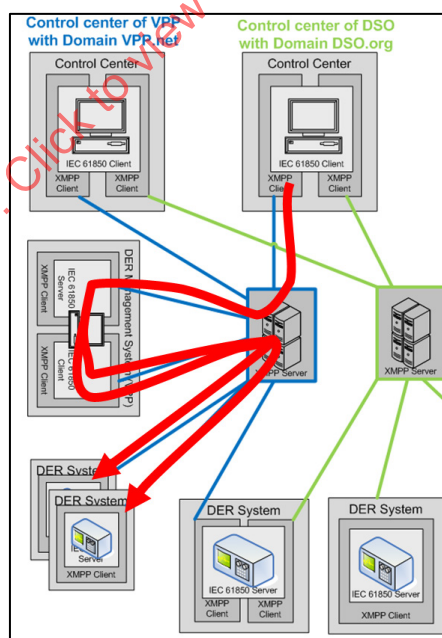


Figure B.10 – DSO Indirect control with VPP JIDs

B.3.1.7 Use case DSO communication to DERs@DSO

Figure B.11 illustrates the communication logical paths involved when the DSO Control Center communicates with DSO.org DERs.

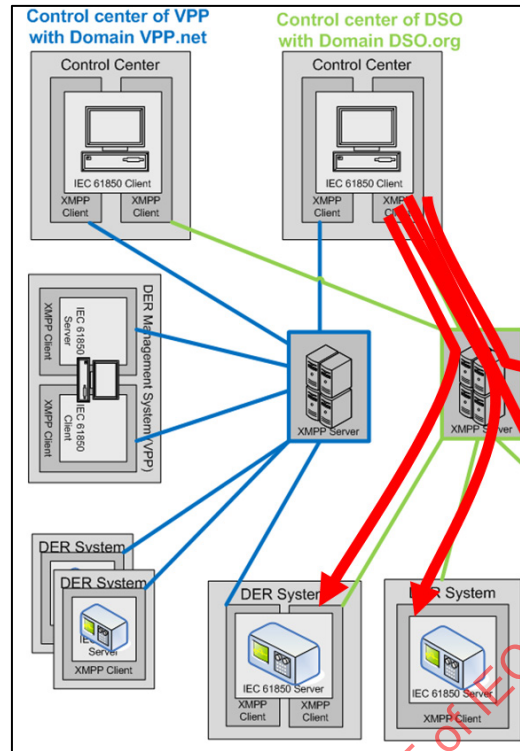


Figure B.11 – DSO direct control

B.3.1.8 Multiple JIDs

B.3.1 illustrates the deployment of multiple JIDs to achieve inter-domain communication. Either the Control Center implements multiple JIDs and / or the DER resource themselves. Such architecture has impact on the use of the bandwidth used by the DER resources since:

- maintaining two streams will be necessary on the single physical link,
- presence stanza, ROSTER information, will be send twice over the physical link (one to each XMPP server),
- XMPP mechanism such as PubSub (Publish – Subscribe – see XEP 0060) can not be used to decrease the bandwidth consumption (the monitoring information cannot be shared with mean of a pub sub node with multiple IEC 61850 Clients).

B.3.2 Definition of a federation communication between XMPP domains

Federation is a capability given to XMPP Clients located in different domains, and therefore connected to different XMPP servers, to communicate with each other. The XMPP server resolves the domain, indicated in the destination field in a stanza, starts a XML stream to the XMPP server of the target domain, and uses that connection for forwarding the message as shown in Figure B.12.

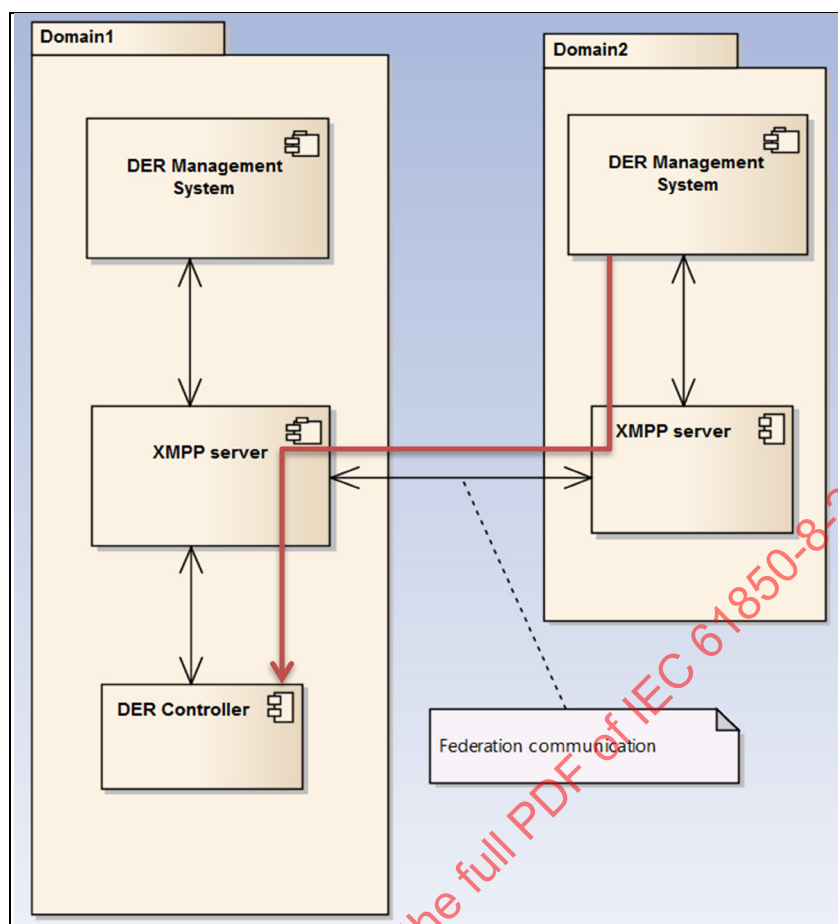


Figure B.12 – Concept of federation in XMPP

B.3.3 Interconnection of Domain with federation

B.3.3.1 Use case Facility at DSO

Figure B.13 illustrates the use of federation in the multi-domain communication DSO.org – facility.org.

Using federation allows the DSO Control Center management@DSO.org to communicate with the virtualized facility resource located at management_facil@facil.org, but possibly also with the single DERx depending of the deployed RBAC.

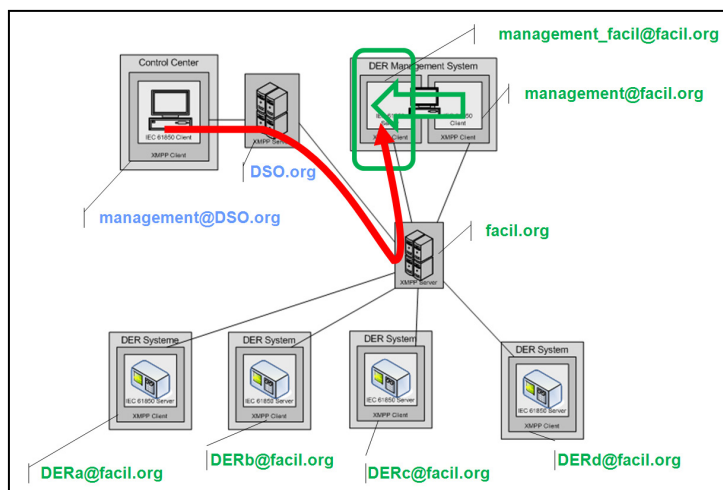


Figure B.13 – Federation DSO – Facility

B.3.3.2 Use case VPP communications

Figure B.14 illustrates the second set of VPP communication using federation with the DSO (first set was presented in B.3.1.3).

The black lines delimit the DERs that are contracted by the VPP.com. The DERs will either have a JID belonging to the domain VPP.com or to the domain DSO.org.

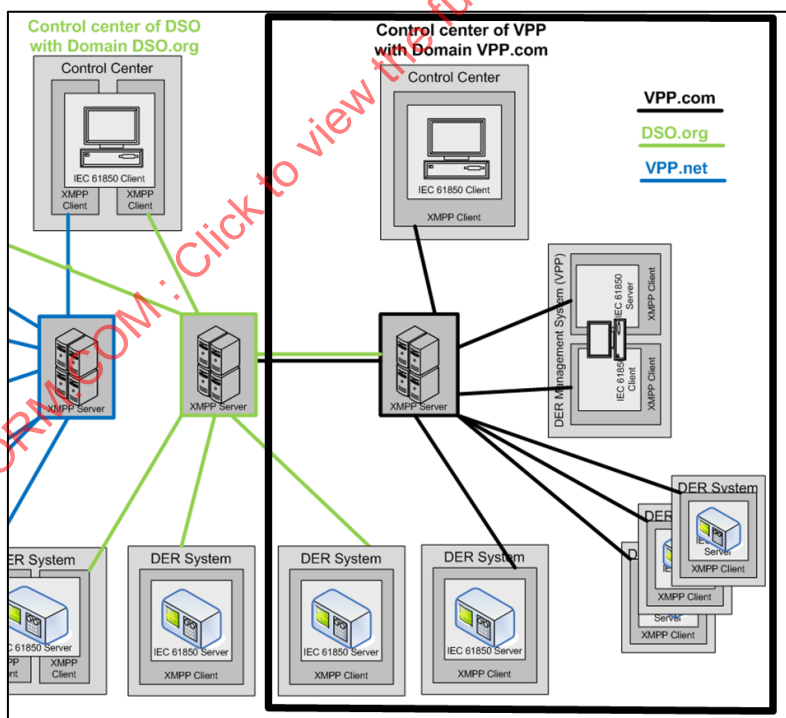


Figure B.14 – Use of federation with VPP

B.3.3.3 Use case VPP communication to DER@VPPs

Figure B.15 and Figure B.16 illustrates the logical paths between the VPP Control Center and the DERs belonging to the same domain.

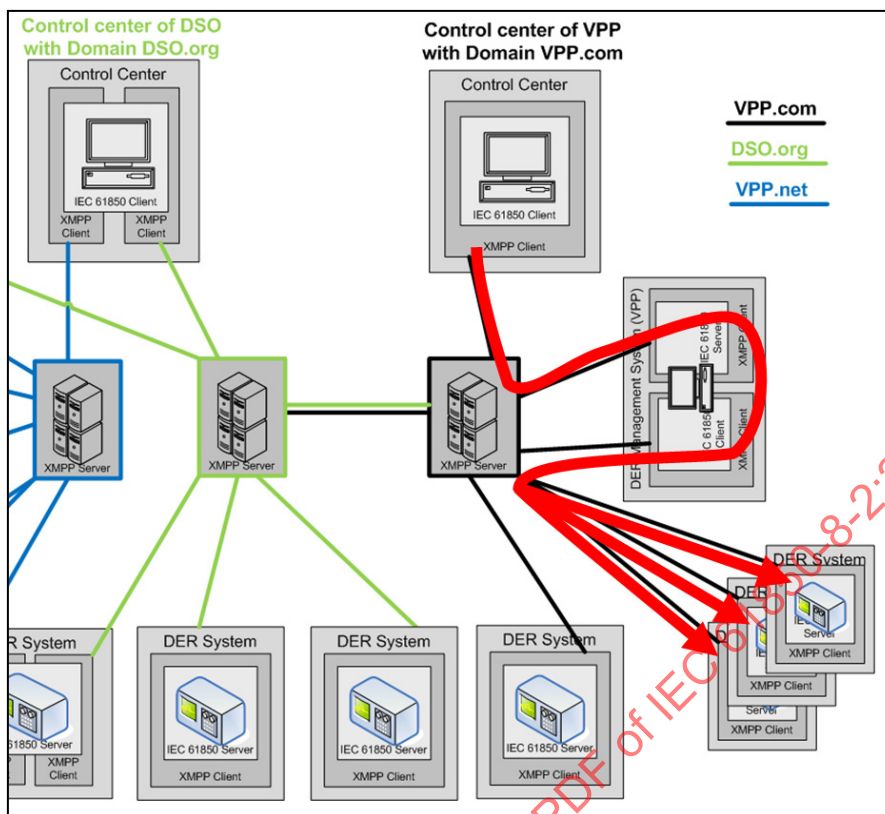


Figure B.15 – communication with VPP JIDs – indirect control

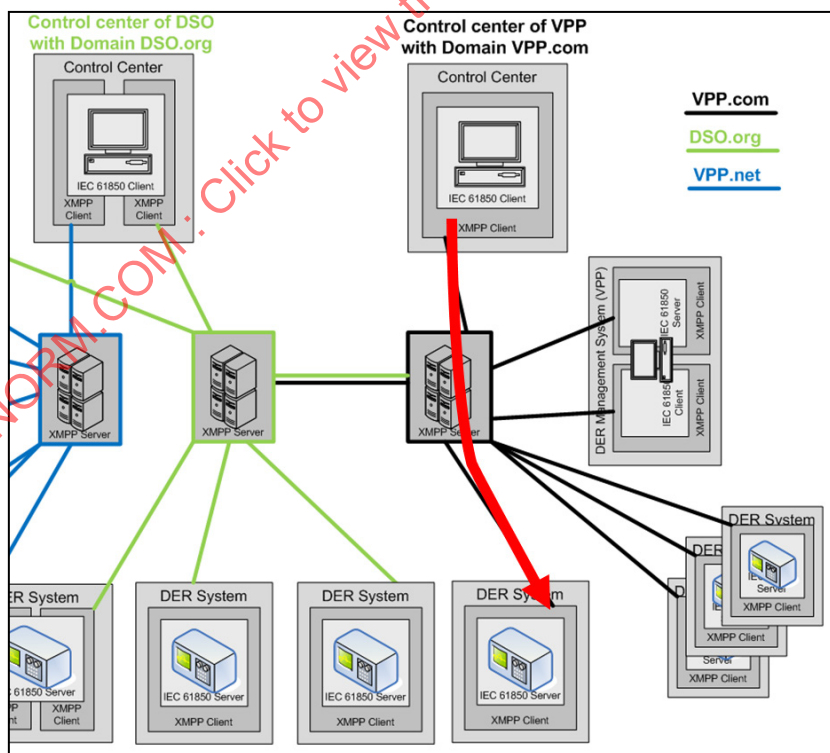


Figure B.16 – VPP communication with VPP JIDs – direct control

B.3.3.4 Use case VPP communication to DERs@DSO

Figure B.17 illustrates the federation deployment to realize the use case. Communication between management@VPP.com and DER@DSO.org will transit over federation between the domain servers VPP.com and DSO.org.

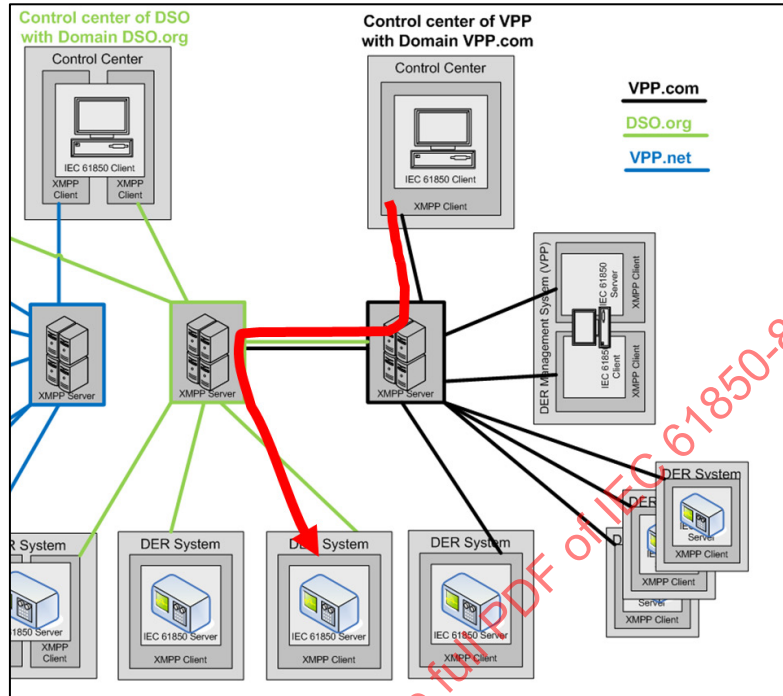


Figure B.17 – VPP communication with DSO JIDs

B.3.3.5 Use case DSO communication to DERs@VPP

Figure B.18 illustrates the federation deployment to realize the use case. Communication between management@DSO.org and DER@VPP.com will transit over federation between the domain servers VPP.com and DSO.org.

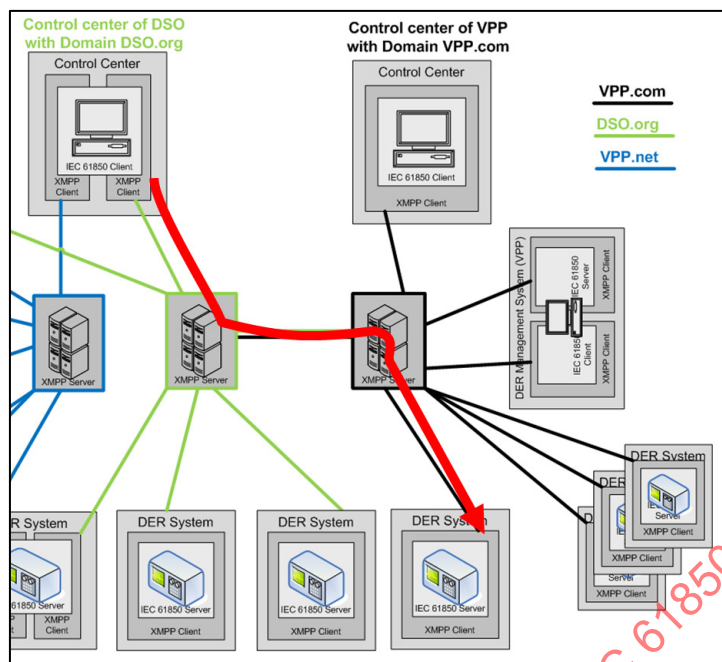


Figure B.18 – DSO Communication with VPP JIDs

B.4 Communication path outage and recovery

XMPP server can be clustered with means of several connection managers. The connection managers of a domain are the entities the DER resource will connect to when establishing the XMPP stream.

Discovering the location of the XMPP server / connection managers responsible for a domain can be done statically with a pre configured IP address, or with use of DNS with SRV records. In that case, depending on the deployment of the communication managers, and the sensibility to the communication channel availability, the service records can return a weighted list of connection managers. The XMPP client can start working the prioritized list of the connection managers until a connection is granted. In case of a communication path outage, using another entry in the list will force the use of another path for the communication.

Annex C (informative)

Security for DER integration based on XMPP

C.1 General

NOTE This annex mentions the case of multicast communications because XMPP can provide this feature, typically by using the Multi User Chat extension. However only unicast communications are in the current scope of the SCSM 8-2.

Security requirements targeting the integration of DER into a power system architecture are typically derived from a given system architecture and use cases describing the interactions of the components building the system architecture including the DER.

These security requirements are used to determine the appropriate security measures, which may be technical and/or organizational security measures.

Figure C.1 is a generic picture for the scenario used to discuss security requirements. In the upper part of the figure, the control side is depicted, and in the lower part of the figure the load and generator side. Note that the control side may comprise different actors, among them the DSO, the market operator or other grid related operators.

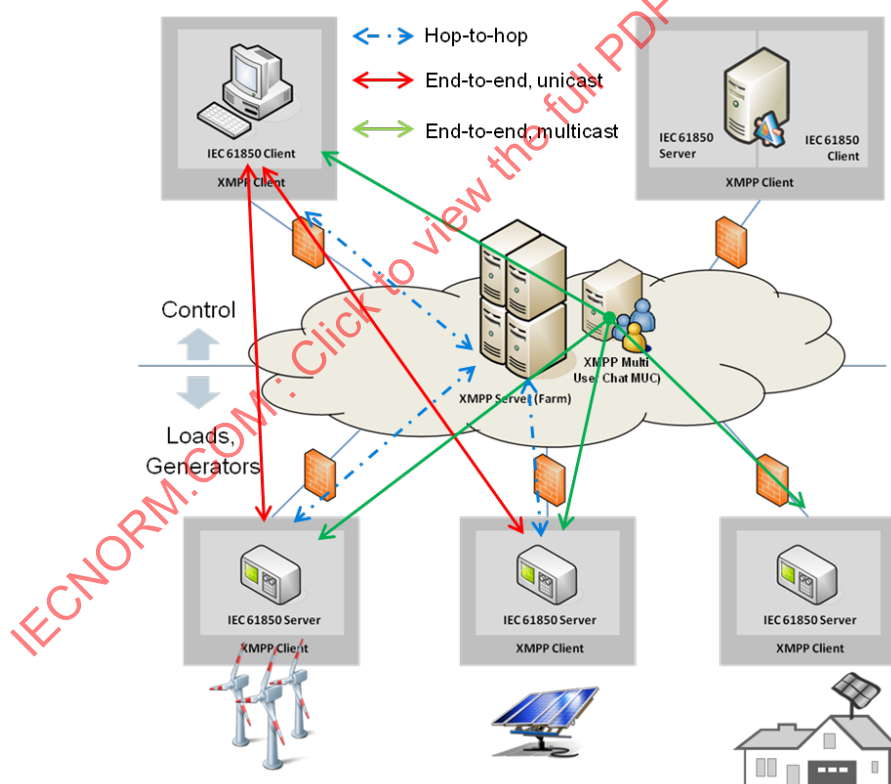


Figure C.1 – Base system for discussion of IT security requirements

As seen in Figure C.1 the prosumer or facility operator in the lower part connects resources and loads to the electrical grid. To be able to control these loads and resources (the upper part), they need to be known at a control center (at an operator (DSO, VPP, etc.) or market side or other third party side acting as part of the electricity grid). As depicted, the communication controller is likely to be operated behind Firewall and NAT. This leads to the problem that the address resolution of the target device to be controlled may not always be

possible. Also, inbound connections may not be possible, as they are likely to be blocked by the firewall.

The chosen communication architecture to address these problems bases on XMPP. XMPP is a standard currently being applied in chat applications where the clients also are likely to reside behind Firewalls and NAT devices. The principle of XMPP is that all connections are initiated by the communicating entities to a central server, thus avoiding any inbound connection on the entity side.

Based on Figure C.1, there are IEC 61850 client and server components running on XMPP clients, while the XMPP server is operated stand alone without a necessary combination with an (own) IEC 61850 component. Thus, it is mainly used to facilitate the IEC 61850 communication connection.

C.2 Assumptions and boundary conditions

Based on the setup depicted in Figure C.1 the following assumptions are made:

- A 1. The DER resource (IEC 61850 server on XMPP client from communication perspective) belongs to a DER owner. The DER owner typically has made some financial investment to own the DER.
- A 2. DER control (IEC 61850 client/server on XMPP client from communication perspective) belongs to DER owner, the DSO/TSO or another 3rd party grid service operator. The relation to control the DER is likely to be made via a contract and / or service level agreement (SLA).
- A 3. The contract between the DER resource owner and 3rd party grid service operator or DSO/TSO builds a trust relation regarding the information exchange between these entities.
- A 4. XMPP (infrastructure) server may belong to the DSO/TSO or another 3rd party grid service provider. It may also be provided by a non-grid related service provider like a telecom operator. Note that the XMPP server may also support a multi user chat room (MUC) capability. This may be an integrated part of the XMPP server or may be stand alone.
- A 5. XMPP server operator is trusted regarding resource discovery and message transfer.
- A 6. XMPP server operator is not trusted to act upon the message content (read, write) between the IEC 61850 client and IEC 61850 server.

As shown in Figure C.1 there are different communication relations. These communication relations can be roughly distinguished into pure transport layer connections, which are marked as hop-to-hop connections and application layer connections, which are marked as end-to-end. Note that depending on the application layer interaction, the connection may be either unicast (between two peers) or multicast (within a group). The latter relates to services provided via the MUC involving one sender and multiple receivers.

C.3 Derivation of security requirements

General security goals in power systems, which typically comprise availability, data integrity, authentication, and confidentiality, are already described within IEC 62351-1.

Security requirements stem on one hand from customer requirements, standard compliance, and regulation, on the other hand from the system design. High level security requirements can be derived from the use cases, the assumptions, and the communication relations in the systems.

Note that the identification of security requirements is purely targeting the communication security of DER related data. It does not target the security of the data when stored or system aspects like resilience.

Based on the assumptions and boundary conditions stated in Clause C.2 there is a distinction between transport layer (end-to-middle) and application layer (end-to-end) security means to address the different trust relations.

The following bullet list provides a first set of security requirements for the different communication connections depicted in Figure C.1.

- R 1. End-to-middle source authentication to ensure that the communication peers are properly identified and authenticated. End-to-middle authentication is required between XMPP client and XMPP server or between XMPP servers. Note that here it may target mainly component authentication.
- R 2. End-to-end source authentication to ensure that the communication peers are properly identified and authenticated. End-to-end authentication is required between IEC 61850 client and server instances. This authentication goes across the XMPP server ("application layer") and may be bound to a dedicated instance running on the IEC 61850 host.
Note, it is open for discussion, if there are distinct credentials to be used to address R 1 and R 2 or if this is addressed through authorization.
- R 3. End-to-middle integrity protection to ensure that data in transit has not been tampered with (unauthorized modification) between the XMPP client and XMPP server. Note, depending on the final security architecture, this requirement may be covered by security means addressing the following requirement R 4.
- R 4. End-to-end integrity protection to ensure that data in transit has not been tampered with (unauthorized modification) between the IEC 61850 client and server instances. Based on the different communication relations, the integrity protection needs to support
 - a) unicast: peer-to-peer related communication;
 - b) multicast: group based communication (via the MUC).
- R 5. End-to-middle confidentiality protection to ensure that data in transit has not been accessed (read) in an unauthorized way between the XMPP client and XMPP server. Note, depending on the final security architecture, this requirement may be covered by security means addressing the following requirement R 6.
- R 6. End-to-end confidentiality protection to ensure that data in transit has not been accessed (read) in an unauthorized way between the IEC 61850 client and server instances. Based on the different communication relations, the confidentiality protection needs to support:
 - a) unicast: peer-to-peer related communication,
 - b) multicast: group based communication (via the MUC).

Any of the requirements stated above requires an appropriate negotiation of key material between the information processing instances. Note that this may lead to additional components in the architecture realizing the key distribution.

C.4 Mapping of security options to XMPP based integration of DER

For unicast communication, the security requirements can be addressed by the security means described below (multicast communication is actually not in the scope of the present document):

- Mutual authentication, session integrity and confidentiality of XMPP client – server or server – server communication (hop-to-hop security from IEC 61850 point of view) to address R 1, R 3, and R 5.

- Utilize TLS as specified in RFC 6120 (XMPP Core)
- Utilize TLS settings according to IEC 62351-6
- Credentials used are the X.509 certificates of the peers
- Verification of a XMPP client or a XMPP server certificates requires root certificate of issuing CA to be available at another XMPP server (most likely the issuer has a relation to the DSO or another 3rd party grid service provider) to be able to validate received certificates
- May additionally require access lists of allowed XMPP clients (DER resources) for the XMPP server upfront provided by the DSO (as blacklist or white list)
- Integrity protection and/or confidentiality protection can be negotiated
- End-to-end authentication, integrity, and confidentiality can be achieved by applying the end-to-end security concept described in IEC 62351-4 to address R 2, R 4a, and R 6a.

C.5 Sequence diagrams

C.5.1 General

The following sequence diagrams illustrate the usage of the end-to-end security protocol in combination with the IEC 61850 actors. The IEC 6150 applications (client or server) are being exposed as different actors to the end-to-end security layer, represented by the actor “E2E Sec Protocol” in sequence diagrams of next subclauses. It is however for the understanding of the layered approach. The two actors can be implemented within a single task context. The parameters used in the functions are for exposing which information is available and can be used/checked at the different levels of the stack.

The E2E security protocol layer needs a configuration phase in each implementation to be conformant to the different security policies specified in the IEC 62351-4.

The associate handshake between the IEC 6150 associate caller and IEC 61850 associate called will be mapped to the E2E security protocol message “assocRequest” and “assocResult”. This handshake will be decisive in the determination of the negotiated confidentiality parameters (none with integrity check, with integrity-check + encryption) that rule the associations.

C.5.2 XMPP and Stream opening

Figure C.2 illustrates the stream establishment from the IEC 61850 Server to the XMPP Server over the XMPP Client component. The IEC 61850 server is configured with a JID – JidDer and owns a certificate.

The OpenStream service between the XMPP Client and XMPP Server is an abstraction of the XML stream creation and negotiation as specified in the XMPP Core – RFC 6120. The stream negotiation shall include the <stream:features/><starttls/> negotiation as soon as the transport used between both XMPP Client and XMPP Server is not secured via a TLS tunnel communication.

The XMPP Server shall authenticate the XMPP Client as being entitled to use the JID (“JidDer” in the figures) it is using. The SASL (Simple Authentication and Security Layer) negotiation can achieve the requirement:

- Either via the SASL EXTERNAL mandating to have the used JID as a certificate entry in the IEC 61850 server,
- Or using the SASL PLAIN mandating to have a unique password associated to each of the JID involved in the system.

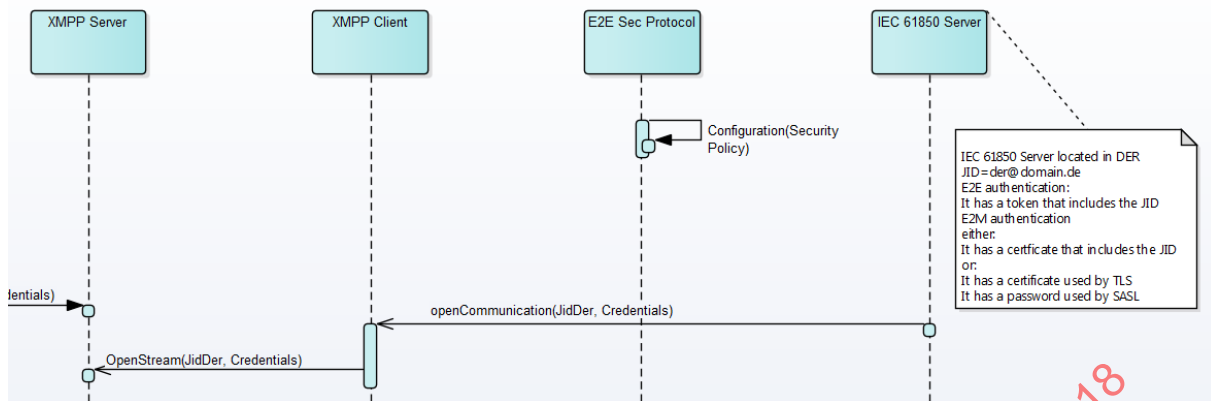


Figure C.2 – XMPP Stream establishment – IEC 61850 Server to the XMPP Server

After the XMPP Stream has been established, stanzas to “JidDer” can be forwarded to the XMPP Client “JidDer”.

The same diagram applies to the stream establishment between the IEC 61850 Client and the XMPP server (see Figure C.3). After the XMPP Stream has been established, stanzas to “JidClient” can be forwarded to the XMPP Client “JidClient”.

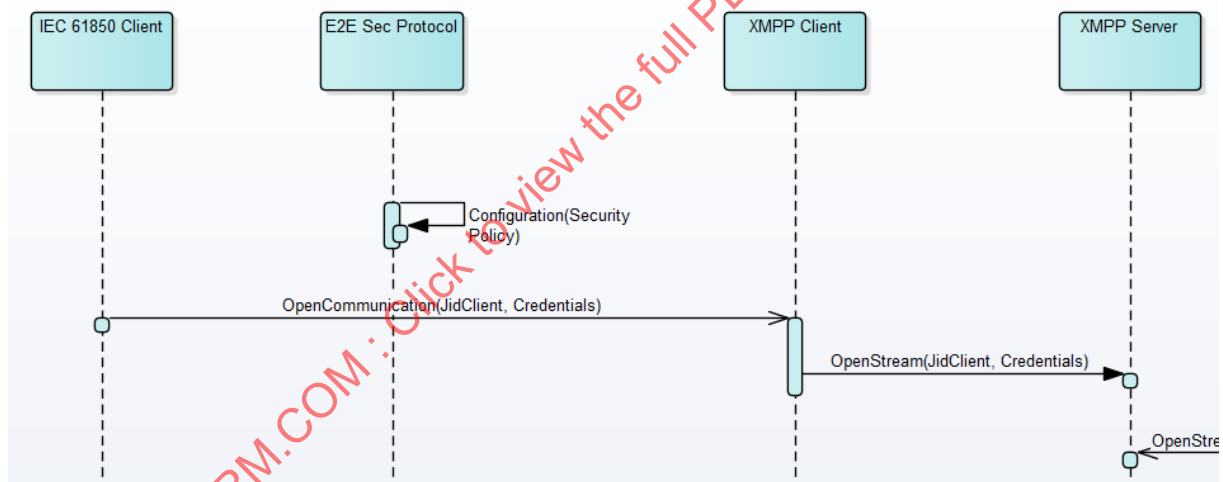


Figure C.3 – XMPP Stream establishment – IEC 61850 Client to the XMPP Server

C.5.3 Stream establishment, ROSTER and presence

According to RFC 6121, after negotiation of the XML Stream, the typical flow for an XMPP session that includes usage of presence consists in:

- Retrieving / updating one’s roster,
- Send an initial presence.

The behaviour is illustrated in Figure C.4.

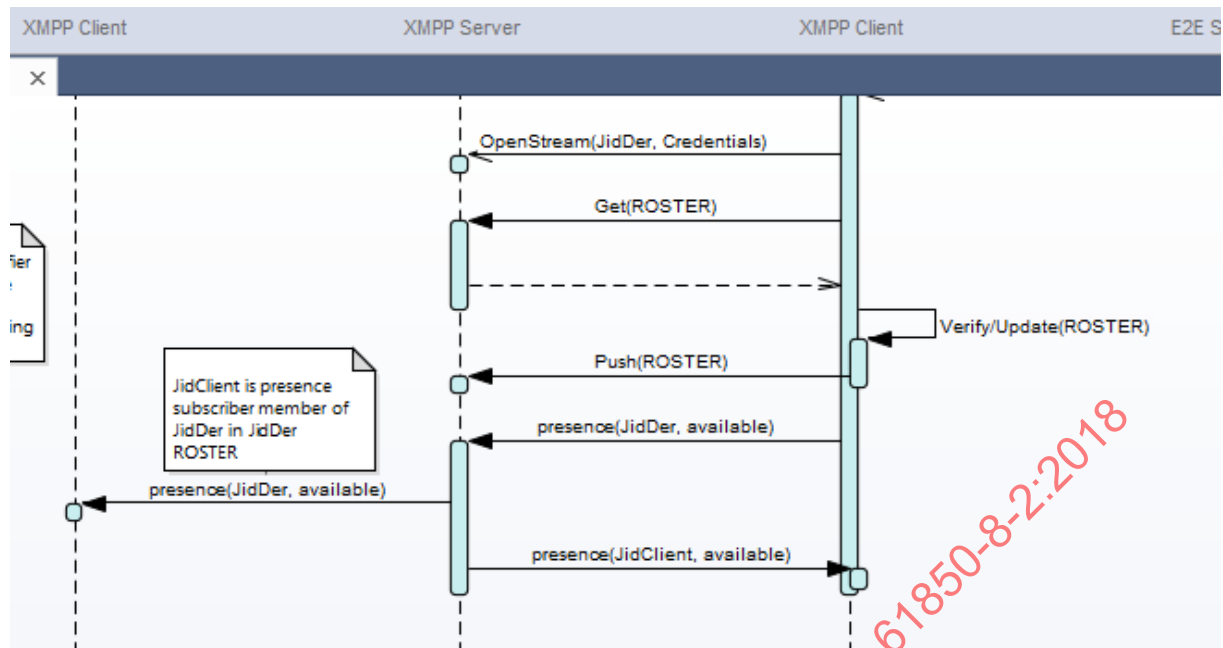


Figure C.4 – Stream establishment, roster and presence

C.5.4 Communication outage

There are two communication outage detections when an IEC 61850 association is established:

- A loss of link on the connection to the XMPP server over which the IEC 61850 association is being transported – Figure C.5,
- An indirect outage being advertised by the presence “unavailable” by the XMPP server – Figure C.6.

Communication outage leads to signaling an Abort to the IEC 61850 server and client.

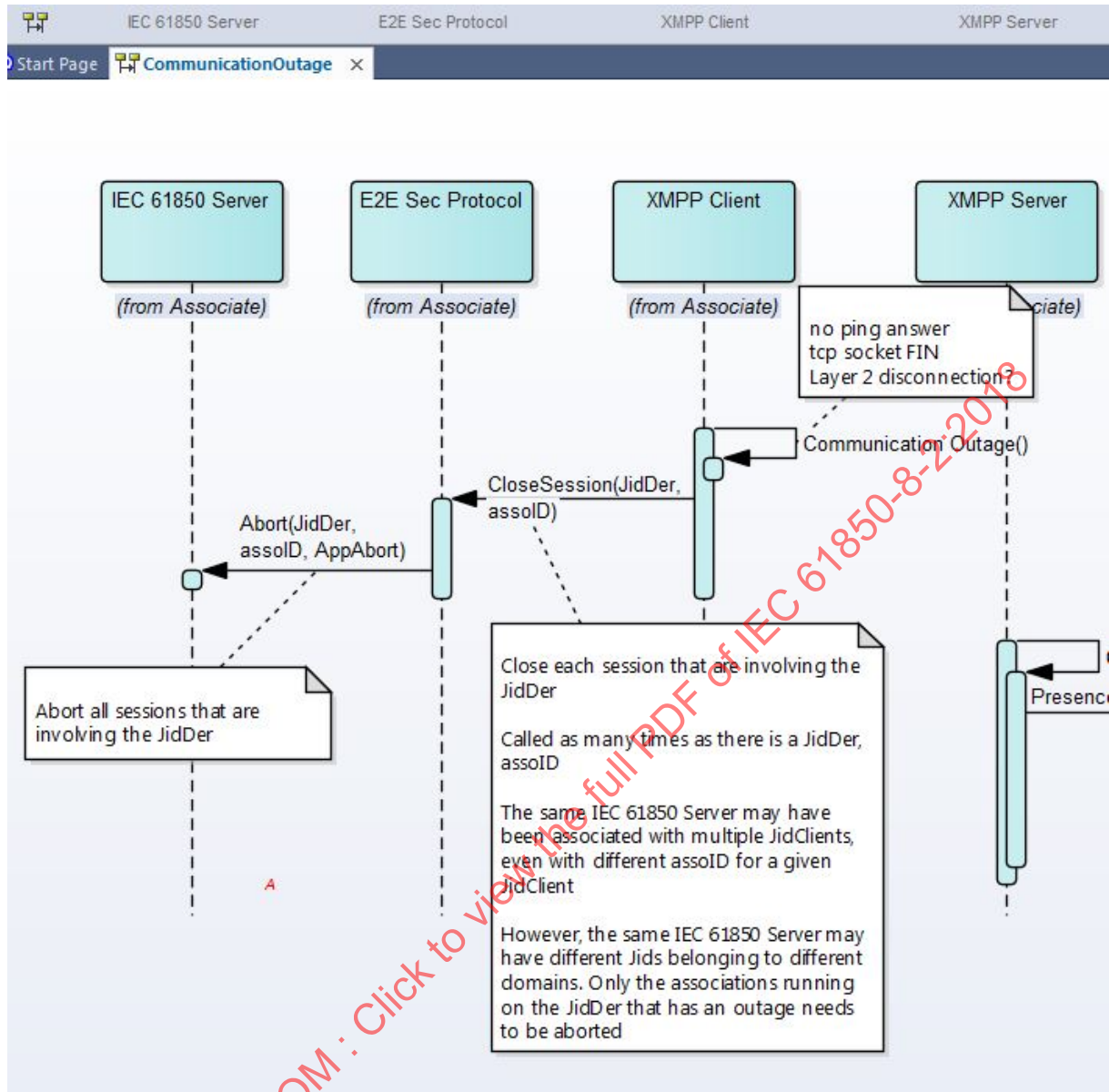


Figure C.5 – Communication outage – Loss of link

In the loss of link scenario, the XMPP client loops on all association contexts involving the own Jid – for each {JidDer, assoID}, to abort each one.

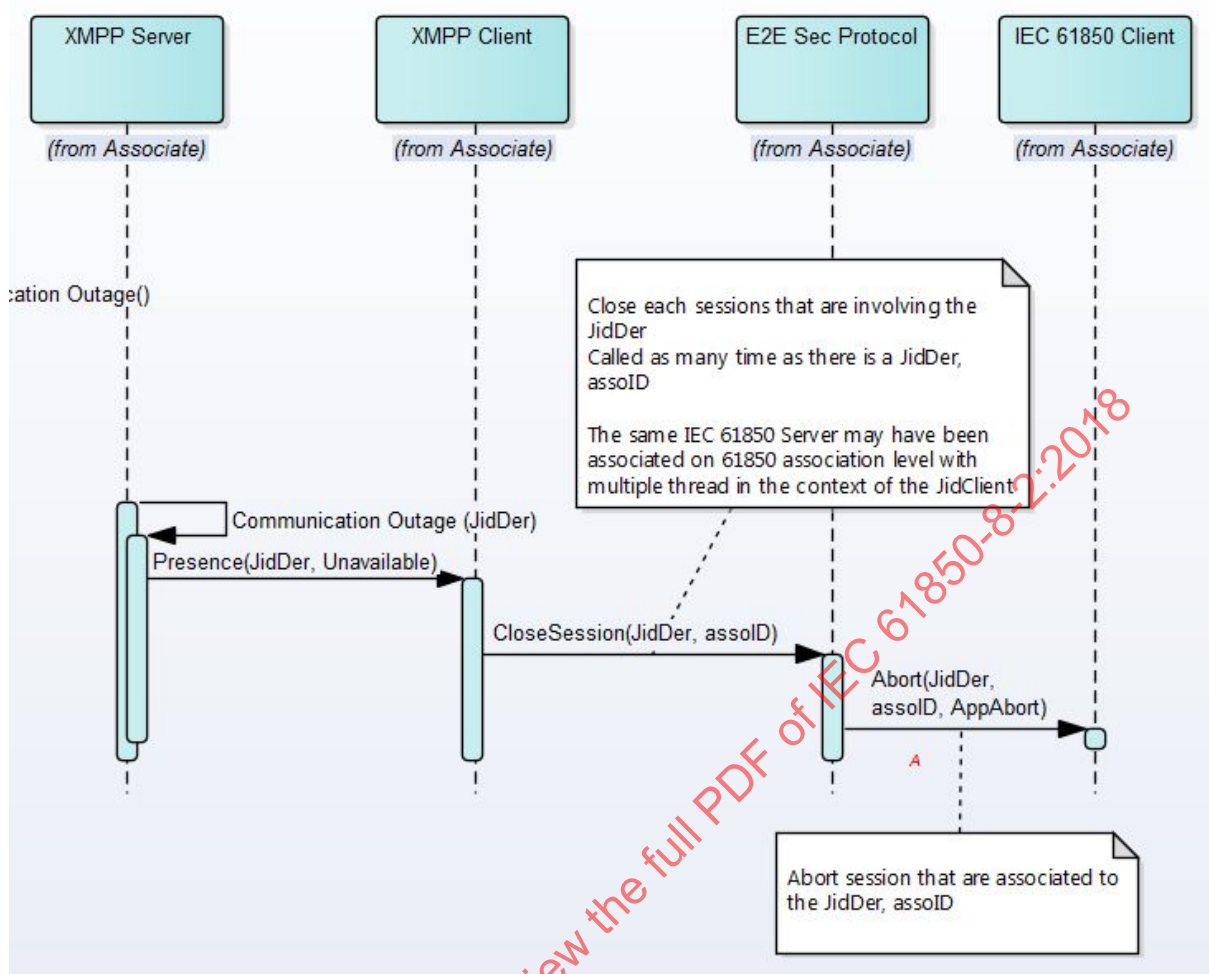


Figure C.6 – Communication outage – Presence unavailable

In the presence unavailable advertisement scenario, the XMPP client loops on all association contexts involving the unavailable (remote) Jid – for each {JidDer, assoID}, to abort each one.

C.5.5 Request Response (Clear Transfer)

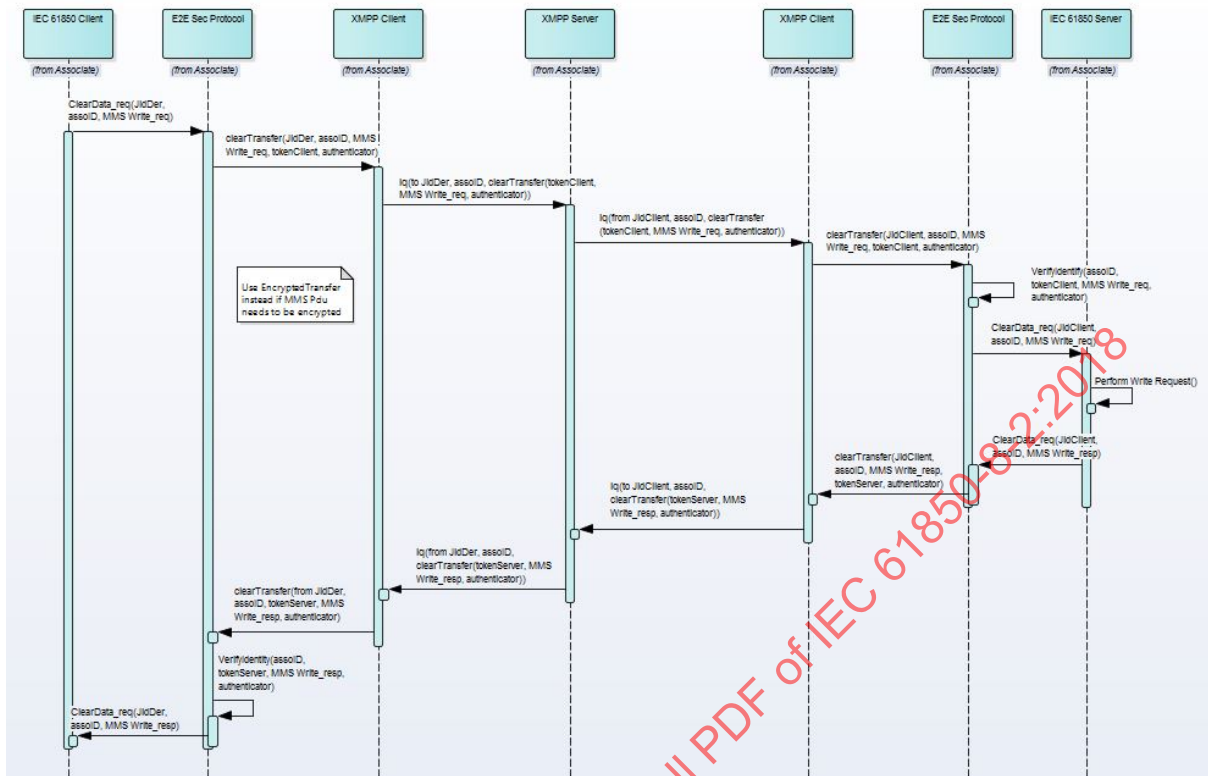


Figure C.7 – Request response

Figure C.7 illustrates the steps of a Request – Response handshake, being first mapped to an Write_req and packed by the E2E Sec Protocol to a clearTransfer (in case the the encryption has been negotiated, the E2E Sec Protocol will pack the encrypted Write_req in an encrTransfer). The clearTransfer includes:

- The unique identifier of the association assoID,
- The security token of the IEC 61850 client,
- The request authenticator (integrity of payload),
- The payload data: the Write_req in Clear form.

The XMPP Client packs the clearTransfer within a stanza iq and sends it over the XMPP Stream to the JidDer.

The XMPP Server forwards the iq to the JidDer from the JidClient. The XMPP Client of the IEC 61850 Server unpacks the iq Stanza and forwards the clearTransfer to the E2E Sec Protocol layer.

In case the association context is not known, the E2E Sec Protocol responds with an IQ result indicating failure as specified in 8.3.3.4.3.

The E2E Sec Protocol verifies the authenticator and the token provided by the client. In case the integrity verification fails, the E2E Sec Protocol raises a ClearTransferError to the IEC 61850 server, and responds to the compromised request with a dtSecAbort. A dtSecAbort is issued when a problem is encountered during the data transfer phase. It leads to the abortion / termination of the association as illustrated in Figure C.8.

The IEC 61850 Server verifies if the access rights for the requested object are granted for the role used by the IEC 61850 Client advertised during the Initiate handshake. In case no privilege are granted, the Write_req is responded with an Write_resp+ indicating failure. In case no error occurs, the IEC 61850 Server responds with an Write_resp+ indicating success, packed by the E2E Sec Protocol to an clearTransfer. The clearTransfer includes:

- The unique identifier of the association assoid,
- The security token of the IEC 61850 server,
- The request authenticator (integrity of payload),
- The payload data: the Write_resp in Clear form.

The XMPP Client packs the clearTransfer within a stanza iq type “result” and sends it over the XMPP Stream to the JidClient.

The XMPP Server forwards the iq to the JidClient for the JIDServer. The XMPP Client unpacks the iq type “result” to a clearTransfer before forwarding it to the E2E Sec Protocol. The E2E Sec Protocol verifies the integrity of the response before forwarding the Write_resp to the IEC 61850 Client.

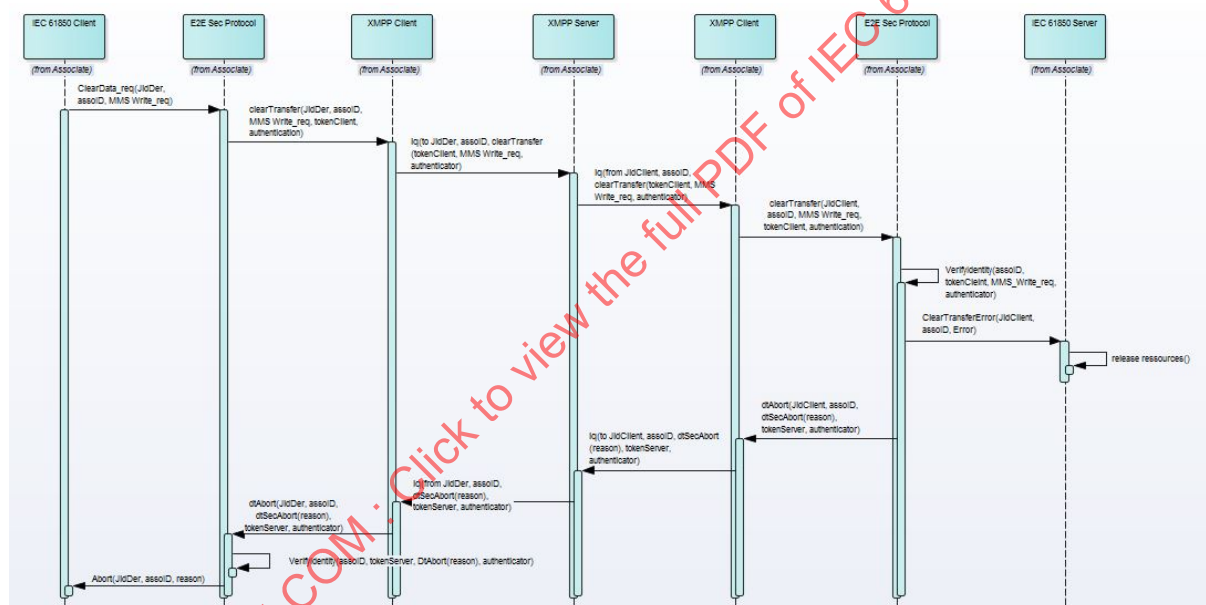


Figure C.8 – Request – Abort

Annex D (normative)

Mapping of services and errors over XMPP stanzas

Table D.1 defines the general structure of the XMPP stanzas exchanged for mapping the ACSI services. Columns 1 to 3 describe the possible structures for the payloads, columns 4 and 5 show how they shall be transported over XMPP stanzas.

NOTE 1 The column 1 indicates in which kind of end-to-end security elements the envelopes of column 2 are embedded. The exact structure of these security elements is defined in IEC 62351-4. Note that these security elements are themselves embedded into the association context part also defined in IEC 62351-4 (see global structure of the XML payload in Figure 2). Note finally that an element <AppData/> is inserted at the interface between the security elements and the applicative payload parts (formally defined in G.2).

NOTE 2 The detailed mapping starting at Clause 9 explains which envelop and service element shall be used for mapping each ACSI service.

NOTE 3 For information, the service elements mentioned here correspond to the MMS services used in IEC 61850-8-1.

Table D.1 – ACSI services mapping over XMPP stanzas

IEC 62351-4 context	Envelop	Service element	Stanza	Type
ClearTransfer or EncrTransfer	confirmed-RequestPDU	getNameList	iq	get
		Read	iq	get
		Write	iq	set
		getVariableAccessAttributes	iq	get
		defineNamedVariableList	iq	set
		getNamedVariableListAttributes	iq	get
		deleteNamedVariableList	iq	set
		getNamedTypeAttributes	iq	get
		readJournal	iq	get
		fileOpen	iq	set
		fileRead	iq	get
		fileClose	iq	set
		fileDelete	iq	set
		fileDirectory	iq	get
		obtainFile	iq	get
		extendedGetNameList	iq	get
	confirmed-ResponsePDU	getNameList	iq	result
		Read	iq	result
		Write	iq	result
		getVariableAccessAttributes	iq	result
		defineNamedVariableList	iq	result
		getNamedVariableListAttributes	iq	result
		deleteNamedVariableList	iq	result
		getNamedTypeAttributes	iq	result
		readJournal	iq	result
		fileOpen	iq	result
		fileRead	iq	result
		fileClose	iq	result

IEC 62351-4 context	Envelop	Service element	Stanza	Type
		fileDelete	iq	result
		fileDirectory	iq	result
		obtainFile	iq	result
		extendedGetNameList	iq	result
	confirmed-ErrorPDU		iq	result
	unconfirmed-PDU	informationReport	message	normal
	rejectPDU		message	normal
	cancel-RequestPDU		iq	get
	cancel-ResponsePDU		iq	result
	cancel-ErrorPDU		iq	result
	conclude-RequestPDU		iq	set
	conclude-ResponsePDU		iq	result
	conclude-ErrorPDU		iq	result
HandshakeReq	initiate-RequestPDU		See IEC 62351-4	
HandshakeAcc	initiate-ResponsePDU			
ApplicationReject	initiate-ErrorPDU			

Table D.2 shows how the different kinds of errors are mapped as XMPP stanzas (ACSI service errors and protocol errors). The left side of the table gives a description of the errors and indicates in which layer of the communication stack the errors may be raised.

Table D.2 – Error mapping over XMPP stanzas

Layer where the error is raised			XMPP mapping				
IEC 61850 client or server	E2E security protocol and Association Context Management	XMPP Client	XMPP concept	Type	First child element	Sub-element	Sub-element namespace
		Unrecoverable error during stream setup or stanza transmission	stream: error / iq / message	error	NA/error/error	Defined in RFC 6120	urn:ietf:params:xml:ns:xmpp-streams
		Recoverable error during stanza transmission	iq / message	error	error/error	Defined in RFC 6120	urn:ietf:params:xml:ns:xmpp-stanzas
	Reception of Unknown Association ID		See IEC 62351-4				
	Security error at reception of stanzas holding various kinds of envelop (Initiate-RequestPDU, Initiate-ResponsePDU, confirmed-RequestPDU, confirmed-ResponsePDU, Confirmed-ErrorPDU, Unconfirmed-PDU, ...)						
initiate-ErrorPDU							
Mapping and Reject errors ^a			message	normal	Mapped as a RejectPDU envelop, see Table D.1		
Error raised during established IEC 61850 association ^b			iq / message	result / normal	May be mapped on envelops confirmed-ResponsePDU, Confirmed-ErrorPDU, rejectPDU, cancel-ErrorPDU, conclude-Error-PDU, see Table D.1		

^a See 8.3.3.4.5 and 8.3.3.4.6

^b Errors specified within the present SCSCM as mappings of IEC 61850-7-2 service errors or SCSCM specific errors.

Annex E

(informative)

Intentional deviations from IEC 61850-8-1 SCSM

IEC 61850-8-1 and the present SCSM are sharing mostly the same abstract definition for the structure of the messages used for mapping the ACSI services as defined by IEC 61850-7-2. This abstract definition is based on ISO 9506 (MMS) and is using the abstract syntax ASN.1. Basically, the main difference between the two SCSMs is that IEC 61850-8-1 is using real messages with a binary encoding while the present SCSM is using messages with an XML encoding.

However there is in fact one difference to be considered between the two SCSMs IEC 61850-8-2 and IEC 61850-8-1, regarding the mapping of the ACSI services GetLogicalDeviceDirectory, GetLogicalNodeDirectory and GetDataDirectory. The way IEC 61850-8-1 is mapping these services onto MMS produces large Response+ messages where most of the information is not required and must be filtered at client side. Therefore IEC 61850-8-2 proposes an extended mode for the considered services in addition to the default mode which is to use a straightforward XML encoding of the abstract MMS messages used by IEC 61850-8-1.

When using the optional extended mode the size of the Response+ messages is reduced by including only the mandatory information. The choice between the default and extended behaviour is made at association time. When the extended behaviour is selected, the concerned ACSI services are using the choice extendedGetNameList instead of their default choice (getNameList or getVariableAccessAttributes) for creating their XML payloads. For more details see Clause 8.5 and the clause related to each considered ACSI services.

NOTE 1 A gateway translating the messages coming from an IEC 61850-8-1 endpoint into messages that an IEC 61850-8-2 endpoint can understand and vice-versa, will have to negotiate the default mode with the IEC 61850-8-2 endpoint. Otherwise no straightforward translation will be possible for each message of the considered services.

NOTE 2 IEC 61850-8-1 and the present SCSM are also sharing the same abstract definition for the part of the payload related to the end-to-end security mechanism defined by IEC 62351-4. However a gateway between IEC 61850-8-1 and IEC 61850-8-2 endpoints is establishing an end-to-end security session separately for each of the two endpoints and the security elements within the payloads are not translated directly between the IEC 61850-8-1 and IEC 61850-8-2 sessions.

SCL conformance

Table F.1 – SCL conformance degrees

	SCL conformance	Client-CR			Server-CR		
		Base	F/S	Value/range	Base	F/S	Value/range
SCL.1	SCL file for implementation available (offline)				m	m	
SCL.2	SCL file available from implementation online	o	o		o	o	
SCL.3	SCL file contains a communication section according to clause 25	o	c		o	m	

c Shall be 'm' if the client can be called

Implementations claiming conformance to SCL 2 may support the ACSI services defined in Table F.2.

Table F.2 – Supported ACSI services for SCL.2

	SCL Conformance	Client-CR			Server-CR		
		Base	F/S	Value/range	Base	F/S	Value/range
ACSI services							
	GetFileAttributeValues	o	o		o	c1	
	GetFile	o	c1		o	c1	
	GetDataValues	o	c1		o	c1	
c1 Shall be 'm' if support for SCL.2 is declared.							

Annex G (normative)

XML schema definitions for the XML payload

G.1 General

The XML payloads transported over XMPP consist of several parts which are not all defined within the present document. The first levels of child elements within the XMPP stanzas are in fact defined in IEC 62351-4 and are mainly related to the management of the end-to-end security. The present annex describes only the applicative part of the payload (see G.3). In order to ease the maintenance and to limit the dependancies, the XSD described in IEC 62351-4 doesn't import directly the XSD presented in G.3, but it rather includes the one presented in G.2 which acts as a Virtual API importing finally the definitions of the applicative payload.

G.2 XML schema of the Virtual API for IEC 61850-8-2

<CODE BEGINS>

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:pp="http://www.iec.ch/61850/2018/SCSM_8_2">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
      COPYRIGHT (c) IEC, 2016. This version of the Virtual API XSD is part of IEC 61850-8-2:2018 ed1.0; see the IEC 61850-8-2:2018 ed1.0 for full legal notices. In case of any differences between the here-below code and the IEC published content, the here-below definition supersedes the IEC publication; it may contain updates. See history files. The whole document has to be taken into account to have a full description of this code component.
      See www.iec.ch/CCv1 for copyright details
    </xsd:documentation>
    <xsd:documentation xml:lang="en">
      schema of IEC 61850-8-2:2018 ed1.0 Version "2018", Revision "A", Release "1", "Draft"
    </xsd:documentation>
  </xsd:annotation>
  <xsd:import namespace="http://www.iec.ch/61850/2018/SCSM_8_2" schemaLocation="IEC61850-8-2.xsd"/>
  <xsd:element name="ApplData" type="pp:MMSpdu"/>
</xsd:schema>
```

<CODE ENDS>

G.3 XML schema of the applicative payload for IEC 61850-8-2

This annex provides the formal W3C XSD specification for the applicative payload defined in this document. This schema shall be combined with other W3C XSDs as specified in G.1.

NOTE 1 A large part of this XSD can be derived automatically from the ASN.1 productions defined in MMS.

NOTE 2 The OSS Novalka ASN.1 to XSD Translator tool was used in the creation of this XSD.

<CODE BEGINS>

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Generated by: OSS ASN.1 to XSD Translator Version 4.1
  Generated for: Bernard Bony, Schneider Electric – Standards Editor, License 16225E,
  only for project "Generation of XML schema to be included in the IEC 61850-2 standard document".
  Created: Fri Apr 21 17:37:27 2017
  Abstract syntax: IEC61850_8_2
  Options and file names specified:
  -exer -useExtensibility IEC61850-8-2.asn
-->
<!--MMS Abstract Syntax -->
<!-- Clause 7 – MMS PDU of ISO/IEC 9506-2:1990 -->
```

```

<!--Module-MMS DEFINITIONS::= BEGIN -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://www.iec.ch/61850/2018/SCSM_8_2"
xmlns:oss="http://www.oss.com/XSD" targetNamespace="http://www.iec.ch/61850/2018/SCSM_8_2" elementFormDefault="qualified">
  <!-- BEGIN not generated part -->
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
      COPYRIGHT (c) IEC, 2016. This version of the Applicative Payload XSD is part of IEC 61850-8-2:2018 ed1.0; see the
      IEC 61850-8-2:2018 ed1.0 for full legal notices. In case of any differences between the here-below code and the IEC published content, the
      here-below definition supersedes the IEC publication; it may contain updates. See history files. The whole document has to be taken into
      account to have a full description of this code component.
      See www.iec.ch/CCv1 for copyright details
    </xsd:documentation>
    <xsd:documentation xml:lang="en">
      schema of IEC 61850-8-2:2018 ed1.0 Version "2018", Revision "A", Release "1", "Draft"
    </xsd:documentation>
  </xsd:annotation>
  <xsd:simpleType name="Version">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="2[0-2][0-9]{2}" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="Revision">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[A-Z]" />
    </xsd:restriction>
  </xsd:simpleType>
  <!-- END not generated part -->
  <xsd:element name="MMSpdu" type="MMSpdu"/>
  <!-- History:
- Initialized with MMS part of file IEC62351-4-total103116.asn used for generating the XSD for the CDV
- Add the EXER encoding instructions at the end
- Add again the DEFAULT values for the elements to be encoded as XML attributes
- Replace named IMPLICIT INTEGER by CHOICE of IMPLICIT NULL except for ScopeOfDelete
- For ParameterSupportOptions and ServiceSupportOptions, remove SEQUENCE of implicit BOOLEAN and come back to BIT STRING
- Remove IMPLICIT before CHOICE in CDV definition of GetNameList-Request.
-->
  <xsd:complexType name="MMSpdu">
    <xsd:choice>
      <xsd:element name="confirmed-RequestPDU" type="Confirmed-RequestPDU"/>
      <xsd:element name="confirmed-ResponsePDU" type="Confirmed-ResponsePDU"/>
      <xsd:element name="confirmed-ErrorPDU" type="Confirmed-ErrorPDU"/>
      <xsd:element name="unconfirmed-PDU" type="Unconfirmed-PDU"/>
      <xsd:element name="rejectPDU" type="RejectPDU"/>
      <xsd:element name="cancel-RequestPDU" type="Cancel-RequestPDU"/>
      <xsd:element name="cancel-ResponsePDU" type="Cancel-ResponsePDU"/>
      <xsd:element name="cancel-ErrorPDU" type="Cancel-ErrorPDU"/>
      <xsd:element name="initiate-RequestPDU" type="Initiate-RequestPDU"/>
      <xsd:element name="initiate-ResponsePDU" type="Initiate-ResponsePDU"/>
      <xsd:element name="initiate-ErrorPDU" type="Initiate-ErrorPDU"/>
      <xsd:element name="conclude-RequestPDU" type="Conclude-RequestPDU"/>
      <xsd:element name="conclude-ResponsePDU" type="Conclude-ResponsePDU"/>
      <xsd:element name="conclude-ErrorPDU" type="Conclude-ErrorPDU"/>
    </xsd:choice>
  </xsd:complexType>
  <xsd:complexType name="Conclude-ErrorPDU">
    <xsd:sequence>
      <xsd:element name="errorClass">
        <xsd:complexType>
          <xsd:choice>
            <xsd:element name="vmd-state">
              <xsd:complexType>
                <xsd:choice>
                  <xsd:element name="other" type="NULL"/>
                  <xsd:element name="vmd-state-conflict" type="NULL"/>
                  <xsd:element name="vmd-operational-problem" type="NULL"/>
                  <xsd:element name="domain-transfer-problem" type="NULL"/>
                  <xsd:element name="state-machine-id-invalid" type="NULL"/>
                </xsd:choice>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="application-reference">
              <xsd:complexType>
                <xsd:choice>
                  <xsd:element name="other" type="NULL"/>
                  <xsd:element name="application-unreachable" type="NULL"/>
                  <xsd:element name="connection-lost" type="NULL"/>
                  <xsd:element name="application-reference-invalid" type="NULL"/>
                </xsd:choice>
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>

```

```

        <xsd:element name="context-unsupported" type="NULL"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="definition">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="other" type="NULL"/>
        <xsd:element name="object-undefined" type="NULL"/>
        <xsd:element name="invalid-address" type="NULL"/>
        <xsd:element name="type-unsupported" type="NULL"/>
        <xsd:element name="type-inconsistent" type="NULL"/>
        <xsd:element name="object-exists" type="NULL"/>
        <xsd:element name="object-attribute-inconsistent" type="NULL"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="resource">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="other" type="NULL"/>
        <xsd:element name="memory-unavailable" type="NULL"/>
        <xsd:element name="processor-resource-unavailable" type="NULL"/>
        <xsd:element name="mass-storage-unavailable" type="NULL"/>
        <xsd:element name="capability-unavailable" type="NULL"/>
        <xsd:element name="capability-unknown" type="NULL"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="service">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="other" type="NULL"/>
        <xsd:element name="primitives-out-of-sequence" type="NULL"/>
        <xsd:element name="object-state-conflict" type="NULL"/>
        <!--defined by NIST/ISP agreements -->
        <xsd:element name="pdu-size" type="NULL"/>
        <xsd:element name="continuation-invalid" type="NULL"/>
        <xsd:element name="object-constraint-conflict" type="NULL"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="service-preempt">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="other" type="NULL"/>
        <xsd:element name="timeout" type="NULL"/>
        <xsd:element name="deadlock" type="NULL"/>
        <xsd:element name="cancel" type="NULL"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="time-resolution">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="other" type="NULL"/>
        <xsd:element name="unsupportable-time-resolution" type="NULL"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="access">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="other" type="NULL"/>
        <xsd:element name="object-access-unsupported" type="NULL"/>
        <xsd:element name="object-non-existent" type="NULL"/>
        <xsd:element name="object-access-denied" type="NULL"/>
        <xsd:element name="object-invalidated" type="NULL"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="initiate">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="other" type="NULL"/>
        <!--defined by NIST/ISP agreements -->
        <xsd:element name="version-incompatible" type="NULL"/>

```

```

<!--defined by NIST/ISP agreements -->
<xsd:element name="max-segment-insufficient" type="NULL"/>
<xsd:element name="max-services-outstanding-calling-insufficient" type="NULL"/>
<xsd:element name="max-services-outstanding-called-insufficient" type="NULL"/>
<xsd:element name="service-CBB-insufficient" type="NULL"/>
<xsd:element name="parameter-CBB-insufficient" type="NULL"/>
<xsd:element name="nesting-level-insufficient" type="NULL"/>
</xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="conclude">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="further-communication-required" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="cancel">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="invoke-id-unknown" type="NULL"/>
      <xsd:element name="cancel-not-possible" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="file">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="filename-ambiguous" type="NULL"/>
      <xsd:element name="file-busy" type="NULL"/>
      <xsd:element name="filename-syntax-error" type="NULL"/>
      <xsd:element name="content-type-invalid" type="NULL"/>
      <xsd:element name="position-invalid" type="NULL"/>
      <xsd:element name="file-access-denied" type="NULL"/>
      <xsd:element name="file-non-existent" type="NULL"/>
      <xsd:element name="duplicate-filename" type="NULL"/>
      <xsd:element name="insufficient-space-in-filestore" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<!--not used by IEC 61850 -->
<xsd:element name="others" type="INTEGER"/>
</xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="additionalCode" type="INTEGER" minOccurs="0"/>
<xsd:element name="additionalDescription" type="VisibleString" minOccurs="0"/>
<xsd:element name="serviceSpecificInformation" minOccurs="0">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="obtainFile" type="ObtainFile-Error"/>
      <!-- start [1] IMPLICIT Start-Error, not used by IEC 61850 -->
      <!-- stop [2] IMPLICIT Stop-Error, not used by IEC 61850 -->
      <!-- resume [3] IMPLICIT Resume-Error, not used by IEC 61850 -->
      <!-- reset [4] IMPLICIT Reset-Error, not used by IEC 61850 -->
      <!-- deleteVariableAccess [5] IMPLICIT DeleteVariableAccess-Error, not used by IEC 61850 -->
      <!-- deleteNamedType [7] IMPLICIT DeleteNamedType-Error, not used by IEC 61850 -->
      <!-- defineEventEnrollment-Error [8] DefineEventEnrollment-Error, not used by IEC 61850 -->
      <!-- fileRename [9] IMPLICIT FileRename-Error -->
      <!-- additionalService [10] AdditionalServiceError not used by IEC 61850 -->
      <xsd:element name="deleteNamedVariableList" type="DeleteNamedVariableList-Error"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<!-- number Deleted -->
<xsd:simpleType name="DeleteNamedVariableList-Error">
  <xsd:restriction base="Unsigned32"/>
</xsd:simpleType>
<!--Unsigned16:= INTEGER not used for IEC 61850 ASN.1 -->
<xsd:simpleType name="Unsigned32">
  <xsd:restriction base="INTEGER"/>
</xsd:simpleType>

```

```

<xsd:simpleType name="INTEGER">
  <xsd:restriction base="xsd:integer"/>
</xsd:simpleType>
<xsd:simpleType name="ObtainFile-Error">
  <xsd:union memberTypes="xsd:integer">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="source-file"/>
        <xsd:enumeration value="destination-file"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>
<xsd:simpleType name="VisibleString">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[~]*"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="NULL">
  <xsd:restriction base="xsd:string">
    <xsd:length value="0"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Conclude-ResponsePDU">
  <xsd:restriction base="NULL"/>
</xsd:simpleType>
<!-- Conclude from Clause 8.3 of ISO/IEC 9506-2:1990 -->
<xsd:simpleType name="Conclude-RequestPDU">
  <xsd:restriction base="NULL"/>
</xsd:simpleType>
<xsd:complexType name="Initiate-ErrorPDU">
  <xsd:sequence>
    <xsd:element name="errorClass">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="vmd-state">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="vmd-state-conflict" type="NULL"/>
                <xsd:element name="vmd-operational-problem" type="NULL"/>
                <xsd:element name="domain-transfer-problem" type="NULL"/>
                <xsd:element name="state-machine-id-invalid" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="application-reference">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="application-unreachable" type="NULL"/>
                <xsd:element name="connection-lost" type="NULL"/>
                <xsd:element name="application-reference-invalid" type="NULL"/>
                <xsd:element name="context-unsupported" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="definition">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="object-undefined" type="NULL"/>
                <xsd:element name="invalid-address" type="NULL"/>
                <xsd:element name="type-unsupported" type="NULL"/>
                <xsd:element name="type-inconsistent" type="NULL"/>
                <xsd:element name="object-exists" type="NULL"/>
                <xsd:element name="object-attribute-inconsistent" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="resource">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="memory-unavailable" type="NULL"/>
                <xsd:element name="processor-resource-unavailable" type="NULL"/>
                <xsd:element name="mass-storage-unavailable" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

```

        <xsd:element name="capability-unavailable" type="NULL"/>
        <xsd:element name="capability-unknown" type="NULL"/>
    </xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="service">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="primitives-out-of-sequence" type="NULL"/>
            <xsd:element name="object-sate-conflict" type="NULL"/>
            <!--defined by NIST/ISP agreements -->
            <xsd:element name="pdu-size" type="NULL"/>
            <xsd:element name="continuation-invalid" type="NULL"/>
            <xsd:element name="object-constraint-conflict" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="service-preempt">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="timeout" type="NULL"/>
            <xsd:element name="deadlock" type="NULL"/>
            <xsd:element name="cancel" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="time-resolution">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="unsupportable-time-resolution" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="access">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="object-access-unsupported" type="NULL"/>
            <xsd:element name="object-non-existent" type="NULL"/>
            <xsd:element name="object-access-denied" type="NULL"/>
            <xsd:element name="object-invalidated" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="initiate">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <!--defined by NIST/ISP agreements -->
            <xsd:element name="version-incompatible" type="NULL"/>
            <!--defined by NIST/ISP agreements -->
            <xsd:element name="max-segment-insufficient" type="NULL"/>
            <xsd:element name="max-services-outstanding-calling-insufficient" type="NULL"/>
            <xsd:element name="max-services-outstanding-called-insufficient" type="NULL"/>
            <xsd:element name="service-CBB-insufficient" type="NULL"/>
            <xsd:element name="parameter-CBB-insufficient" type="NULL"/>
            <xsd:element name="nesting-level-insufficient" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="conclude">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="further-communication-required" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="cancel">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="invoke-id-unknown" type="NULL"/>
            <xsd:element name="cancel-not-possible" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>

```



```

        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="file">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="other" type="NULL"/>
          <xsd:element name="filename-ambiguous" type="NULL"/>
          <xsd:element name="file-busy" type="NULL"/>
          <xsd:element name="filename-syntax-error" type="NULL"/>
          <xsd:element name="content-type-invalid" type="NULL"/>
          <xsd:element name="position-invalid" type="NULL"/>
          <xsd:element name="file-access-denied" type="NULL"/>
          <xsd:element name="file-non-existent" type="NULL"/>
          <xsd:element name="duplicate-filename" type="NULL"/>
          <xsd:element name="insufficient-space-in-filestore" type="NULL"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
    <!-- not used by IEC 61850 -->
    <xsd:element name="others" type="INTEGER"/>
  </xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="additionalCode" type="INTEGER" minOccurs="0"/>
<xsd:element name="additionalDescription" type="VisibleString" minOccurs="0"/>
<xsd:element name="serviceSpecificInformation" minOccurs="0">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="obtainFile" type="ObtainFile-Error"/>
      <!-- start [1] IMPLICIT Start-Error, not used by IEC 61850 -->
      <!-- stop [2] IMPLICIT Stop-Error, not used by IEC 61850 -->
      <!-- resume [3] IMPLICIT Resume-Error, not used by IEC 61850 -->
      <!-- reset [4] IMPLICIT Reset-Error, not used by IEC 61850 -->
      <!-- deleteVariableAccess [5] IMPLICIT DeleteVariableAccess-Error, not used by IEC 61850 -->
      <!-- deleteNamedType [7] IMPLICIT DeleteNamedType-Error, not used by IEC 61850 -->
      <!-- defineEventEnrollment-Error [8] DefineEventEnrollment-Error, not used by IEC 61850 -->
      <!-- fileRename [9] IMPLICIT FileRename-Error -->
      <!-- additionalService [10] AdditionalServiceError not used by IEC 61850 -->
      <xsd:element name="deleteNamedVariableList" type="DeleteNamedVariableList-Error"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
</xsd:sequence>
<!-- BEGIN not generated part -->
<xsd:attribute name="version" type="Version" default="2018"/>
<xsd:attribute name="revision" type="Revision" default="A"/>
<xsd:attribute name="release" type="xsd:unsignedShort" default="1"/>
<!-- END not generated part -->
</xsd:complexType>
<xsd:complexType name="Initiate-ResponsePDU">
  <xsd:sequence>
    <!-- Optional for MMS, Mandatory for IEC 61850 -->
    <xsd:element name="localDetailCalled" type="Integer32"/>
    <xsd:element name="negotiatedMaxServOutstandingCalling" type="Integer16"/>
    <xsd:element name="negotiatedMaxServOutstandingCalled" type="Integer16"/>
    <xsd:element name="negotiatedDataStructureNestingLevel" type="Integer8"/>
    <xsd:element name="mmsInitResponseDetail" type="InitResponseDetail"/>
  </xsd:sequence>
  <!-- BEGIN not generated part -->
  <xsd:attribute name="version" type="Version" default="2018"/>
  <xsd:attribute name="revision" type="Revision" default="A"/>
  <xsd:attribute name="release" type="xsd:unsignedShort" default="1"/>
  <!-- END not generated part -->
</xsd:complexType>
<!-- From Clause 19 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="InitResponseDetail">
  <xsd:sequence>
    <xsd:element name="negotiatedVersionNumber" type="Integer16"/>
    <xsd:element name="negotiatedParameterCBB" type="ParameterSupportOptions"/>
    <xsd:element name="servicesSupportedCalled" type="ServiceSupportOptions"/>
  </xsd:sequence>
</xsd:complexType>
<!-- Remove SEQUENCE of implicit BOOLEAN and come back to named BIT STRING -->
<xsd:simpleType name="ServiceSupportOptions">
  <xsd:union>
    <xsd:simpleType>

```



```

<xsd:restriction base="xsd:string">
  <xsd:whiteSpace value="collapse"/>
  <xsd:pattern value="( *(1|0) *)*/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType>
  <xsd:list>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="status"/>
        <xsd:enumeration value="getNameList"/>
        <xsd:enumeration value="identify"/>
        <xsd:enumeration value="rename"/>
        <xsd:enumeration value="read"/>
        <xsd:enumeration value="write"/>
        <xsd:enumeration value="getVariableAccessAttributes"/>
        <xsd:enumeration value="defineNamedVariable"/>
        <xsd:enumeration value="defineScatteredAccess"/>
        <xsd:enumeration value="getScatteredAccessAttributes"/>
        <xsd:enumeration value="deleteVariableAccess"/>
        <xsd:enumeration value="defineNamedVariableList"/>
        <xsd:enumeration value="getNamedVariableListAttributes"/>
        <xsd:enumeration value="deleteNamedVariableList"/>
        <xsd:enumeration value="defineNamedType"/>
        <xsd:enumeration value="getNamedTypeAttributes"/>
        <xsd:enumeration value="deleteNamedType"/>
        <xsd:enumeration value="input"/>
        <xsd:enumeration value="output"/>
        <xsd:enumeration value="takeControl"/>
        <xsd:enumeration value="relinquishControl"/>
        <xsd:enumeration value="defineSemaphore"/>
        <xsd:enumeration value="deleteSemaphore"/>
        <xsd:enumeration value="reportSemaphoreStatus"/>
        <xsd:enumeration value="reportPoolSemaphoreStatus"/>
        <xsd:enumeration value="reportSemaphoreEntryStatus"/>
        <xsd:enumeration value="initiateDownloadSequence"/>
        <xsd:enumeration value="downloadSegment"/>
        <xsd:enumeration value="terminateDownloadSequence"/>
        <xsd:enumeration value="initiateUploadSequence"/>
        <xsd:enumeration value="uploadSegment"/>
        <xsd:enumeration value="terminateUploadSequence"/>
        <xsd:enumeration value="requestDomainDownload"/>
        <xsd:enumeration value="requestDomainUpload"/>
        <xsd:enumeration value="loadDomainContent"/>
        <xsd:enumeration value="storeDomainContent"/>
        <xsd:enumeration value="deleteDomain"/>
        <xsd:enumeration value="getDomainAttributes"/>
        <xsd:enumeration value="createProgramInvocation"/>
        <xsd:enumeration value="deleteProgramInvocation"/>
        <xsd:enumeration value="start"/>
        <xsd:enumeration value="stop"/>
        <xsd:enumeration value="resume"/>
        <xsd:enumeration value="reset"/>
        <xsd:enumeration value="kill"/>
        <xsd:enumeration value="getProgramInvocationAttributes"/>
        <xsd:enumeration value="obtainFile"/>
        <xsd:enumeration value="defineEventCondition"/>
        <xsd:enumeration value="deleteEventCondition"/>
        <xsd:enumeration value="getEventConditionAttributes"/>
        <xsd:enumeration value="reportEventConditionStatus"/>
        <xsd:enumeration value="alterEventConditionMonitoring"/>
        <xsd:enumeration value="triggerEvent"/>
        <xsd:enumeration value="defineEventAction"/>
        <xsd:enumeration value="deleteEventAction"/>
        <xsd:enumeration value="getEventActionAttributes"/>
        <xsd:enumeration value="reportActionStatus"/>
        <xsd:enumeration value="defineEventEnrollment"/>
        <xsd:enumeration value="deleteEventEnrollment"/>
        <xsd:enumeration value="alterEventEnrollment"/>
        <xsd:enumeration value="reportEventEnrollmentStatus"/>
        <xsd:enumeration value="getEventEnrollmentAttributes"/>
        <xsd:enumeration value="acknowledgeEventNotification"/>
        <xsd:enumeration value="getAlarmSummary"/>
        <xsd:enumeration value="getAlarmEnrollmentSummary"/>
        <xsd:enumeration value="readJournal"/>
        <xsd:enumeration value="writeJournal"/>
        <xsd:enumeration value="initializeJournal"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:list>
</xsd:simpleType>

```

```

        <xsd:enumeration value="reportJournalStatus"/>
        <xsd:enumeration value="createJournal"/>
        <xsd:enumeration value="deleteJournal"/>
        <xsd:enumeration value="getCapabilityList"/>
        <xsd:enumeration value="fileOpen"/>
        <xsd:enumeration value="fileRead"/>
        <xsd:enumeration value="fileClose"/>
        <xsd:enumeration value="fileRename"/>
        <xsd:enumeration value="fileDelete"/>
        <xsd:enumeration value="fileDirectory"/>
        <xsd:enumeration value="unsolicitedStatus"/>
        <xsd:enumeration value="informationReport"/>
        <xsd:enumeration value="eventNotification"/>
        <xsd:enumeration value="attachToEventCondition"/>
        <xsd:enumeration value="attachToSemaphore"/>
        <xsd:enumeration value="conclude"/>
        <xsd:enumeration value="cancel"/>
        <xsd:enumeration value="extendedGetNameList"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:list>
</xsd:simpleType>
</xsd:union>
</xsd:simpleType>
<!-- Remove SEQUENCE of implicit BOOLEAN and come back to named BIT STRING -->
<xsd:simpleType name="ParameterSupportOptions">
    <xsd:union>
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:whiteSpace value="collapse"/>
                <xsd:pattern value="( *(1|0) *)*/>
            </xsd:restriction>
        </xsd:simpleType>
        <xsd:simpleType>
            <xsd:list>
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:enumeration value="str1"/>
                        <xsd:enumeration value="str2"/>
                        <xsd:enumeration value="vnam"/>
                        <xsd:enumeration value="valt"/>
                        <xsd:enumeration value="vadr"/>
                        <xsd:enumeration value="vsca"/>
                        <xsd:enumeration value="tpy"/>
                        <xsd:enumeration value="vlis"/>
                        <xsd:enumeration value="real"/>
                        <xsd:enumeration value="cei"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:list>
        </xsd:simpleType>
    </xsd:union>
</xsd:simpleType>
<xsd:simpleType name="Integer16">
    <xsd:restriction base="INTEGER"/>
</xsd:simpleType>
<xsd:simpleType name="Integer8">
    <xsd:restriction base="INTEGER"/>
</xsd:simpleType>
<xsd:simpleType name="Integer32">
    <xsd:restriction base="INTEGER"/>
</xsd:simpleType>
<!-- Priority from clause 7.6.6 -->
<!--Priority::= INTEGER really Unsigned8 -->
<!--normalPriority Priority::= 64 -->
<!-- Severity from clause 7.6.7 -->
<!--Severity::= INTEGER really Unsigned8 -->
<!--normalSeverity Severity::= 64 -->
<!-- Clause 8 – Environment and General Management Protocol of ISO/IEC 9506-2:1990 -->
<!-- Initiate from Clause 8.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Initiate-RequestPDU">
    <xsd:sequence>
        <!--Optional for MMS, Mandatory for IEC 61850 -->
        <xsd:element name="localDetailCalling" type="Integer32"/>
        <xsd:element name="proposedMaxServOutstandingCalling" type="Integer16"/>
        <xsd:element name="proposedMaxServOutstandingCalled" type="Integer16"/>
        <xsd:element name="proposedDataStructureNestingLevel" type="Integer8"/>
    </xsd:sequence>

```

```
<xsd:element name="mmsInitRequestDetail" type="InitRequestDetail"/>
</xsd:sequence>
<!-- BEGIN not generated part -->
<xsd:attribute name="version" type="Version" default="2018"/>
<xsd:attribute name="revision" type="Revision" default="A"/>
<xsd:attribute name="release" type="xsd:unsignedShort" default="1"/>
<!-- END not generated part -->
</xsd:complexType>
<!-- From Clause 19 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="InitRequestDetail">
  <xsd:sequence>
    <xsd:element name="proposedVersionNumber" type="Integer16"/>
    <xsd:element name="proposedParameterCBB" type="ParameterSupportOptions"/>
    <xsd:element name="servicesSupportedCalling" type="ServiceSupportOptions"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Cancel-ErrorPDU">
  <xsd:sequence>
    <xsd:element name="originalInvokeID" type="Unsigned32"/>
    <xsd:element name="serviceError" type="ServiceError"/>
  </xsd:sequence>
</xsd:complexType>
<!-- ServiceError from Clause 7.5.5 of ISO/IEC 9506-2:1990 -->
<!-- Change named IMPLICIT INTEGER by CHOICE of IMPLICIT NULL -->
<xsd:complexType name="ServiceError">
  <xsd:sequence>
    <xsd:element name="errorClass">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="vmd-state">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="vmd-state-conflict" type="NULL"/>
                <xsd:element name="vmd-operational-problem" type="NULL"/>
                <xsd:element name="domain-transfer-problem" type="NULL"/>
                <xsd:element name="state-machine-id-invalid" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="application-reference">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="application-unreachable" type="NULL"/>
                <xsd:element name="connection-lost" type="NULL"/>
                <xsd:element name="application-reference-invalid" type="NULL"/>
                <xsd:element name="context-unsupported" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="definition">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="object-undefined" type="NULL"/>
                <xsd:element name="invalid-address" type="NULL"/>
                <xsd:element name="type-unsupported" type="NULL"/>
                <xsd:element name="type-inconsistent" type="NULL"/>
                <xsd:element name="object-exists" type="NULL"/>
                <xsd:element name="object-attribute-inconsistent" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="resource">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="other" type="NULL"/>
                <xsd:element name="memory-unavailable" type="NULL"/>
                <xsd:element name="processor-resource-unavailable" type="NULL"/>
                <xsd:element name="mass-storage-unavailable" type="NULL"/>
                <xsd:element name="capability-unavailable" type="NULL"/>
                <xsd:element name="capability-unknown" type="NULL"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="service">
```

```

<xsd:complexType>
  <xsd:choice>
    <xsd:element name="other" type="NULL"/>
    <xsd:element name="primitives-out-of-sequence" type="NULL"/>
    <xsd:element name="object-sate-conflict" type="NULL"/>
    <!--defined by NIST/ISP agreements -->
    <xsd:element name="pdu-size" type="NULL"/>
    <xsd:element name="continuation-invalid" type="NULL"/>
    <xsd:element name="object-constraint-conflict" type="NULL"/>
  </xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="service-preempt">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="timeout" type="NULL"/>
      <xsd:element name="deadlock" type="NULL"/>
      <xsd:element name="cancel" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="time-resolution">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="unsupportable-time-resolution" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="access">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="object-access-unsupported" type="NULL"/>
      <xsd:element name="object-non-existent" type="NULL"/>
      <xsd:element name="object-access-denied" type="NULL"/>
      <xsd:element name="object-invalidated" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="initiate">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <!--defined by NIST/ISP agreements -->
      <xsd:element name="version-incompatible" type="NULL"/>
      <!--defined by NIST/ISP agreements -->
      <xsd:element name="max-segment-insufficient" type="NULL"/>
      <xsd:element name="max-services-outstanding-calling-insufficient" type="NULL"/>
      <xsd:element name="max-services-outstanding-called-insufficient" type="NULL"/>
      <xsd:element name="service-CBB-insufficient" type="NULL"/>
      <xsd:element name="parameter-CBB-insufficient" type="NULL"/>
      <xsd:element name="nesting-level-insufficient" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="conclude">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="further-communication-required" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="cancel">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="other" type="NULL"/>
      <xsd:element name="invoke-id-unknown" type="NULL"/>
      <xsd:element name="cancel-not-possible" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="file">
  <xsd:complexType>
    <xsd:choice>

```

```

        <xsd:element name="other" type="NULL"/>
        <xsd:element name="filename-ambiguous" type="NULL"/>
        <xsd:element name="file-busy" type="NULL"/>
        <xsd:element name="filename-syntax-error" type="NULL"/>
        <xsd:element name="content-type-invalid" type="NULL"/>
        <xsd:element name="position-invalid" type="NULL"/>
        <xsd:element name="file-access-denied" type="NULL"/>
        <xsd:element name="file-non-existent" type="NULL"/>
        <xsd:element name="duplicate-filename" type="NULL"/>
        <xsd:element name="insufficient-space-in-filestore" type="NULL"/>
    </xsd:choice>
</xsd:complexType>
</xsd:element>
<!-- not used by IEC 61850 -->
<xsd:element name="others" type="INTEGER"/>
</xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="additionalCode" type="INTEGER" minOccurs="0"/>
<xsd:element name="additionalDescription" type="VisibleString" minOccurs="0"/>
<xsd:element name="serviceSpecificInformation" minOccurs="0">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="obtainFile" type="ObtainFile-Error"/>
            <!-- start [1] IMPLICIT Start-Error, not used by IEC 61850 -->
            <!-- stop [2] IMPLICIT Stop-Error, not used by IEC 61850 -->
            <!-- resume [3] IMPLICIT Resume-Error, not used by IEC 61850 -->
            <!-- reset [4] IMPLICIT Reset-Error, not used by IEC 61850 -->
            <!-- deleteVariableAccess [5] IMPLICIT DeleteVariableAccess-Error, not used by IEC 61850 -->
            <!-- deleteNamedType [7] IMPLICIT DeleteNamedType-Error, not used by IEC 61850 -->
            <!-- defineEventEnrollment-Error [8] DefineEventEnrollment-Error, not used by IEC 61850 -->
            <!-- fileRename [9] IMPLICIT FileRename-Error -->
            <!-- additionalService [10] AdditionalServiceError not used by IEC 61850 -->
            <xsd:element name="deleteNamedVariableList" type="DeleteNamedVariableList-Error"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<!-- originalInvokeID -->
<xsd:simpleType name="Cancel-ResponsePDU">
    <xsd:restriction base="Unsigned32"/>
</xsd:simpleType>
<!-- Abort from Clause 8.4 of ISO/IEC 9506-2:1990 -->
<!-- The abort service is directly mapped to the ACSE A-Abort service (see clause 17 of ISO/IEC 9506-2:1990) -->
<!-- Cancel from Clause 8.5 of ISO/IEC 9506-2:1990 -->
<!-- originalInvokeID -->
<xsd:simpleType name="Cancel-RequestPDU">
    <xsd:restriction base="Unsigned32"/>
</xsd:simpleType>
<!-- Reject from Clause 8.6 of ISO/IEC 9506-2:1990 -->
<!-- Change named IMPLICIT INTEGER by CHOICE of IMPLICIT NULL -->
<xsd:complexType name="RejectPDU">
    <xsd:sequence>
        <xsd:element name="originalInvokeID" type="Unsigned32" minOccurs="0"/>
        <xsd:element name="rejectReason">
            <xsd:complexType>
                <xsd:choice>
                    <xsd:element name="confirmed-requestPDU">
                        <xsd:complexType>
                            <xsd:choice>
                                <xsd:element name="other" type="NULL"/>
                                <xsd:element name="unrecognized-service" type="NULL"/>
                                <xsd:element name="unrecognized-modifier" type="NULL"/>
                                <xsd:element name="invalid-invokeID" type="NULL"/>
                                <xsd:element name="invalid-argument" type="NULL"/>
                                <xsd:element name="invalid-modifier" type="NULL"/>
                                <xsd:element name="max-serv-outstanding-exceeded" type="NULL"/>
                                <!-- Value 7 reserved for further definition -->
                                <xsd:element name="max-recursion-exceeded" type="NULL"/>
                                <xsd:element name="value-out-of-range" type="NULL"/>
                            </xsd:choice>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="confirmed-responsePDU">
                        <xsd:complexType>
                            <xsd:choice>

```

```

        <xsd:element name="other" type="NULL"/>
        <xsd:element name="unrecognized-service" type="NULL"/>
        <xsd:element name="invalid-invokeID" type="NULL"/>
        <xsd:element name="invalid-result" type="NULL"/>
        <!-- Value 4 reserved for further definition -->
        <xsd:element name="max-recursion-exceeded" type="NULL"/>
        <xsd:element name="value-out-of-range" type="NULL"/>
    </xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="confirmed-errorPDU">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="unrecognized-service" type="NULL"/>
            <xsd:element name="invalid-invokeID" type="NULL"/>
            <xsd:element name="invalid-serviceError" type="NULL"/>
            <xsd:element name="value-out-of-range" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="unconfirmedPDU">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="unrecognized-service" type="NULL"/>
            <xsd:element name="invalid-argument" type="NULL"/>
            <xsd:element name="max-recursion-exceeded" type="NULL"/>
            <xsd:element name="value-out-of-range" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="pdu-error">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="unknown-pdu-type" type="NULL"/>
            <xsd:element name="invalid-pdu" type="NULL"/>
            <xsd:element name="illegal-acse-mapping" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="cancel-requestPDU">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="invalid-invokeID" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="cancel-responsePDU">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="invalid-invokeID" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="cancel-errorPDU">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="invalid-invokeID" type="NULL"/>
            <xsd:element name="invalid-serviceError" type="NULL"/>
            <xsd:element name="value-out-of-range" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="conclude-requestPDU">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="invalid-argument" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="conclude-responsePDU">
    <xsd:complexType>

```

```

        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="invalid-result" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
<xsd:element name="conclude-errorPDU">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element name="other" type="NULL"/>
            <xsd:element name="invalid-serviceError" type="NULL"/>
            <xsd:element name="value-out-of-range" type="NULL"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<!-- Unconfirmed-PDU from Clause 7.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Unconfirmed-PDU">
    <xsd:sequence>
        <xsd:element name="UnconfirmedService" type="UnconfirmedService" form="qualified"/>
    </xsd:sequence>
</xsd:complexType>
<!-- UnconfirmedService from Clause 7.5.3 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="UnconfirmedService">
    <xsd:choice>
        <xsd:element name="informationReport" type="InformationReport"/>
    </xsd:choice>
</xsd:complexType>
<!-- InformationReport from Clause 12.8 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="InformationReport">
    <xsd:sequence>
        <xsd:element name="variableAccessSpecification" type="VariableAccessSpecification"/>
        <xsd:element name="listOfAccessResult">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="AccessResult" type="AccessResult" form="qualified"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<!-- AccessResult from Clause 12.4.1 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="AccessResult">
    <xsd:choice>
        <xsd:element name="failure" type="DataAccessError"/>
        <xsd:element name="success" type="Data"/>
    </xsd:choice>
</xsd:complexType>
<!-- Data from Clause 12.4.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Data">
    <xsd:choice>
        <!-- context tag 0 is reserved for AccessResult -->
        <xsd:element name="array">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="Data" type="Data" form="qualified"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="structure">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="Data" type="Data" form="qualified"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="boolean" type="BOOLEAN"/>
        <xsd:element name="bit-string" type="BIT_STRING"/>
        <xsd:element name="integer" type="INTEGER"/>
        <xsd:element name="unsigned" type="INTEGER"/>
        <xsd:element name="floating-point" type="FloatingPoint"/>
        <!-- not used for IEC 61850 -->
        <xsd:element name="real" type="REAL"/>
    </xsd:choice>

```



```

<xsd:element name="octet-string" type="OCTET_STRING"/>
<xsd:element name="visible-string" type="VisibleString"/>
<xsd:element name="binary-time" type="TimeOfDay"/>
<!-- not used for IEC 61850 -->
<xsd:element name="bcd" type="INTEGER"/>
<!-- not used for IEC 61850 -->
<xsd:element name="booleanArray" type="BIT_STRING"/>
<xsd:element name="objId" type="OBJECT_IDENTIFIER"/>
<!-- added by a later version of ISO 9506, not used by IEC 61850 -->
<xsd:element name="mMSString" type="MMSString"/>
<!-- UTC Time, added by IEC 61850, format defined in IEC 61850-8-1 -->
<xsd:element name="utc-time" type="UtcTime"/>
</xsd:choice>
</xsd:complexType>
<xsd:simpleType name="UtcTime">
  <xsd:restriction base="OCTET_STRING"/>
</xsd:simpleType>
<xsd:simpleType name="OCTET_STRING">
  <xsd:restriction base="xsd:string">
    <xsd:whiteSpace value="collapse"/>
    <xsd:pattern value="( *([0-9][A-F][a-f] *)*)"/>
  </xsd:restriction>
</xsd:simpleType>
<!-- MMSString::= VisibleString -->
<xsd:complexType name="MMSString" mixed="true">
  <xsd:complexContent mixed="true">
    <xsd:extension base="UTF8String"/>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="UTF8String" mixed="true">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:choice>
      <xsd:element name="nul" type="NULL"/>
      <xsd:element name="soh" type="NULL"/>
      <xsd:element name="stx" type="NULL"/>
      <xsd:element name="etx" type="NULL"/>
      <xsd:element name="eot" type="NULL"/>
      <xsd:element name="enq" type="NULL"/>
      <xsd:element name="ack" type="NULL"/>
      <xsd:element name="bel" type="NULL"/>
      <xsd:element name="bs" type="NULL"/>
      <xsd:element name="vt" type="NULL"/>
      <xsd:element name="ff" type="NULL"/>
      <xsd:element name="so" type="NULL"/>
      <xsd:element name="si" type="NULL"/>
      <xsd:element name="dle" type="NULL"/>
      <xsd:element name="dc1" type="NULL"/>
      <xsd:element name="dc2" type="NULL"/>
      <xsd:element name="dc3" type="NULL"/>
      <xsd:element name="dc4" type="NULL"/>
      <xsd:element name="nak" type="NULL"/>
      <xsd:element name="syn" type="NULL"/>
      <xsd:element name="etb" type="NULL"/>
      <xsd:element name="can" type="NULL"/>
      <xsd:element name="em" type="NULL"/>
      <xsd:element name="sub" type="NULL"/>
      <xsd:element name="esc" type="NULL"/>
      <xsd:element name="is4" type="NULL"/>
      <xsd:element name="is3" type="NULL"/>
      <xsd:element name="is2" type="NULL"/>
      <xsd:element name="is1" type="NULL"/>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="OBJECT_IDENTIFIER">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="( *[a-z][0-9]\{\|\}\.)* */>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="BIT_STRING">
  <xsd:restriction base="xsd:string">
    <xsd:whiteSpace value="collapse"/>
    <xsd:pattern value="( *(1|0) *) */>
  </xsd:restriction>
</xsd:simpleType>
<!-- Common MMS Types from Clause 7.6 of ISO/IEC 9506-2:1990 -->
<!-- from clause 7.6.1, Restricted to 6 Octets since it is only used for EntryTime -->

```



```

<xsd:simpleType name="TimeOfDay">
  <xsd:restriction base="OCTET_STRING">
    <xsd:pattern value="( *[0-9][A-F][a-f] *) {12,12}"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="REAL">
  <xsd:restriction base="xsd:double"/>
</xsd:simpleType>
<xsd:simpleType name="FloatingPoint">
  <xsd:restriction base="OCTET_STRING"/>
</xsd:simpleType>
<xsd:simpleType name="BOOLEAN">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:complexType name="DataAccessError">
  <xsd:choice>
    <xsd:element name="object-invalidated" type="NULL"/>
    <xsd:element name="hardware-fault" type="NULL"/>
    <xsd:element name="temporarily-unavailable" type="NULL"/>
    <xsd:element name="object-access-denied" type="NULL"/>
    <xsd:element name="object-undefined" type="NULL"/>
    <xsd:element name="invalid-address" type="NULL"/>
    <xsd:element name="type-unsupported" type="NULL"/>
    <xsd:element name="type-inconsistent" type="NULL"/>
    <xsd:element name="object-attribute-inconsistent" type="NULL"/>
    <xsd:element name="object-access-unsupported" type="NULL"/>
    <xsd:element name="object-non-existent" type="NULL"/>
    <xsd:element name="object-value-invalid" type="NULL"/>
  </xsd:choice>
</xsd:complexType>
<!-- VariableAccessSpecification from Clause 12.5.1 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="VariableAccessSpecification">
  <xsd:choice>
    <xsd:element name="listOfVariable">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="SEQUENCE">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="variableSpecification" type="VariableSpecification"/>
                <xsd:element name="alternateAccess" type="AlternateAccess" minOccurs="0"/>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:choice>
    <xsd:element name="variableListName" type="ObjectName"/>
  </xsd:choice>
</xsd:complexType>
<!-- ObjectName from clause 7.6.3 -->
<xsd:complexType name="ObjectName">
  <xsd:choice>
    <xsd:element name="vmd-specific" type="Identifier"/>
    <xsd:element name="domain-specific">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="domainId" type="Identifier"/>
          <xsd:element name="itemId" type="Identifier"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="aa-specific" type="Identifier"/>
  </xsd:choice>
</xsd:complexType>
<!-- Identifiers and Integer Types from clause 7.6.2 -->
<!-- for IEC 61850, the size is 64 Octets -->
<xsd:simpleType name="Identifier">
  <xsd:restriction base="VisibleString"/>
</xsd:simpleType>
<!-- AlternateAccess from Clause 12.3.1 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="AlternateAccess">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="CHOICE">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="unnamed" type="AlternateAccessSelection"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

```

        <xsd:element name="named">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="componentName" type="Identifier"/>
                    <xsd:element name="access" type="AlternateAccessSelection"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:choice>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="AlternateAccessSelection">
    <xsd:choice>
        <xsd:element name="selectAlternateAccess">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="accessSelection">
                        <xsd:complexType>
                            <xsd:choice>
                                <xsd:element name="component" type="Identifier"/>
                                <xsd:element name="index" type="Unsigned32"/>
                                <xsd:element name="indexRange">
                                    <xsd:complexType>
                                        <xsd:sequence>
                                            <xsd:element name="lowIndex" type="Unsigned32"/>
                                            <xsd:element name="numberOfElements" type="Unsigned32"/>
                                        </xsd:sequence>
                                    </xsd:complexType>
                                </xsd:element>
                                <!-- all array elements -->
                                <xsd:element name="allElements" type="NULL"/>
                            </xsd:choice>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="alternateAccess" type="AlternateAccess"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="selectAccess">
            <xsd:complexType>
                <xsd:choice>
                    <xsd:element name="component" type="Identifier"/>
                    <xsd:element name="index" type="Unsigned32"/>
                    <xsd:element name="indexRange">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="lowIndex" type="Unsigned32"/>
                                <xsd:element name="numberOfElements" type="Unsigned32"/>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                    <!-- all array elements -->
                    <xsd:element name="allElements" type="NULL"/>
                </xsd:choice>
            </xsd:complexType>
        </xsd:element>
    </xsd:choice>
</xsd:complexType>
<!-- VariableSpecification from Clause 12.5.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="VariableSpecification">
    <xsd:choice>
        <xsd:element name="name" type="ObjectName"/>
        <!-- this is out of scope of IEC 61850 -->
        <xsd:element name="address" type="Address"/>
        <!-- this is out of scope of IEC 61850 -->
        <!-- invalidated [4] IMPLICIT NULL excluded for IEC 61850 -->
        <xsd:element name="variableDescription">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="address" type="Address"/>
                    <xsd:element name="typeSpecification" type="TypeSpecification"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:choice>

```

```

</xsd:complexType>
<!-- Clause 12 -Variable Access Protocol of ISO/IEC 9506-2:1990 -->
<!-- TypeSpecification from Clause 12.2.1 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="TypeSpecification">
  <xsd:choice>
    <!-- not used for IEC 61850 -->
    <xsd:element name="typeName" type="ObjectName"/>
    <xsd:element name="array">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="numberOfElements" type="Unsigned32"/>
          <xsd:element name="elementType" type="TypeSpecification"/>
        </xsd:sequence>
        <xsd:attribute name="packed" type="MMS_BOOLEAN" default="false"/>
        <!-- not used for IEC 61850 -->
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="structure">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="components">
            <xsd:complexType>
              <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="SEQUENCE">
                  <xsd:complexType>
                    <xsd:sequence>
                      <xsd:element name="componentName" type="Identifier" minOccurs="0"/>
                      <xsd:element name="componentType" type="TypeSpecification"/>
                    </xsd:sequence>
                  </xsd:complexType>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:attribute name="packed" type="MMS_BOOLEAN_1" default="false"/>
        </xsd:complexType>
      </xsd:element>
    <!-- Simple Type -->
    <!-- Size -->
    <!-- boolean -->
    <xsd:element name="boolean" type="NULL"/>
    <xsd:element name="bit-string" type="Integer32"/>
    <xsd:element name="integer" type="Unsigned8"/>
    <xsd:element name="unsigned" type="Unsigned8"/>
    <xsd:element name="floating-point">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="format-width" type="Unsigned8"/>
          <xsd:element name="exponent-width" type="Unsigned8"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <!-- not used in IEC 61850 -->
    <xsd:element name="real">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="base">
            <xsd:simpleType>
              <xsd:restriction base="INTEGER">
                <xsd:minInclusive value="2"/>
                <xsd:maxInclusive value="10"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="exponent" type="INTEGER" minOccurs="0"/>
          <xsd:element name="mantissa" type="INTEGER" minOccurs="0"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="octet-string" type="Integer32"/>
    <xsd:element name="visible-string" type="Integer32"/>
    <xsd:element name="generalized-time" type="NULL"/>
    <xsd:element name="binary-time" type="BOOLEAN_2"/>
    <!-- not used in IEC 61850 -->
  </xsd:choice>

```

```

    <xsd:element name="bcd" type="Unsigned8"/>
    <xsd:element name="objId" type="NULL"/>
    <!-- added by a later version of ISO 9506, not used by IEC 61850 -->
    <xsd:element name="mMSString" type="Integer32"/>
    <!-- UTC Time, added by IEC 61850, defined in IEC 61850-8-1 -->
    <xsd:element name="utc-time" type="NULL"/>
  </xsd:choice>
</xsd:complexType>
<xsd:simpleType name="Unsigned8">
  <xsd:restriction base="INTEGER"/>
</xsd:simpleType>
<xsd:simpleType name="BOOLEAN_2">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:simpleType name="MMS_BOOLEAN_1">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:simpleType name="MMS_BOOLEAN">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<!-- Address from Clause 12.5.4 of ISO/IEC 9506-2:1990 out of scope of IEC 61850 -->
<xsd:complexType name="Address">
  <xsd:choice>
    <xsd:element name="numericAddress" type="Unsigned32"/>
    <xsd:element name="symbolicAddress" type="VisibleString"/>
    <xsd:element name="unconstrainedAddress" type="OCTET_STRING"/>
  </xsd:choice>
</xsd:complexType>
<!-- Confirmed-ErrorPDU from Clause 7.4 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Confirmed-ErrorPDU">
  <xsd:sequence>
    <xsd:element name="invokeID" type="Unsigned32"/>
    <xsd:element name="modifierPosition" type="Unsigned32" minOccurs="0"/>
    <xsd:element name="serviceError" type="ServiceError"/>
  </xsd:sequence>
</xsd:complexType>
<!-- Confirmed-ResponsePDU from Clause 7.3 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Confirmed-ResponsePDU">
  <xsd:sequence>
    <xsd:element name="invokeID" type="Unsigned32"/>
    <xsd:element name="ConfirmedServiceResponse" type="ConfirmedServiceResponse" form="qualified"/>
  </xsd:sequence>
</xsd:complexType>
<!-- ConfirmedServiceResponse from Clause 7.3.4 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="ConfirmedServiceResponse">
  <xsd:choice>
    <!-- not required for IEC 61850, but suggested to implement -->
    <xsd:element name="status" type="Status-Response"/>
    <xsd:element name="getNameList" type="GetNameList-Response"/>
    <!-- not required for IEC 61850, but mandatory for MMS -->
    <xsd:element name="identify" type="Identify-Response"/>
    <xsd:element name="read" type="Read-Response"/>
    <xsd:element name="write" type="Write-Response"/>
    <xsd:element name="getVariableAccessAttributes" type="GetVariableAccessAttributes-Response"/>
    <xsd:element name="defineNamedVariableList" type="DefineNamedVariableList-Response"/>
    <xsd:element name="getNamedVariableListAttributes" type="GetNamedVariableListAttributes-Response"/>
    <xsd:element name="deleteNamedVariableList" type="DeleteNamedVariableList-Response"/>
    <xsd:element name="getDomainAttributes" type="GetDomainAttributes-Response"/>
    <xsd:element name="obtainFile" type="ObtainFile-Response"/>
    <xsd:element name="fileOpen" type="FileOpen-Response"/>
    <xsd:element name="readJournal" type="ReadJournal-Response"/>
    <!-- not required for IEC 61850, but suggested to implement -->
    <xsd:element name="getCapabilityList" type="GetCapabilityList-Response"/>
    <xsd:element name="fileRead" type="FileRead-Response"/>
    <xsd:element name="fileClose" type="FileClose-Response"/>
    <xsd:element name="fileDelete" type="FileDelete-Response"/>
    <xsd:element name="fileDirectory" type="FileDirectory-Response"/>
    <!-- added for IEC 61850-8-2 -->
    <xsd:element name="extendedGetNameList" type="GetNameList-Response"/>
  </xsd:choice>
</xsd:complexType>
<xsd:complexType name="GetNameList-Response">
  <xsd:sequence>
    <xsd:element name="listOfIdentifier">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="Identifier" type="Identifier" form="qualified"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

```

        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="moreFollows" type="MMS_BOOLEAN_2" default="true"/>
</xsd:complexType>
<xsd:simpleType name="MMS_BOOLEAN_2">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:complexType name="FileDirectory-Response">
  <xsd:sequence>
    <xsd:element name="listOfDirectoryEntry">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="DirectoryEntry" type="DirectoryEntry" form="qualified"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="moreFollows" type="MMS_BOOLEAN_3" default="false"/>
</xsd:complexType>
<xsd:simpleType name="MMS_BOOLEAN_3">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:complexType name="DirectoryEntry">
  <xsd:sequence>
    <xsd:element name="fileName" type="FileName"/>
    <xsd:element name="fileAttributes" type="FileAttributes"/>
  </xsd:sequence>
</xsd:complexType>
<!-- FileAttributes from Clause C.8 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="FileAttributes">
  <xsd:sequence>
    <xsd:element name="sizeOfFile" type="Unsigned32"/>
    <xsd:element name="lastModified" type="GeneralizedTime" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="GeneralizedTime">
  <xsd:restriction base="xsd:string">
    <xsd:whiteSpace value="collapse"/>
    <xsd:pattern value="([0-9]{8}([0-9]{2})?([0-9]{2})?([0-9]{2})?(\.[0-9]+)?((Z)|(\+|-)[0-9]{4}))?" />
  </xsd:restriction>
</xsd:simpleType>
<!-- FileName from clause 7.6.5 -->
<xsd:complexType name="FileName">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="GraphicStringMMS" type="GraphicStringMMS" form="qualified"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="GraphicStringMMS">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[ -~]{1,64}" />
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="FileDelete-Response">
  <xsd:restriction base="NULL"/>
</xsd:simpleType>
<xsd:simpleType name="FileClose-Response">
  <xsd:restriction base="NULL"/>
</xsd:simpleType>
<xsd:complexType name="FileRead-Response">
  <xsd:sequence>
    <xsd:element name="fileData" type="OCTET_STRING"/>
  </xsd:sequence>
  <xsd:attribute name="moreFollows" type="MMS_BOOLEAN_4" default="true"/>
</xsd:complexType>
<xsd:simpleType name="MMS_BOOLEAN_4">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:complexType name="GetCapabilityList-Response">
  <xsd:sequence>
    <xsd:element name="listOfCapabilities">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="VisibleString" type="VisibleString"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

```

    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="moreFollows" type="MMS_BOOLEAN_5" default="true"/>
</xsd:complexType>
<xsd:simpleType name="MMS_BOOLEAN_5">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:complexType name="ReadJournal-Response">
  <xsd:sequence>
    <xsd:element name="listOfJournalEntry">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="JournalEntry" type="JournalEntry" form="qualified"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="moreFollows" type="MMS_BOOLEAN_6" default="false"/>
</xsd:complexType>
<xsd:simpleType name="MMS_BOOLEAN_6">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:complexType name="JournalEntry">
  <xsd:sequence>
    <xsd:element name="entryIdentifier" type="OCTET_STRING"/>
    <xsd:element name="originatingApplication" type="ApplicationReference"/>
    <xsd:element name="entryContent" type="EntryContent"/>
  </xsd:sequence>
</xsd:complexType>
<!-- EntryContent from Clause 16.8.1 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="EntryContent">
  <xsd:sequence>
    <xsd:element name="occurrenceTime" type="TimeOfDay"/>
    <xsd:element name="entryForm">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="data">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="event", minOccurs="0">
                  <xsd:complexType>
                    <xsd:sequence>
                      <xsd:element name="eventConditionName" type="ObjectName"/>
                      <xsd:element name="currentState" type="EC-State"/>
                    </xsd:sequence>
                  </xsd:complexType>
                </xsd:element>
                <xsd:element name="listOfVariables" minOccurs="0">
                  <xsd:complexType>
                    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                      <xsd:element name="SEQUENCE">
                        <xsd:complexType>
                          <xsd:sequence>
                            <xsd:element name="variableTag" type="VisibleString"/>
                            <xsd:element name="valueSpecification" type="Data"/>
                          </xsd:sequence>
                        </xsd:complexType>
                      </xsd:element>
                    </xsd:sequence>
                  </xsd:complexType>
                </xsd:element>
                <xsd:element name="annotation" type="VisibleString"/>
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="EC-State">
      <xsd:union memberTypes="xsd:integer">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="disabled"/>
            <xsd:enumeration value="idle"/>
            <xsd:enumeration value="active"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:union>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

```

        </xsd:restriction>
    </xsd:simpleType>
</xsd:union>
</xsd:simpleType>
<!-- ApplicationReference from clause 7.6.4 -->
<xsd:complexType name="ApplicationReference">
    <xsd:sequence>
        <xsd:element name="ap-title" type="OBJECT_IDENTIFIER" minOccurs="0"/>
        <xsd:element name="ap-invocation-id" type="INTEGER" minOccurs="0"/>
        <xsd:element name="ae-qualifier" type="INTEGER" minOccurs="0"/>
        <xsd:element name="ae-invocation-id" type="INTEGER" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="FileOpen-Response">
    <xsd:sequence>
        <xsd:element name="frsmID" type="Integer32"/>
        <xsd:element name="fileAttributes" type="FileAttributes"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="ObtainFile-Response">
    <xsd:restriction base="NULL"/>
</xsd:simpleType>
<xsd:complexType name="GetDomainAttributes-Response">
    <xsd:sequence>
        <xsd:element name="listOfCapabilities">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="VisibleString" type="VisibleString"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="state" type="DomainState"/>
        <xsd:element name="mmsDeletable" type="BOOLEAN_3"/>
        <xsd:element name="sharable" type="BOOLEAN_4"/>
        <!-- PI Names -->
        <xsd:element name="listOfProgramInvocations">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="Identifier" type="Identifier" form="qualified"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="uploadInProgress" type="Integer8"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="BOOLEAN_4">
    <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:simpleType name="BOOLEAN_3">
    <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:simpleType name="DomainState">
    <xsd:union memberTypes="xsd:integer">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="non-existent"/>
                <xsd:enumeration value="loading"/>
                <xsd:enumeration value="ready"/>
                <xsd:enumeration value="in-use"/>
                <xsd:enumeration value="complete"/>
                <xsd:enumeration value="incomplete"/>
                <xsd:enumeration value="d1"/>
                <xsd:enumeration value="d2"/>
                <xsd:enumeration value="d3"/>
                <xsd:enumeration value="d4"/>
                <xsd:enumeration value="d5"/>
                <xsd:enumeration value="d6"/>
                <xsd:enumeration value="d7"/>
                <xsd:enumeration value="d8"/>
                <xsd:enumeration value="d9"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:union>
</xsd:simpleType>
<xsd:complexType name="DeleteNamedVariableList-Response">
    <xsd:sequence>
        <xsd:element name="numberMatched" type="Unsigned32"/>
    </xsd:sequence>
</xsd:complexType>

```



```

        <xsd:element name="numberDeleted" type="Unsigned32"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="GetNamedVariableListAttributes-Response">
    <xsd:sequence>
        <xsd:element name="mmsDeletable" type="BOOLEAN_5"/>
        <xsd:element name="listOfVariable">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="SEQUENCE">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="variableSpecification" type="VariableSpecification"/>
                                <xsd:element name="alternateAccess" type="AlternateAccess" minOccurs="0"/>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="BOOLEAN_5">
    <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:simpleType name="DefineNamedVariableList-Response">
    <xsd:restriction base="NULL"/>
</xsd:simpleType>
<xsd:complexType name="GetVariableAccessAttributes-Response">
    <xsd:sequence>
        <xsd:element name="mmsDeletable" type="BOOLEAN_6"/>
        <!-- out of scope of IEC 61850 -->
        <xsd:element name="address" type="Address" minOccurs="0"/>
        <!-- out of scope of IEC 61850 -->
        <xsd:element name="typeSpecification" type="TypeSpecification"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="BOOLEAN_6">
    <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<xsd:complexType name="Write-Response">
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="CHOICE">
            <xsd:complexType>
                <xsd:choice>
                    <xsd:element name="failure" type="DataAccessError"/>
                    <xsd:element name="success" type="NULL"/>
                </xsd:choice>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Read-Response">
    <xsd:sequence>
        <xsd:element name="variableAccessSpecification" type="VariableAccessSpecification" minOccurs="0"/>
        <xsd:element name="listOfAccessResult">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="AccessResult" type="AccessResult" form="qualified"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Identify-Response">
    <xsd:sequence>
        <xsd:element name="vendorName" type="VisibleString"/>
        <xsd:element name="modelName" type="VisibleString"/>
        <xsd:element name="revision" type="VisibleString"/>
        <xsd:element name="listOfAbstractSyntaxes" minOccurs="0">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="OBJECT_IDENTIFIER" type="OBJECT_IDENTIFIER"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

```



```

</xsd:complexType>
<!-- Change named IMPLICIT INTEGER by CHOICE of IMPLICIT NULL -->
<xsd:complexType name="Status-Response">
  <xsd:sequence>
    <xsd:element name="vmdLogicalStatus">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="state-changes-allowed" type="NULL"/>
          <xsd:element name="no-state-changes-allowed" type="NULL"/>
          <xsd:element name="limited-services-permitted" type="NULL"/>
          <xsd:element name="support-services-allowed" type="NULL"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="vmdPhysicalStatus">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="operational" type="NULL"/>
          <xsd:element name="partially-operational" type="NULL"/>
          <xsd:element name="inoperable" type="NULL"/>
          <xsd:element name="needs-commissioning" type="NULL"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="localDetail" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="BIT STRING">
          <xsd:pattern value="( *{0,1} *){0,128}" />
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<!-- Confirmed-RequestPDU from Clause 7.1 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Confirmed-RequestPDU">
  <xsd:sequence>
    <xsd:element name="invokeID" type="Unsigned32"/>
    <xsd:element name="ConfirmedServiceRequest" type="ConfirmedServiceRequest" form="qualified"/>
  </xsd:sequence>
</xsd:complexType>
<!-- ConfirmedServiceRequest from Clause 7.5.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="ConfirmedServiceRequest">
  <xsd:choice>
    <!--not required for IEC 61850, but suggested to implement -->
    <xsd:element name="status" type="Status-Request"/>
    <xsd:element name="getNameList" type="GetNameList-Request"/>
    <!--not required by IEC 61850, but Mandatory for MMS -->
    <xsd:element name="identify" type="Identify-Request"/>
    <xsd:element name="read" type="Read-Request"/>
    <xsd:element name="write" type="Write-Request"/>
    <xsd:element name="getVariableAccessAttributes" type="GetVariableAccessAttributes-Request"/>
    <xsd:element name="defineNamedVariableList" type="DefineNamedVariableList-Request"/>
    <xsd:element name="getNamedVariableListAttributes" type="GetNamedVariableListAttributes-Request"/>
    <xsd:element name="deleteNamedVariableList" type="DeleteNamedVariableList-Request"/>
    <!--not required for IEC 61850, but suggested to implement -->
    <xsd:element name="getDomainAttributes" type="GetDomainAttributes-Request"/>
    <xsd:element name="obtainFile" type="ObtainFile-Request"/>
    <xsd:element name="readJournal" type="ReadJournal-Request"/>
    <!--not required for IEC 61850, but suggested to implement -->
    <xsd:element name="getCapabilityList" type="GetCapabilityList-Request"/>
    <xsd:element name="fileOpen" type="FileOpen-Request"/>
    <xsd:element name="fileRead" type="FileRead-Request"/>
    <xsd:element name="fileClose" type="FileClose-Request"/>
    <xsd:element name="fileDelete" type="FileDelete-Request"/>
    <xsd:element name="fileDirectory" type="FileDirectory-Request"/>
    <!-- added for IEC 61850-8-2 -->
    <xsd:element name="extendedGetNameList" type="GetNameList-Request"/>
  </xsd:choice>
</xsd:complexType>
<!-- GetNameList from Clause 9.4 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="GetNameList-Request">
  <xsd:sequence>
    <xsd:element name="extendedObjectClass">
      <xsd:complexType>
        <xsd:choice>
          <!-- csObjectClass [1] CsAdditionalObjectClasses not used for IEC 61850 -->
          <xsd:element name="objectClass">

```

```

<xsd:complexType>
  <xsd:choice>
    <xsd:element name="namedVariable" type="NULL"/>
    <!-- use deprecated -->
    <xsd:element name="scatteredAccess" type="NULL"/>
    <xsd:element name="namedVariableList" type="NULL"/>
    <!-- not used for IEC 61850 -->
    <xsd:element name="namedType" type="NULL"/>
    <!-- not used for IEC 61850 -->
    <xsd:element name="semaphore" type="NULL"/>
    <!-- not used for IEC 61850 -->
    <xsd:element name="eventCondition" type="NULL"/>
    <!-- not used for IEC 61850 -->
    <xsd:element name="eventAction" type="NULL"/>
    <!-- not used for IEC 61850 -->
    <xsd:element name="eventEnrollment" type="NULL"/>
    <xsd:element name="journal" type="NULL"/>
    <xsd:element name="domain" type="NULL"/>
    <!-- not used for IEC 61850 -->
    <xsd:element name="programInvocation" type="NULL"/>
    <!-- not used for IEC 61850 -->
    <xsd:element name="operatorStation" type="NULL"/>
  </xsd:choice>
</xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:complexType>
</xsd:element>
<xsd:element name="objectScope">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="vmdSpecific" type="NULL"/>
      <xsd:element name="domainSpecific" type="Identifier"/>
      <xsd:element name="aaSpecific" type="NULL"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="continueAfter" type="Identifier" minOccurs="0"/>
</xsd:sequence>
</xsd:complexType>
<!-- FileDirectory from Clause C.7 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="FileDirectory-Request">
  <xsd:sequence>
    <xsd:element name="fileSpecification" type="FileName" minOccurs="0"/>
    <xsd:element name="continueAfter" type="FileName" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<!-- FileDelete Productions from Clause C.6 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="FileDelete-Request">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="GraphicStringMMS" type="GraphicStringMMS" form="qualified"/>
  </xsd:sequence>
</xsd:complexType>
<!-- FileClose from Clause C.3 of ISO/IEC 9506-2:1990 -->
<!-- FRSMID -->
<xsd:simpleType name="FileClose-Request">
  <xsd:restriction base="Integer32"/>
</xsd:simpleType>
<!-- FileRead from Clause C.3 of ISO/IEC 9506-2:1990 -->
<!-- FRSMID -->
<xsd:simpleType name="FileRead-Request">
  <xsd:restriction base="Integer32"/>
</xsd:simpleType>
<!-- Annex C -File Management Protocol of ISO/IEC 9506-2:1990 -->
<!-- FileOpen from Clause C.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="FileOpen-Request">
  <xsd:sequence>
    <xsd:element name="fileName" type="FileName"/>
    <xsd:element name="initialPosition" type="Unsigned32"/>
  </xsd:sequence>
</xsd:complexType>
<!-- GetCapabilist from Clause 9.7 of ISO/IEC 9506-2:1990 not used for IEC 61850 -->
<xsd:complexType name="GetCapabilityList-Request">
  <xsd:sequence>
    <xsd:element name="continueAfter" type="VisibleString" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

```

```

<!-- Clause 16 -Journal Management Protocol of ISO/IEC 9506-2:1990 -->
<!-- ReadJournal from Clause 16.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="ReadJournal-Request">
  <xsd:sequence>
    <xsd:element name="journalName" type="ObjectName"/>
    <xsd:element name="rangeStartSpecification" minOccurs="0">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="startingTime" type="TimeOfDay"/>
          <xsd:element name="startingEntry" type="OCTET_STRING"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="rangeStopSpecification" minOccurs="0">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element name="endingTime" type="TimeOfDay"/>
          <xsd:element name="numberOfEntries" type="Integer32"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="listOfVariables" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="VisibleString" type="VisibleString"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="entryToStartAfter" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="timeSpecification" type="TimeOfDay"/>
          <xsd:element name="entrySpecification" type="OCTET_STRING"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<!-- Annex B -File Access Protocol of ISO/IEC 9506-2:1990 -->
<!-- ObtainFile from Clause B.2 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="ObtainFile-Request">
  <xsd:sequence>
    <!--not used in IEC 61850 -->
    <xsd:element name="sourceFileServer" type="ApplicationReference" minOccurs="0"/>
    <xsd:element name="sourceFile" type="FileName"/>
    <xsd:element name="destinationFile" type="FileName"/>
  </xsd:sequence>
</xsd:complexType>
<!-- Clause 10 – Domain Management Protocol of ISO/IEC 9506-2:1990 -->
<!-- GetDomainAttribute from Clause 10.13 of ISO/IEC 9506-2:1990 not used for IEC 61850, but recommended -->
<!-- Domain Name -->
<xsd:simpleType name="GetDomainAttributes-Request">
  <xsd:restriction base="Identifier"/>
</xsd:simpleType>
<!-- DeleteNamedVariableList from Clause 12.16 of ISO/IEC 9506-2:1990 -->
<!-- Type IMPLICIT INTEGER of scopeOfDelete has not been replaced by CHOICE of IMPLICIT NULL, so that scopeOfDelete can be
encoded as an XML attribute with a default value -->
<xsd:complexType name="DeleteNamedVariableList-Request">
  <xsd:sequence>
    <xsd:element name="listOfVariableListName" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="ObjectName" type="ObjectName" form="qualified"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <!-- not used in IEC 61850 -->
    <xsd:element name="domainName" type="Identifier" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="scopeOfDelete" type="MMS_INTEGER" default="0"/>
</xsd:complexType>
<xsd:simpleType name="MMS_INTEGER">
  <xsd:union memberTypes="xsd:integer">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="specific"/>
        <xsd:enumeration value="aa-specific"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>

```

```

        <xsd:enumeration value="domain"/>
        <xsd:enumeration value="vmd"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:union>
</xsd:simpleType>
<!-- GetNamedVariableListAttributes from Clause 12.15 of ISO/IEC 9506-2:1990 -->
<!-- VariableListName -->
<xsd:complexType name="GetNamedVariableListAttributes-Request">
    <xsd:choice>
        <xsd:element name="vmd-specific" type="Identifier"/>
        <xsd:element name="domain-specific">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="domainId" type="Identifier"/>
                    <xsd:element name="itemId" type="Identifier"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="aa-specific" type="Identifier"/>
    </xsd:choice>
</xsd:complexType>
<!-- DefineNamedVariableList from Clause 12.14 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="DefineNamedVariableList-Request">
    <xsd:sequence>
        <xsd:element name="variableListName" type="ObjectName"/>
        <xsd:element name="listOfVariable">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="SEQUENCE">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="variableSpecification" type="VariableSpecification"/>
                                <xsd:element name="alternateAccess" type="AlternateAccess" minOccurs="0"/>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<!-- GetVariableAccessAttributes from Clause 12.9 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="GetVariableAccessAttributes-Request">
    <xsd:choice>
        <xsd:element name="name" type="ObjectName"/>
        <!-- out of scope of IEC 61850 -->
        <xsd:element name="address" type="Address"/>
    </xsd:choice>
</xsd:complexType>
<!-- Write from Clause 12.7 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Write-Request">
    <xsd:sequence>
        <xsd:element name="variableAccessSpecification" type="VariableAccessSpecification"/>
        <xsd:element name="listOfData">
            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                    <xsd:element name="Data" type="Data" form="qualified"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<!-- Read from Clause 12.6 of ISO/IEC 9506-2:1990 -->
<xsd:complexType name="Read-Request">
    <xsd:sequence>
        <xsd:element name="variableAccessSpecification" type="VariableAccessSpecification"/>
    </xsd:sequence>
    <xsd:attribute name="specificationWithResult" type="MMS_BOOLEAN_7" default="false"/>
</xsd:complexType>
<xsd:simpleType name="MMS_BOOLEAN_7">
    <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
<!-- Identify from Clause 9.5 of ISO/IEC 9506-2:1990 -->
<xsd:simpleType name="Identify-Request">
    <xsd:restriction base="NULL"/>
</xsd:simpleType>

```

```

<!-- Clause 9 – VMD Support Protocol of ISO/IEC 9506-2:1990 -->
<!-- Status from Clause 9.2 of ISO/IEC 9506-2:1990 -->
<xsd:simpleType name="Status-Request">
  <xsd:restriction base="BOOLEAN_7"/>
</xsd:simpleType>
<xsd:simpleType name="BOOLEAN_7">
  <xsd:restriction base="xsd:boolean"/>
</xsd:simpleType>
</xsd:schema>
<CODE ENDS>

```

G.4 Extension of IEC 62351-4 when E2E security is turned off

This clause shall be used only for specific use cases when an appropriate security study has shown that the end-to-end security mechanism is not required. In that case, the applicative payload shall be embedded directly within the association context part.

Therefore, the XSDs defined in IEC 62351-4 for the XMPP operational environment shall be extended as follows:

- Within the definition of the association context element for both kinds of stanza, iq and message, add the element `<xsd:element ref="ns1:MMSpdu"/>` as a possible choice in addition to the choices already defined by IEC 62351-4 for the end-to-end security handshakes, errors and data transfers.
- Add an XML import directive for the XML namespace defined by the present document (http://www.iec.ch/61850/2018/SCSM_8_2) and assign a prefix to it within the schema element: e.g. `xmlns:ns1="http://www.iec.ch/61850/2018/SCSM_8_2"`.

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

Bibliography

IEC 61850-7-1, *Communication networks and systems for power utility automation – Part 7-1: Basic communication structure – Principles and models*

IEC 61850-7-410, *Communication networks and systems for power utility automation – Part 7-410: Basic communication structure – Hydroelectric power plants – Communication for monitoring and control*

IEC 61850-7-420, *Communication networks and systems for power utility automation – Part 7-420: Basic communication structure – Distributed energy resources logical nodes*

ISO/IEC 8825-1:2015 [ITU-T X.690:2015], *Information technology – ASN.1 encoding rules: Specification of XML Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

SOMMAIRE

AVANT-PROPOS	265
INTRODUCTION.....	267
1 Domaine d'application	268
1.1 Généralités	268
1.2 Nom et version d'espace de nommage.....	268
1.3 Répartition des éléments de code	269
2 Références normatives	269
3 Termes et définitions	272
4 Abréviations	273
5 Vue d'ensemble	274
5.1 Généralités	274
5.2 Mapping des services client/serveur	275
5.2.1 Généralités	275
5.2.2 Données utiles XML.....	277
5.2.3 Accords de mise en œuvre	278
5.2.4 XMPP	278
5.3 Services de synchronisation temporelle	279
6 Utilisation de XMPP.....	279
6.1 Principes.....	279
6.2 Établissement d'une connexion	279
6.2.1 Généralités	279
6.2.2 Utilisation de TLS et de SASL.....	279
6.2.3 Compression de flux	280
6.3 Mapping des services ACSI	280
6.4 Utilisation de la présence XMPP	281
6.5 Utilisation de la liste.....	281
6.6 Extensions XMPP.....	281
6.6.1 Utilisation de XMPP PING – XEP 0199	281
6.6.2 Utilisation de la gestion de flux – XEP 0198.....	281
6.7 Accords de mise en œuvre – XMPP PING – XEP 0199	282
7 Sécurité de bout en bout.....	282
8 Description des données utiles	282
8.1 Aperçu XSD	282
8.2 Objets de l'IEC 61850	283
8.2.1 Généralités	283
8.2.2 Nœud logique (LN)	283
8.2.3 Mapping des références de nœuds logiques à VariableAccessSpecifications	286
8.2.4 Mapping d'une référence DataObjects avec VariableAccessSpecifications	287
8.2.5 Mapping de la référence DataAttributes (DataAttr) à VariableAccessSpecifications	288
8.2.6 Utilisation de l'accès alternatif pour les références DataObjects et DataAttributes.....	289
8.3 Mapping des attributs de données de l'IEC 61850-7-2.....	294
8.3.1 BasicTypes.....	294
8.3.2 Définitions additionnelles de BasicType	295

8.3.3	ACSITypes communs	297
8.3.4	Mapping de type d'attribut de données de qualité communes spécifié dans l'IEC 61850-7-2	309
8.4	Mapping général des valeurs de données dans des données utiles XML	310
8.5	Comportement étendu pour l'optimisation de la bande passante	311
9	Modèle de la classe Server	311
9.1	Généralités	311
9.2	GetServerDirectory	312
9.2.1	Généralités	312
10	Modèle d'association	315
10.1	Relation d'association avec les profils de communication	315
10.2	Modèle d'association bipartite pour le profil de communication client/serveur	316
10.2.1	Établissement d'une association de bout en bout sécurisée	316
10.2.2	Services d'association	316
11	Modèle de dispositif logique	321
11.1	Généralités	321
11.2	Response-	323
11.3	Comportement étendu	323
12	Modèle de nœud logique	325
12.1	Généralités	325
12.2	GetLogicalNodeDirectory	325
12.2.1	Généralités	325
12.2.2	Response-	327
12.2.3	Comportement étendu	328
12.3	GetAllDataValues	330
12.3.1	Généralités	330
13	Modèle DataObject, DataAttribute, SubDataAttribute	333
13.1	Généralités	333
13.2	GetDataValues	333
13.3	SetDataValues	335
13.4	GetDataDirectory	337
13.4.1	Généralités	337
13.4.2	Response-	338
13.4.3	Comportement étendu	339
13.5	GetDataDefinition	341
14	Modèle de classe d'ensemble de données	341
14.1	Généralités	341
14.2	GetDataSetValues	342
14.2.1	Généralités	342
14.2.2	Response-	344
14.3	DataSetValues	345
14.3.1	Généralités	345
14.3.2	Response-	346
14.4	CreateDataSet	347
14.4.1	Généralités	347
14.4.2	Response-	349
14.5	DeleteDataSet	350
14.5.1	Généralités	350

14.5.2	Erreurs	352
14.6	GetDataSetDirectory	353
14.6.1	Généralités	353
14.6.2	Response-	354
15	Modèle ServiceTracking	355
16	Modèle de classe de commande de groupe de réglage	358
16.1	Définition de bloc de commande de groupe de réglage (SGCB)	358
16.2	Services de classe de commande de groupe de réglage	358
16.2.1	SelectActiveSG	358
16.2.2	SelectEditSG	360
16.2.3	SetEditSGValue	360
16.2.4	ConfirmEditSGValues	361
16.2.5	GetEditSGValue	362
16.2.6	GetSGCBValues	363
17	Modèle de classe de reporting et de journalisation	364
17.1	Modèle de rapport – Blocs de commande de rapport	364
17.1.1	Bloc de commande de rapport mis en mémoire tampon	364
17.1.2	Bloc de commande de rapport non mis en mémoire tampon	366
17.2	Services de reporting	367
17.2.1	Service de rapport	367
17.2.2	GetBRCBValues	371
17.2.3	SetBRCBValues	374
17.2.4	GetURCBValues	377
17.2.5	SetURCBValues	377
17.3	Modèle Log (journal)	377
17.3.1	Vue d'ensemble	377
17.3.2	Description des attributs LCB	379
17.3.3	Mapping des services de journal et de commande de journal	380
17.3.4	Conformité	385
18	Mapping du modèle d'événement de poste générique (GSE) – Événement de poste orienté objet générique (GOOSE)	386
18.1	Définition de commande GOOSE	386
18.2	Spécialisation pour le GoCB Couche 2	386
18.3	Spécialisation pour le GOOSE acheminable	387
18.4	Services GOOSE	387
18.4.1	Généralités	387
18.4.2	GetGoCBValues	387
18.4.3	SetGoCBValues	387
19	Transmission de modèle de classe de valeurs échantillonnées	387
19.1	Bloc de commande de valeur échantillonnée	387
19.1.1	Généralités	387
19.1.2	Spécialisation pour la valeur échantillonnée Couche 2	387
19.1.3	Spécialisation pour la valeur échantillonnée acheminable	387
19.1.4	Spécialisation pour la valeur échantillonnée envoi individuel	388
19.2	Services de valeur échantillonnée	388
19.2.1	Généralités	388
19.2.2	GetMSVCBValues	388
19.2.3	SetMSVCBValues	388

20	Modèle de classe de commande	388
20.1	Généralités	388
20.2	Vue d'ensemble du mapping des services de commande	389
20.3	Select	390
20.4	SelectWithValue	392
20.4.1	Mapping des paramètres de service SelectWithValue	392
20.4.2	Mapping général du service SelectWithValue	392
20.4.3	Response– de SelectWithValue	395
20.5	Cancel	395
20.5.1	Mapping des paramètres de service Cancel	395
20.5.2	Mapping général du service Cancel	396
20.5.3	Cancel response–	398
20.6	Operate	398
20.6.1	Mapping du paramètre de service Operate	398
20.6.2	Mapping général du service Operate	398
20.6.3	Operate response–	400
20.6.4	Mapping des paramètres de service CommandTermination	400
20.6.5	Mapping général du service CommandTermination	400
20.7	TimeActivatedOperate	404
20.7.1	Mapping des paramètres de service TimeActivatedOperate	404
20.7.2	Mapping du service TimeActivatedOperate	404
20.8	Service TimeActivatedOperateTermination	404
20.9	AdditionalCauseDiagnosis dans les réponses de service de commande négatives	405
20.10	Mapping des services de commande	408
20.10.1	Généralités	408
20.10.2	Mapping du Control Service Tracking (CTS)	408
21	Modèle de temps et de synchronisation temporelle	409
22	Conventions de dénomination	409
23	Transfert de fichier	409
23.1	Modèle de transfert de fichier	409
23.2	Services de fichier	411
23.2.1	GetFile	411
23.2.2	SetFile	417
23.2.3	DeleteFile	419
23.2.4	GetFileAttributeValues	421
24	Conformité	422
24.1	Notation	422
24.2	PICS	422
24.2.1	Conformité de profil	422
24.2.2	Conformité des données utiles XML	424
24.3	Déclaration PICS	445
24.3.1	Généralités	445
24.3.2	Langage de configuration de poste	445
25	Langage de configuration de poste (SCL)	445
Annexe A (normative)	Pile de communication	446
A.1	Vue d'ensemble	446
A.1.1	Généralités	446

A.1.2	Profils de communication XMPP	447
A.1.3	Profils de communication non XMPP	447
A.2	Pile de communication	448
A.2.1	Présentation de l'utilisation du protocole.....	448
A.2.2	Services client/serveur et profils de communication	448
A.2.3	Time sync (synchronisation temporelle)	450
Annexe B (informative)	Déploiement de l'infrastructure XMPP	452
B.1	Généralités	452
B.2	Déploiement de XMPP dans un domaine XMPP	452
B.2.1	Cas d'utilisation Installation	452
B.2.2	Hiérarchie de cas d'utilisation à l'intérieur d'une installation.....	453
B.3	Déploiement de XMPP lors de l'interconnexion de plusieurs domaines XMPP	454
B.3.1	Interconnexion du domaine XMPP	454
B.3.2	Définition d'une communication de fédération entre des domaines XMPP	462
B.3.3	Interconnexion du domaine avec la fédération	463
B.4	Indisponibilité et récupération de chemin de communication	469
Annexe C (informative)	Sécurité pour l'intégration DER reposant sur XMPP	470
C.1	Généralités	470
C.2	Hypothèses et conditions aux limites	471
C.3	Écart en matière d'exigences de sécurité	472
C.4	Mapping des options de sécurité vers l'intégration basée sur XMPP des DER.....	473
C.5	Diagrammes de séquence.....	473
C.5.1	Généralités	473
C.5.2	XMPP et ouverture de flux	474
C.5.3	Établissement de flux, ROSTER et Présence	475
C.5.4	Indisponibilité de communication	476
C.5.5	Demande Réponse (clearTransfer)	479
Annexe D (normative)	Mapping des services et des erreurs sur les strophes XMPP	481
Annexe E (informative)	Écarts volontaires par rapport au SCSM IEC 61850-8-1	484
Annexe F (informative)	Conformité SCL	485
Annexe G (normative)	Définitions de schéma XML pour les données utiles XML.....	486
G.1	Généralités	486
G.2	Schéma XML de l'API virtuelle de l'IEC 61850-8-2	486
G.3	Schéma XML des données utiles applicatives pour l'IEC 61850-8-2	486
G.4	Extension de l'IEC 62351-4 lorsque la sécurité E2E est désactivée.....	513
Bibliographie.....		514
Figure 1 – Vue d'ensemble des fonctionnalités et des profils		275
Figure 2 – Exemple de données utiles XML		278
Figure 3 – Structure générique des services ACSI client/serveur		280
Figure 4 – Algorithme pour le mapping de nœud logique.....		283
Figure 5 – Liste ordonnée des contraintes fonctionnelles		284
Figure 6 – Exemple de description de type de nœud logique.....		285
Figure 7 – Liste des variables nommées aplanies correspondant à un nœud logique		286
Figure 8 – Mapping XML d'une LNReference avec un accès direct		287
Figure 9 – Mapping XML d'une LNReference avec un accès alternatif		287

Figure 10 – Mapping XML direct d'un FCD	288
Figure 11 – Mapping XML direct d'un FCDA.....	288
Figure 12 – Accès alternatif sans élément d'ensemble	290
Figure 13 – Accès alternatif avec élément d'ensemble	292
Figure 14 – Accès alternatif avec variable aplanie et élément d'ensemble.....	293
Figure 15 – Structure XML de GetServerDirectory-Request (LD).....	313
Figure 16 – Structure XML de GetServerDirectory-Response (LD)	313
Figure 17 – Structure XML de GetServerDirectory-Request (FICHIER)	315
Figure 18 – Structure XML de GetServerDirectory-Response (FICHIER).....	315
Figure 19 – Structure XML d'Associate-Request	318
Figure 20 – Structure XML d'Associate-Response.....	318
Figure 21 – Structure XML de GetLogicalDeviceDirectory-Request	322
Figure 22 – Structure XML de GetLogicalDeviceDirectory-Response	323
Figure 23 – Structure XML de GetLogicalDeviceDirectory-Request étendu	324
Figure 24 – Structure XML de GetLogicalDeviceDirectory-Response étendu	325
Figure 25 – Structure XML de GetLogicalNodeDirectory-Request	327
Figure 26 – Structure XML de GetLogicalNodeDirectory-Response.....	327
Figure 27 – Structure XML de GetLogicalNodeDirectory-Request étendu (étape 1).....	329
Figure 28 – Structure XML de GetLogicalNodeDirectory-Response étendu (étape 1)	329
Figure 29 – Structure XML de GetLogicalNodeDirectory-Request étendu (étape 2).....	330
Figure 30 – Structure XML de GetLogicalNodeDirectory-Response étendu (étape 2)	330
Figure 31 – Structure XML de GetAllDataValues-Request.....	332
Figure 32 – Structure XML de GetAllDataValues-Response	333
Figure 33 – Structure XML de GetDataValues-Request.....	335
Figure 34 – Structure XML de GetDataValues-Response	335
Figure 35 – Structure XML de SetDataValues-Request	336
Figure 36 – Structure XML de SetDataValues-Response	337
Figure 37 – Structure XML de GetDataDirectory-Request	338
Figure 38 – Structure XML de GetDataDirectory-Response.....	338
Figure 39 – Structure XML de GetDataDirectory-Request étendu.....	340
Figure 40 – Structure XML de GetDataDirectory-Response étendu	341
Figure 41 – Mapping de la référence à un ensemble de données persistant dans un dispositif logique	342
Figure 42 – Mapping de la référence à un ensemble de données persistant à l'extérieur d'un dispositif logique	342
Figure 43 – Mapping de la référence à un ensemble de données non persistant	342
Figure 44 – Structure XML de GetDataSetValues-Request.....	343
Figure 45 – Structure XML de GetDataSetValues-Response	344
Figure 46 – Structure XML de SetDataSetValues-Request.....	346
Figure 47 – Structure XML de SetDataSetValues-Response	346
Figure 48 – Structure XML de CreateDataSet-Request	349
Figure 49 – Structure XML de CreateDataSet-Response.....	349
Figure 50 – Structure XML de DeleteDataSet-Request.....	351
Figure 51 – Structure XML de DeleteDataSet-Response	352

Figure 52 – Structure XML de GetDataSetDirectory-Request	353
Figure 53 – Structure XML de GetDataSetDirectory-Response	354
Figure 54 – Structure XML de SelectActiveSG-Request	359
Figure 55 – Structure XML de SelectActiveSG-Response+	359
Figure 56 – Structure XML de SelectActiveSG-Response-	359
Figure 57 – Structure XML de SelectEditSG-Request	360
Figure 58 – Structure XML de SetEditSGValue-Request	361
Figure 59 – Structure XML de ConfirmEditSGValues	362
Figure 60 – Structure XML de GetEditSGValue-Request	362
Figure 61 – Structure XML de GetEditSGValue-Response	363
Figure 62 – Structure XML de GetSGCBValues-Request	363
Figure 63 – Structure XML de GetSGCBValues-Response	364
Figure 64 – Structure XML de Report	370
Figure 65 – Structure XML de GetBRCBValues-Request	372
Figure 66 – Structure XML de GetBRCBValues-Response	374
Figure 67 – Structure XML de SetBRCBValues-Request	376
Figure 68 – Structure XML de SetBRCBValues-Response	377
Figure 69 – Relation des attributs LCB avec les définitions de journal (log) de l'IEC 61850-7-2	378
Figure 70 – Structure XML de QueryLogByTime-Request	381
Figure 71 – Structure XML de QueryLogByTime-Response	383
Figure 72 – Structure XML de QueryLogAfter-Request	385
Figure 73 – Structure XML de Select-Request	391
Figure 74 – Structure XML de Select-Response	392
Figure 75 – Structure XML de SelectWithValue-Request	394
Figure 76 – Structure XML de SelectWithValue-Response+	395
Figure 77 – Structure XML de SelectWithValue-Response-	395
Figure 78 – Structure XML de la Request+ de CommandTermination	402
Figure 79 – Structure XML de la Request- de CommandTermination	404
Figure 80 – Structure XML d'InformationReport avec AdditionalCauseDiagnosis	406
Figure 81 – Mapping de GetFile ACSI à FileOpen, FileRead, FileClose	412
Figure 82 – Exemple XML de FileOpen Request	414
Figure 83 – Exemple XML de FileOpen Response+	414
Figure 84 – Exemple XML de FileRead Request (premier)	414
Figure 85 – Exemple XML de FileRead Response+ (premier)	415
Figure 86 – Exemple XML de FileRead Request (second)	415
Figure 87 – Exemple XML de FileRead Response+ (second)	415
Figure 88 – Exemple XML de FileClose Request	415
Figure 89 – Exemple XML de FileClose Response	415
Figure 90 – Mapping de service SetFile ACSI	417
Figure 91 – Exemple XML d'ObtainFile Request	418
Figure 92 – Exemple XML d'ObtainFile Response	418
Figure 93 – Exemple XML de DeleteFile Request	420

Figure 94 – Exemple XML de DeleteFile Response	420
Figure 95 – Exemple XML de GetFileAttributeValues Request	421
Figure 96 – Exemple XML de GetFileAttributeValues Response	422
Figure 97 – VariableSpecification pour LDevice/MHA11.HA.phsAHar(7).cVal.mag.f	434
Figure 98 – VariableSpecification plus court pour LDevice/MHA11.HA.phsAHar(7).cVal.mag.f	435
Figure 99 – VariableSpecification I non conforme	436
Figure 100 – VariableSpecification II non conforme	437
Figure 101 – VariableSpecification pour LDevice/MHA11.HA.phsAHar(7) [MX]	438
Figure 102 – VariableSpecification plus court pour LDevice/MHA11.HA.phsAHar(7) [MX]	439
Figure A.1 – Vue d'ensemble des fonctionnalités et des profils	447
Figure A.2 – Modèle de référence et profils OSI	448
Figure B.1 – Domaine d'installation	453
Figure B.2 – Agrégation hiérarchique au niveau de l'installation	453
Figure B.3 – Système de gestion DER au niveau de l'installation	454
Figure B.4 – Intégration de la gestion d'installation au niveau du DSO	455
Figure B.5 – Plusieurs installations au niveau du DSO	456
Figure B.6 – VPP et DER contractés	457
Figure B.7 – Commande indirecte utilisant des JID VPP	458
Figure B.8 – Commande directe VPP utilisant des JID VPP	459
Figure B.9 – Commande directe VPP utilisant des JID DSO	460
Figure B.10 – Commande indirecte DSO avec des JID VPP	461
Figure B.11 – Commande directe DSO	462
Figure B.12 – Concept de fédération dans XMPP	463
Figure B.13 – DSO de fédération – Installation	464
Figure B.14 – Utilisation de la fédération avec VPP	465
Figure B.15 – Communication avec des JID VPP – commande indirecte	466
Figure B.16 – Communication VPP avec des JID VPP – commande directe	467
Figure B.17 – Communication VPP avec des JID DSO	468
Figure B.18 – Communication DSO avec des JID VPP	469
Figure C.1 – Système de base pour la présentation des exigences de sécurité IT	471
Figure C.2 – Établissement de flux XMPP – Serveur IEC 61850 au serveur XMPP	475
Figure C.3 – Établissement de flux XMPP – Client IEC 61850 au serveur XMPP	475
Figure C.4 – Établissement de flux, liste et présence	476
Figure C.5 – Indisponibilité de communication – Perte de liaison	477
Figure C.6 – Indisponibilité de communication – Présence indisponible	478
Figure C.7 – Demande Réponse	479
Figure C.8 – Demande – Abandon	480
 Tableau 1 – Services exigeant un profil de communication client/serveur	 276
Tableau 2 – Mapping des classes ACSI aux concepts MMS	283
Tableau 3 – Mapping des BasicTypes ACSI	294
Tableau 4 – Structure PhyComAddr pour la communication Couche 2	299

Tableau 5 – PhyComAddr pour la communication UDP/IP	299
Tableau 6 – Conflit de GetNameList pour un objectClass et un objectScope IEC 61850.....	301
Tableau 7 – Mappings d'erreur de service pour les services ACSI utilisant GetNameList.....	301
Tableau 8 – Mappings d'erreurs de service de lecture.....	302
Tableau 9 – Mappings des erreurs de service d'écriture.....	304
Tableau 10 – Mappings des erreurs de service GetFileAttributeValues	305
Tableau 11 – Codage de TimeQuality selon l'IEC 61850-7-2.....	306
Tableau 12 – Codage de TriggerConditions	307
Tableau 13 – Codage de ReasonForInclusionInReport.....	307
Tableau 14 – Codage de ReasonForInclusionInLog	308
Tableau 15 – Codage de RCBReportOptions	308
Tableau 16 – Codage de SVMessageOptions	308
Tableau 17 – Codage de CheckConditions.....	309
Tableau 18 – Codage de quality (qualité) de l'IEC 61850-7-2.....	310
Tableau 19 – Exemples de codage de valeurs de données	311
Tableau 20 – Mapping de GetServerDirectory (DISPOSITIF LOGIQUE) ACSI.....	312
Tableau 21 – Mapping de GetServerDirectory (FICHIER) ACSI.....	314
Tableau 22 – Modèle d'association/profils de communication.....	315
Tableau 23 – Mapping du service Associate ACSI	317
Tableau 24 – Description des éléments de demande Associate	317
Tableau 25 – Description des éléments de réponse Associate	317
Tableau 26 – Mappings des erreurs de service ACSI d'association	319
Tableau 27 – Mapping du service Release ACSI.....	321
Tableau 28 – Mappings des erreurs de service Release	321
Tableau 29 – Mapping de GetLogicalDeviceDirectory ACSI	322
Tableau 30 – Mapping étendu de GetLogicalDeviceDirectory ACSI.....	324
Tableau 31 – Classes d'objets pour le service GetLogicalNodeDirectory.....	325
Tableau 32 – Mapping de GetLogicalNodeDirectory ACSI.....	326
Tableau 33 – Mapping étendu de GetLogicalNodeDirectory ACSI	328
Tableau 34 – Mapping de GetAllDataValues ACSI	331
Tableau 35 – Mapping des paramètres de service GetDataValues	334
Tableau 36 – Mapping des paramètres de service SetDataValues	336
Tableau 37 – Mapping des paramètres de service GetDataDirectory.....	337
Tableau 38 – Mappings des erreurs de service GetDataDirectory	339
Tableau 39 – Mapping étendu de GetDataDirectory ACSI	340
Tableau 40 – Mapping des paramètres de service GetDataSetValues.....	342
Tableau 41 – Mappings des erreurs GetDataSetValues	345
Tableau 42 – Mapping des paramètres de service SetDataSetValues	345
Tableau 43 – Mappings des erreurs SetDataSetValues.....	347
Tableau 44 – Mapping des paramètres de service CreateDataSet	348
Tableau 45 – Mappings des erreurs de service CreateDataSet	350
Tableau 46 – Mapping des paramètres de service DeleteDataSet.....	351

Tableau 47 – Mappings des erreurs de service DeleteDataSet.....	352
Tableau 48 – Mapping des paramètres de service GetDataSetDirectory	353
Tableau 49 – Mappings des erreurs de service GetDataSetDirectory	355
Tableau 50 – Mapping des valeurs de ServiceType ACSI	355
Tableau 51 – Mapping des valeurs d'errorCode ACSI	357
Tableau 52 – Mapping de CDC LTS	357
Tableau 53 – Mapping de CDC GTS	358
Tableau 54 – Mapping de SGCb	358
Tableau 55 – Structure BRCb	365
Tableau 56 – Structure URcb	366
Tableau 57 – Ordre des AccessResults pour Report	368
Tableau 58 – Mapping des paramètres de service GetBRCbValues	371
Tableau 59 – Mapping des paramètres de service SetBRCbValues	375
Tableau 60 – Structure LCB	379
Tableau 61 – Mapping des paramètres de service QueryLogByTime	381
Tableau 62 – Mappings de ServiceError pour les services Log	384
Tableau 63 – Mapping des paramètres de QueryLogAfter-Request	385
Tableau 64 – Exigences de conformité de journal	385
Tableau 65 – Définition de TypeDescription pour la structure de GoCB	386
Tableau 66 – Paramètres de service contrôlables	389
Tableau 67 – Mapping du modèle de commande de l'IEC 61850-7-2 aux composants de commande	389
Tableau 68 – Mapping des services de commande	390
Tableau 69 – Mapping des paramètres Select	391
Tableau 70 – Mapping des paramètres de service SelectWithValue	392
Tableau 71 – Mapping des paramètres SelectWithValue	393
Tableau 72 – Spécification de SelectWithValue, d'Oper et de Cancel DataAccessError	395
Tableau 73 – Mapping des paramètres de service Cancel	396
Tableau 74 – Mapping du service Cancel	397
Tableau 75 – Mapping du paramètre de service Operate	398
Tableau 76 – Mapping du service Operate	399
Tableau 77 – Mapping du service CommandTermination	401
Tableau 78 – Définition de la structure de variable LastApplError	405
Tableau 79 – Mapping des valeurs AddCause ACSI	408
Tableau 80 – Mapping de la CDC CTS	409
Tableau 81 – Mapping de la classe fichier ACSI à l'objet fichier MMS	409
Tableau 82 – Suffixes de fichier réservés	410
Tableau 83 – Mapping du service GetFile ACSI	413
Tableau 84 – Mappings des erreurs de service GetFile	414
Tableau 85 – Mappings des ServiceError ACSI aux erreurs de service FileOpen	416
Tableau 86 – Mappings des ServiceError ACSI aux erreurs de service FileRead	416
Tableau 87 – Mappings des ServiceError ACSI aux erreurs de service FileClose	417
Tableau 88 – Mapping des paramètres SetFile ACSI	418

Tableau 89 – Mappings des ServiceError ACSI aux erreurs de service ObtainFile	419
Tableau 90 – Mapping du service DeleteFile ACSI.....	419
Tableau 91 – Mappings des ServiceError ACSI aux erreurs de service DeleteFile	420
Tableau 92 – Mapping des paramètres GetFileAttributeValues ACSI	421
Tableau 93 – PICS pour la prise en charge de profil A.....	423
Tableau 94 – PICS pour la prise en charge de profil A TimeSync.....	423
Tableau 95 – PICS pour la prise en charge du profil T	424
Tableau 96 – Paramètres généraux d'InitiateRequest MMS	424
Tableau 97 – Paramètres généraux d'InitiateResponse MMS	425
Tableau 98 – Table de conformité de service MMS pris en charge	425
Tableau 99 – Paramètre MMS CBB.....	428
Tableau 100 – Déclaration de conformité pour GetNameList.....	429
Tableau 101 – Déclaration de conformité pour GetCapabilityList.....	429
Tableau 102 – Déclaration de conformité pour GetDomainAttributes.....	430
Tableau 103 – Déclaration de conformité pour Status	430
Tableau 104 – Déclaration de conformité pour Cancel	431
Tableau 105 – Déclaration de conformité pour Identify.....	431
Tableau 106 – Déclaration de conformité pour AlternateAccess	432
Tableau 107 – Déclaration de conformité pour AlternateAccessSelection.....	432
Tableau 108 – Déclaration de conformité pour VariableAccessSpecification	439
Tableau 109 – Déclaration de conformité pour VariableSpecification	440
Tableau 110 – Déclaration de conformité pour le service Read	440
Tableau 111 – Déclaration de conformité pour Write	440
Tableau 112 – Déclaration de conformité pour InformationReport	441
Tableau 113 – Déclaration de conformité pour GetVariableAccessAttributes	441
Tableau 114 – Déclaration de conformité pour DefineNamedVariableList.....	441
Tableau 115 – Déclaration de conformité pour GetNamedVariableListAttributes	442
Tableau 116 – Déclaration de conformité pour DeleteNamedVariableList.....	442
Tableau 117 – Déclaration de conformité pour ReadJournal	442
Tableau 118 – Déclaration de conformité pour EntryContent.....	443
Tableau 119 – Déclaration de conformité pour FileDirectory	444
Tableau 120 – Déclaration de conformité pour FileOpen	444
Tableau 121 – Déclaration de conformité pour FileRead	444
Tableau 122 – Déclaration de conformité pour FileClose	445
Tableau 123 – Définitions de type P autorisé pour l'adressage client/serveur.....	445
Tableau A.1 – Services et protocoles pour le profil A de communication client/serveur	449
Tableau A.2 – Services et protocoles pour le profil T XMPP client/serveur	450
Tableau A.3 – Profil A de synchronisation temporelle.....	450
Tableau A.4 – Profil T de synchronisation temporelle.....	451
Tableau D.1 – Mappings de services ACSI sur des strophes XMPP	481
Tableau D.2 – Mappings d'erreurs sur des strophes XMPP	483
Tableau F.1 – Degrés de conformité SCL.....	485
Tableau F.2 – Services ACSI pris en charge pour SCL.2	485

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

**RÉSEAUX ET SYSTÈMES DE COMMUNICATION POUR
L'AUTOMATISATION DES SYSTÈMES ÉLECTRIQUES –****Partie 8-2: Mapping des services de communication spécifiques (SCSM) –
Mapping avec le protocole XMPP
(Extensible Messaging Presence Protocol)**

AVANT-PROPOS

- 1) La Commission Électrotechnique Internationale (IEC) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de l'IEC). L'IEC a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. À cet effet, l'IEC – 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 l'IEC"). 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 l'IEC, participent également aux travaux. L'IEC 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 l'IEC concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de l'IEC intéressés sont représentés dans chaque comité d'études.
- 3) Les Publications de l'IEC se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de l'IEC. Tous les efforts raisonnables sont entrepris afin que l'IEC s'assure de l'exactitude du contenu technique de ses publications; l'IEC 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 l'IEC s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de l'IEC dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de l'IEC et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
- 5) L'IEC 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 l'IEC. L'IEC 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 l'IEC, 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 l'IEC, 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 l'IEC ou de toute autre Publication de l'IEC, 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 l'IEC peuvent faire l'objet de droits de brevet. L'IEC 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 IEC 61850-8-2 a été établie par le comité d'études 57 de l'IEC: Gestion des systèmes de puissance et échanges d'informations associés.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
57/2020/FDIS	57/2039/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 internationale.

Le présent document a été rédigé selon les Directives ISO/IEC, Partie 2.

La présente norme IEC inclut les éléments de code, c'est-à-dire les éléments destinés à être traités directement par un ordinateur. Ce type de contenu est un texte placé entre les marqueurs <CODE BEGINS> et <CODE ENDS> ou est clairement étiqueté dans la présente norme en tant qu'élément de code.

L'achat de la présente norme IEC fait l'objet d'une licence de droits d'auteur permettant à l'acheteur de vendre le logiciel contenant des composants de code issus de la présente norme à des utilisateurs, soit directement soit par l'intermédiaire de distributeurs, soumis aux conditions de licence logicielle IEC, qui peuvent être consultées à l'adresse ci-dessous: www.iec.ch/CCv1.

Une liste de toutes les parties de la série IEC 61850, publiées sous le titre général *Réseaux et systèmes de communication pour l'automatisation des systèmes électriques*, peut être consultée sur le site web de l'IEC.

Le comité a décidé que le contenu de ce document ne sera pas modifié avant la date de stabilité indiquée sur le site web de l'IEC sous "<http://webstore.iec.ch>" dans les données relatives à la publication recherchée. À cette date, le document sera

- reconduit,
- supprimé,
- remplacé par une édition révisée, ou
- amendé.

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

La présente partie de l'IEC 61850 fait partie d'un ensemble de spécifications qui détaille une architecture de communication de service en couches.

La norme de communication IEC 61850 est largement utilisée dans tous les domaines liés au réseau intelligent, poussant à utiliser des technologies adaptées à la connexion d'un très grand nombre d'applications et d'appareils sur Intranet et Internet. Les domaines concernés utilisent déjà des protocoles bien établis, en général destinés à l'échange de données avec les applications au niveau IT (la planification des ressources, la gestion des actifs et de la maintenance, etc.). Il devient donc impératif de proposer une stratégie d'intégration de l'IEC 61850 dans ces différents protocoles et informations disparates.

Dans ce contexte, la présente partie de l'IEC 61850 décrit un mapping de service de communication spécifique (SCSM) sur le protocole XMPP (eXtensible Messaging and Presence Protocol), en donnant des informations détaillées sur la création et l'échange de messages de communication concrets qui mettent en œuvre des services abstraits et les modèles spécifiés dans l'IEC 61850-7-4, l'IEC 61850-7-3 et l'IEC 61850-7-2.

Ce mapping est destiné à être utilisé entre tous les types d'appareils de ressources énergétiques réparties et leurs systèmes de gestion connexes, en particulier sur les réseaux publics.

NOTE La présente partie de l'IEC 61850 ne contient pas de support de formation. À cet effet, l'IEC 61850-5 et l'IEC 61850-7-1 peuvent être lues conjointement avec l'IEC 61850-7-2.

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

RÉSEAUX ET SYSTÈMES DE COMMUNICATION POUR L'AUTOMATISATION DES SYSTÈMES ÉLECTRIQUES –

Partie 8-2: Mapping des services de communication spécifiques (SCSM) – Mapping avec le protocole XMPP (Extensible Messaging Presence Protocol)

1 Domaine d'application

1.1 Généralités

La présente partie de l'IEC 61850 spécifie une méthode d'échange de données par tous types de réseaux, y compris les réseaux publics. Parmi les différents types de services spécifiés dans l'IEC 61850-7-2, seuls les services client/serveur et les services de synchronisation temporelle sont pris en considération.

NOTE Les services client/serveur des modèles GOOSE et SMV sont également mappés (voir le Tableau 1).

Pour les services client/serveur, le principe consiste à mapper les objets et les services de l'interface abstraite des services de communication (ACSI – Abstract Communication Service Interface) définie dans l'IEC 61850-7-2) aux messages XML transportés sur XMPP. La description du mapping inclut trois principaux aspects:

- L'utilisation du protocole XMPP lui-même, décrivant en détail les fonctions réellement utilisées et la manière dont le mapping les utilise (voir l'Article 6).
- La manière d'obtenir des communications sécurisées de bout en bout (voir l'Article 7).
- La description des données utiles XML correspondant à chaque service ACSI au moyen, en particulier, du schéma XML et des exemples de message XML (commençant à l'Article 9).

NOTE 1 Le présent document ne concerne pas l'utilisation détaillée du protocole XMPP.

NOTE 2 Le présent document ne concerne pas les services de gestion du système.

NOTE 3 Pour l'information des personnes familiarisées avec le mapping défini dans l'IEC 61850-8-1, les messages XML définis dans le présent document sont déduits de ceux de l'IEC 61850-8-1, mais avec un codage XML en lieu et place du codage binaire. De cette manière, la mise en œuvre de passerelles entre l'IEC 61850-8-1 et l'IEC 61850-8-2 est très directe dans les deux sens. Toutefois, il n'est pas indispensable de lire l'IEC 61850-8-1 pour comprendre le présent document, sauf s'il est utilisé conjointement avec l'un des mappings GOOSE décrits dans l'IEC 61850-8-1.

1.2 Nom et version d'espace de nommage

Cette nouvelle section est obligatoire pour tous les espaces de nommage IEC 61850 (tels que définis par l'IEC 61850-7-1).

Les paramètres qui définissent cette version de l'espace de nommage SCSM_8_2 `xmlns="http://www.iec.ch/61850/2018/SCSM_8_2"` sont les suivants:

- Version de l'espace de nommage: 2018
- Révision de l'espace de nommage: A
- Diffusion de l'espace de nommage: 1
- Date de diffusion de l'espace de nommage: 2018-12

Édition	Date de publication	Webstore	Espace de nommage
Édition 1.0	2018-12	IEC 61850-8-2:2018	IEC 61850-8-2:2018

1.3 Répartition des éléments de code

Les éléments de code inclus dans la présente norme IEC sont également disponibles sous forme de fichier lisible par une machine électronique à l'adresse suivante:

http://www.iec.ch/tc57/supportdocuments/IEC_61850-8-2.2018_ed1.0.XSD.2018A1.full.zip

Les éléments de code inclus dans la présente norme IEC font potentiellement l'objet de travaux de maintenance, et l'utilisateur doit choisir la révision la plus récente dans le référentiel se trouvant à l'adresse suivante: <http://www.iec.ch/tc57/supportdocuments>.

La version/révision la plus récente du document est disponible en sélectionnant le fichier IEC 61850-8-2.2018_ed1.0.XSD.{VersionStateInfo}.full.zip dont la valeur du VersionStateInfo classé est la plus élevée.

En cas de différences entre le code téléchargeable indiqué ci-dessus et le contenu publié en PDF de l'IEC, les codes téléchargeables sont ceux qui sont valides. Ils peuvent faire l'objet de mises à jour. Voir les fichiers historiques.

2 Références normatives

Les documents de référence suivants sont indispensables pour l'application du présent document. 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).

IEC 61850 (toutes les parties), *Réseaux et systèmes de communication pour l'automatisation des systèmes électriques*

IEC TS 61850-2, *Communication networks and systems in substations – Part 2: Glossary* (disponible en anglais seulement)

IEC 61850-5, *Réseaux et systèmes de communication pour l'automatisation des systèmes électriques – Partie 5: Exigences de communication pour les modèles de fonctions et d'appareils*

IEC 61850-6, *Réseaux et systèmes de communication pour l'automatisation des systèmes électriques – Partie 6: Langage pour la description de configuration pour la communication dans les postes électriques, entre les dispositifs électroniques intelligents (IED)*

IEC 61850-7-1, *Réseaux et systèmes de communication pour l'automatisation des systèmes électriques – Partie 7-1: Structure de communication de base – Principes et modèles*

IEC 61850-7-2:2010, *Réseaux et systèmes de communication dans les postes – Partie 7-2: Structure des communications de base pour les postes électriques et les équipements de lignes – Interface abstraite des services de communication (ACSI)*
IEC 61850-7-2:2010/AMD1:2018

IEC 61850-7-3, *Réseaux et systèmes de communication pour l'automatisation des systèmes électriques – Partie 7-3: Structure de communication de base – Classes de données communes*

IEC 61850-7-4, *Réseaux et systèmes de communication pour automatisation des systèmes électriques – Partie 7-4: Structure de communication de base*

IEC 61850-8-1:2011, *Réseaux et systèmes de communication pour l'automatisation des systèmes électriques – Partie 8-1: Mise en correspondance des services de communication spécifiques (SCSM) – Mises en correspondance pour MMS (ISO 9506-1 et ISO 9506-2) et pour l'ISO/CEI 8802-3*
IEC 67850-8-1:2011/AMD1:2018

IEC 62351 (toutes les parties), *Gestion des systèmes de puissance et échanges d'informations associés – Sécurité des communications et des données*

IEC 62351-4:2018, *Power systems management and associated information exchange – Data and communication security – Part 4: Profiles including MMS and derivatives* (disponible en anglais seulement)¹

IEC 62351-6, *Power systems management and associated information exchange – Data and communication security – Part 6: Security for IEC 61850* (disponible en anglais seulement)²

IEC TR 62357-200, *Power systems management and associated information exchange – Part 200: Guidelines for migration from Internet Protocol version 4 (IPv4) to Internet Protocol version 6 (IPv6)* (disponible en anglais seulement)

ISO/IEC 7498-1:1994, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base: Le modèle de base*

ISO/IEC 8824-1:2015 [UIT-T X.680:2015], *Technologies de l'information – Notation de syntaxe abstraite numéro un (ASN. 1): Spécification de la notation de base*

ISO/IEC 8825-4:2015 [UIT-T X.693:2015], *Technologies de l'information – Règles de codage ASN.1: Règles de codage XML (XER)*

ISO 9506 (toutes les parties), *Systèmes d'automation industrielle – Spécification de messagerie industrielle*

ISO 9506-1:2003, *Systèmes d'automation industrielle – Spécification de messagerie industrielle – Partie 1: Définition des services*

ISO 9506-2:2003, *Systèmes d'automation industrielle – Spécification de messagerie industrielle – Partie 2: Spécifications du protocole*

IEEE C37.111:1999, *IEEE standard common format for transient data exchange (COMTRADE) for power systems* (disponible en anglais seulement)

IEEE 754:1985, *IEEE standard for binary floating-point arithmetic* (disponible en anglais seulement)

RFC 768, *User datagram protocol*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 791, *Internet protocol – DARPA internet program – Protocol specification*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

¹ En cours d'élaboration. Stade au moment de la publication: IEC/DECFDIS 62351-4:2018.

² Au moment de la publication, une nouvelle édition était en cours d'élaboration.

RFC 792, *Internet control message protocol – DARPA internet program – Protocol specification*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 793, *Transmission control protocol – DARPA internet program – Protocol specification*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 826, *An Ethernet address resolution protocol or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 919, *Broadcasting internet datagrams*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 922, *Broadcasting internet datagrams in the presence of subnets*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 950, *Internet standard subnetting procedure*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 1112, *Host extensions for IP multicasting*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 2460, *Internet protocol, Version 6 (IPv6) specification*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 3629, *UTF-8, a transformation format of ISO 1646* – IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 4422, *Simple authentication and security layer (SASL)*, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 5246:2008, *The TLS protocol version 1.2*, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 5905, *Network time protocol (NTP) Version 4: Protocol and algorithms specification*, IETF, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 6120, *Extensible messaging and presence protocol (XMPP): Core*, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 6121, *Extensible messaging and presence protocol (XMPP): Instant messaging and presence*, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

RFC 6122, *Extensible messaging and presence protocol (XMPP): Address format*, disponible à l'adresse suivante: <http://www.ietf.org> (disponible en anglais seulement)

XEP-0198, *Stream management* (disponible en anglais seulement)

NOTE Cette spécification définit une extension de protocole XMPP pour la gestion active d'un flux XML entre deux entités XMPP, y compris les fonctionnalités de reconnaissance de strophe et de reprise de flux.

XEP-0199, *XMPP Ping* (disponible en anglais seulement)

NOTE Cette spécification définit une extension de protocole XMPP pour l'envoi de commandes ping au niveau de l'application sur des flux XML. Ces commandes ping peuvent être envoyées par un client à un serveur, par un serveur à un autre serveur ou de bout en bout.

3 Termes et définitions

Pour les besoins du présent document, les termes et définitions donnés dans l'IEC TS 61850-2 ainsi que les suivants s'appliquent.

L'ISO et l'IEC tiennent à jour des bases de données terminologiques destinées à être utilisées en normalisation, consultables aux adresses suivantes:

- IEC Electropedia: disponible à l'adresse <http://www.electropedia.org/>
- ISO Online browsing platform: disponible à l'adresse <http://www.iso.org/obp>

3.1

classe

description d'un ensemble d'objets qui partagent les mêmes attributs, services, relations et sémantiques

[SOURCE: IEC 61850-7-2:2010, 3.1]

3.2

client

entité demandant un service à un serveur et qui reçoit des messages non sollicités depuis un serveur

[SOURCE: IEC 61850-7-2:2010, 3.2]

3.3

dispositif

entité remplissant les fonctions de commande, d'actionnement et/ou de détection et est interfacée avec d'autres entités similaires dans un système d'automatisation

Note 1 à l'article: Les dispositifs seuls ne remplissent pas de fonctions de transport d'énergie.

[SOURCE: IEC 61850-7-2:2010, 3.3]

3.4

dispositif logique

entité représentant un ensemble de fonctions types d'un poste

[SOURCE: IEC 61850-7-2:2010, 3.6]

3.5

nœud logique

entité représentant une fonction type d'un poste

[SOURCE: IEC 61850-7-2:2010, 3.7]

3.6

dispositif physique

entité représentant les parties physiques d'un dispositif (matériel et système d'exploitation, etc.)

Note 1 à l'article: Les dispositifs physiques hébergent des dispositifs logiques.

[SOURCE: IEC 61850-7-2:2010, 3.8]

3.7

profils d'application et de transport

profil A et profil T

ensemble de protocoles utilisés pour une application spécifique

3.8

attribut de données

élément nommé d'objet de données qui a un type spécifique

3.9

objet de données (DataObject)

informations structurées significatives d'applications situées dans un dispositif d'automatisation qui peuvent être lues ou écrites

Note 1 à l'article: Cette définition est conforme à la définition d'un DataObject (objet de données) de l'IEC 61850-7-2. L'utilisation de MMS pour un DataObject correspond aux types de données et aux instanciations potentielles de variables MMS.

3.10

serveur

entité qui fournit des services à des clients ou émet des messages non sollicités

3.11

réseau privé

réseau utilisé par une seule entité qui gère tous les flux de données et dont les performances sont garanties en ce qui concerne la bande passante, le débit, le délai de transmission, la disponibilité, etc.

Note 1 à l'article: Si ces conditions ne sont pas satisfaites, le réseau est considéré comme étant un réseau public.

Note 2 à l'article: Un réseau privé peut reposer sur une infrastructure publique ou partagée, dès que le niveau de services peut être garanti.

3.12

réseau intelligent

application de l'intelligence informatique et des capacités de mise en réseau au système de distribution d'électricité, de manière à lui permettre de contrôler, protéger et optimiser automatiquement le fonctionnement de ses éléments interconnectés, depuis le système de production centralisée et décentralisée d'énergie en passant par le réseau haute tension et le système de distribution, jusqu'aux utilisateurs industriels et aux systèmes immotiques, aux installations de stockage d'énergie et aux consommateurs finals et à leurs thermostats, aux véhicules électriques, aux appareils et autres articles électroménagers

4 Abréviations

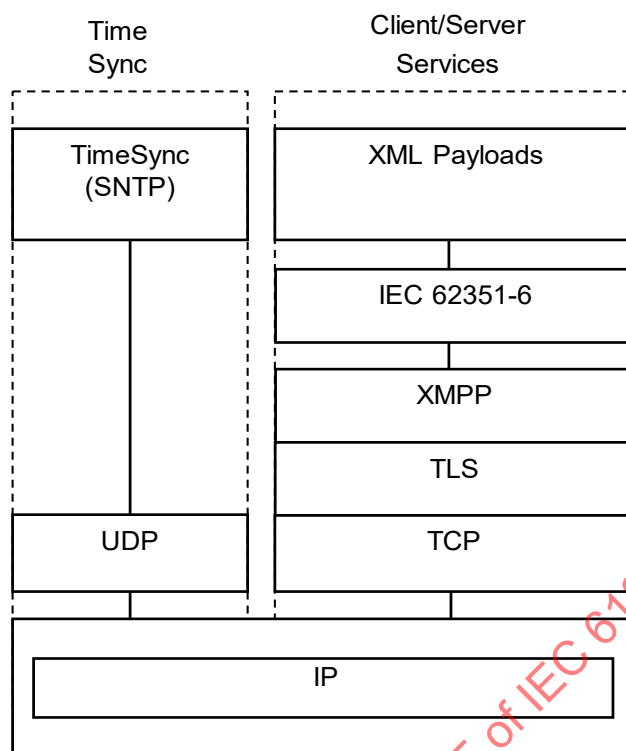
ACSI	Abstract Communication Service Interface (interface abstraite des services de communication)
BRCB	Buffered Report Control Block (bloc de commande de rapport mis en mémoire tampon)
CDC	Classe de Données Communes
DER	Distributed Energy Resource (ressources énergétiques réparties)
DNS	Domain Name System (système de nom de domaine)
DSO	Distribution System Operator (gestionnaire de réseau de distribution)
E2E	End-to-end (bout en bout)
FC	Functional Constraint (contrainte fonctionnelle)
FCD	Functionally Constrained Data (données sous contrainte fonctionnelle)
IED	Intelligent Electronic Device (dispositif électronique intelligent)
IETF	Internet Engineering Task Force

IP	Internet Protocol (protocole Internet)
JID	Jabber Identifier (identifiant Jabber)
LAN	Local Area Network (réseau local)
LCB	Log Control Block (bloc de commande de journal)
LD	Logical Device (dispositif logique)
LN	Logical Node (nœud logique)
MMS	Manufacturing Message Specification (spécification de messagerie industrielle) (ISO 9506)
OSI	Open Systems Interconnection (interconnexion de systèmes ouverts)
PDU	Protocol Data Unit (unité de données de protocole)
PICS	Protocol Implementation Conformance Statement (déclaration de conformité de mise en œuvre du protocole)
PIXIT	Protocol Implementation Extra Information (informations complémentaires de mise en œuvre du protocole)
RBAC	Role-Based Access Control (contrôle d'accès à base de rôle)
RFC	Request for Comments (demande de commentaires)
SBO	Select Before Operate (sélection avant opération)
SCL	substation automation System Configuration Language (langue de configuration de système d'automatisation de poste) (IEC 61850-6)
SCSM	Specific Communication Service Mapping (mapping des services de communication spécifiques)
SCSM 8-1	mapping des services de communication spécifiques tel que défini par l'IEC 61850-8-1
SCSM 8-2	mapping des services de communication spécifiques tel que défini par l'IEC 61850-8-2
SG	Setting Group (groupe de réglage)
SNTP	Simple Network Time Protocol
SV	Sampled Values (valeurs échantillonnées)
TCP	Transmission Control Protocol
UCA	Utility Communication Architecture
URCB	Unbuffered Report Control Block (bloc de commande de rapport non mis en mémoire tampon)
TUC	Temps universel coordonné
VMD	Virtual Manufacturing Device (dispositif de fabrication virtuel)
VPP	Virtual Power Plant (centrale électrique virtuelle)
WAN	Wide Area Network (réseau étendu)
XER	XML Encoding Rules (Règles de codage XML)
XMPP	Extensible Messaging Presence Protocol
XSD	XML Schema Definition (définition de schéma XML)

5 Vue d'ensemble

5.1 Généralités

Le présent document a pour objet de fournir des instructions/spécifications détaillées relatives aux mécanismes et aux règles exigées pour mettre en œuvre les principaux services ACSI spécifiés dans l'IEC 61850-7-2, tout en utilisant le protocole XMPP, ainsi qu'à l'utilisation du protocole SNTP pour la synchronisation temporelle (voir la Figure 1).



Anglais	Français
Time Sync	Synchronisation temporelle
Client/Server Services	Services client/serveur
XML Payloads	Données utiles XML

Figure 1 – Vue d'ensemble des fonctionnalités et des profils

L'Article 6 contient des détails supplémentaires concernant l'utilisation et la spécification réelle de chaque profil.

NOTE Les profils de pile décrits ici commencent au niveau de la couche IP. L'indication des spécifications qui doivent être utilisées pour les couches sous IP est hors du domaine d'application de cette SCSM.

5.2 Mapping des services client/serveur

5.2.1 Généralités

Le profil de communication client/serveur doit être utilisé pour toute mise en œuvre revendiquant la conformité au présent document et déclarant la prise en charge pour un des services IEC 61850-7-2 présentés au Tableau 1.

Tableau 1 – Services exigeant un profil de communication client/serveur

Modèle IEC 61850-7-2	Service IEC 61850-7-2
Serveur	GetServerDirectory
Association	Associate
	Abort
	Release
Dispositif logique	GetLogicalDeviceDirectory
Nœud logique	GetLogicalNodeDirectory
	GetAllDataValues
Données	GetDataValues
	SetDataValues
	GetDataDirectory
	GetDataDefinition
Ensembles de données	GetDataSetValues
	DataSetValues
	CreateDataSet
	DeleteDataSet
	GetDataSetDirectory
Définition de bloc de commande de groupe de réglage	SelectActiveSG
	SelectEditSG
	SetEditSGValue
	ConfirmEditSGValues
	GetEditSGValue
	GetSGCBValues
Bloc de commande de rapport	Rapport
	GetBRCBValues
	SetBRCBValues
	GetURCBValues
	SetURCBValues
Bloc de commande de journal	GetLCBValues
	SetLCBValues
	GetLogStatusValues
	QueryLogByTime
	QueryLogAfter
GOOSE ^a	GetGoCBValues
	SetGoCBValues
Valeurs échantillonnées multicast ^a	GetMSVCBValues
	SetMSVCBValues
Control	Select
	SelectWithValue
	Cancel
	Operate
	CommandTermination
	TimeActivatedOperate
Transfert de FICHIER	GetFile
	SetFile
	DeleteFile
	GetFileAttributeValues
^a Les services SendGOOSEMessage, GetGoReference, GetGOOSEElementNumber, SendMSVMessage, GetMsvReference et GetMSVElementNumber ne sont pas mappés dans le présent document, car un mapping XMPP ne satisferait pas aux performances exigées.	

Les messages de demande et de réponse des services ACSI client/serveur doivent être représentés par des messages XML transportés entre l'application appelante et l'application appelée sur le protocole XMPP. En fonction du protocole XMPP, un flux XML doit être en premier lieu établi entre les deux applications et un serveur XMPP. Ensuite, les messages XML représentant les services ACSI doivent être intégrés dans les constructions XML définies

par XMPP et les strophes appelées. Chaque strophe doit être envoyée par l'appelant dans son flux XMPP, acheminée par le serveur XMPP, puis délivrée à l'application appelée via son propre flux XMPP.

NOTE Voir A.2.2 pour une description formelle des couches de communication concernées de ce profil.

5.2.2 Données utiles XML

Lors de la conception des messages XML transportés sur XMPP, les mises en œuvre doivent avoir conscience que chaque message XML est composé de trois parties séparées et imbriquées (voir la Figure 2):

- La PDU de service représentant le service ACSI lui-même est le niveau imbriqué le plus profond. Il est représenté dans les messages XML par un élément comme <confirmed-RequestPDU> ou <confirmed-ResponsePDU> pour une demande de service et une réponse de service, respectivement. Le présent document vise globalement à décrire la manière dont ces données utiles applicatives sont construites. Leur structure doit être formellement définie par le schéma XML fourni à l'Annexe G.3.
- La PDU de service est ensuite encapsulée dans des éléments de sécurité de bout en bout, contenant des informations relatives à la sécurité comme les certificats, les clés de session et d'autres paramètres. Ces informations permettent d'établir et de gérer un canal sécurisé de bout en bout entre les applications appelantes et appelées.
- L'élément de contexte d'association est l'élément enfant direct de la strophe XMPP, qui encapsule la PDU de service et les éléments de sécurité. Il transporte un identifiant utilisé pour créer et gérer l'association entre les applications. La Figure 2 donne un exemple de données utiles XML correspondant à une demande ACSI GetDataValues, en partant du principe que le chiffrement des données utiles applicatives n'est pas activé.

Les deux dernières parties mentionnées ci-dessus sont spécifiées dans l'IEC 62351-4.

<Élément(s) de gestion du contexte d'association>	Contexte d'association
<Éléments de gestion de la sécurité de bout en bout>	Éléments de bout en bout
< >	
[...]	
< >	
<confirmed-RequestPDU>	PDU de service
<invokeID>2676121</invokeID>	(données utiles applicatives)
<ConfirmedServiceRequest>	
<read specificationWithResult="false" >	
<variableAccessSpecification>	
<listOfVariable>	
<SEQUENCE>	
<variableSpecification>	
<name>	
<domain-specific>	
<domainId>IEDNameLDINst</domainId>	
<itemId>LLN0\$ST\$Beh\$stVal</itemId>	
</domain-specific>	
</name>	
</variableSpecification>	
</SEQUENCE>	
</listOfVariable>	
</variableAccessSpecification>	
</read>	
</ConfirmedServiceRequest>	
</confirmed-RequestPDU>	
</ >	
[...]	
</ >	
<Éléments de gestion de la sécurité de bout en bout>	Éléments de sécurité
<Élément(s) de gestion du contexte d'association >	Contexte d'association

Figure 2 – Exemple de données utiles XML

5.2.3 Accords de mise en œuvre

La prise en charge du mécanisme de sécurité de bout en bout ci-dessus est obligatoire pour toutes les mises en œuvre conformes au présent document. Toutefois, pour un déploiement donné, la fonction de sécurité de bout en bout peut être désactivée. Dans ce cas, le comportement des applications client et serveur doit être le suivant:

- La désactivation du mécanisme de sécurité de bout en bout est une configuration dédiée des deux applications IEC 61850, client et serveur. Cette configuration est hors du domaine d'application du présent document.
- Si les deux applications qui communiquent sont correctement configurées pour fonctionner sans sécurité de bout en bout, les données utiles échangées ne doivent pas contenir d'éléments de sécurité et la PDU de service doit être directement encapsulée dans la partie de contexte d'association (voir G.4).
- L'utilisation des strophes XMPP et la gestion du contexte d'association doivent rester identiques à ceux correspondant à l'application d'une sécurité de bout en bout, et doivent donc être telles que décrites dans l'IEC 62351-4.
- Si les entités client et serveur sont configurées de manière différente, l'une utilisant la sécurité de bout en bout et l'autre pas, aucune interaction n'est possible. Toutes les demandes d'établissement d'une communication doivent échouer.

5.2.4 XMPP

XMPP est un protocole défini au niveau de l'IETF (Internet Engineering Task Force) par la RFC 6120, la RFC 6121 et la RFC 6122. Outre les spécifications de base, la XMPP Standards Foundation développe des extensions appelées XEP.

Pour le présent SCSM, le rôle de XMPP est de transporter les données utiles XML entre deux applications IEC 61850

L'Article 6 donne des détails relatifs à la manière dont le présent mapping utilise XMPP, y compris les spécifications de base et les XEP possibles.

5.3 Services de synchronisation temporelle

Les services de synchronisation temporelle spécifiés dans l'IEC 61850-7-2 doivent être mappés sur le protocole SNTP (Simple Network Time Protocol) de l'IETF.

NOTE 1 L'utilisation de SNTP sur des réseaux publics peut exiger des techniques d'ingénierie particulières visant à assurer une cybersécurité appropriée (l'utilisation d'horloges redondantes, la reconnaissance des horloges dignes de confiance, par exemple). La définition ou la recommandation de ce type de techniques va au-delà du domaine d'application du présent document.

NOTE 2 Voir A.2.3 pour une description formelle des couches de communication concernées de ce profil.

6 Utilisation de XMPP

6.1 Principes

XMPP repose sur une architecture répartie de serveurs XMPP. Du point de vue de XMPP, une application conforme au présent SCSM doit agir comme un client XMPP connecté à un ou plusieurs serveurs XMPP.

NOTE Les serveurs XMPP peuvent être utilisés par l'un des acteurs impliqués dans les interactions IEC 61850 au niveau applicatif ou par un fournisseur tiers.

Chaque IED doit avoir un compte sur le serveur XMPP auquel il est connecté, et la partie ressource du JID utilisé pour identifier une connexion donnée doit être fixe. Pour qu'un protocole IEC puisse utiliser la même connexion en fonction de XMPP, la valeur générique IEC doit être utilisée.

Exemple de JID complet: myIED@DSO.org/IEC.

6.2 Établissement d'une connexion

6.2.1 Généralités

Pour établir une connexion au serveur XMPP, un IED doit procéder comme suit:

- Initier une connexion TCP au serveur
- Ouvrir un flux XML
- Négocier les fonctions de flux
- S'authentifier auprès du serveur XMPP
- Sélectionner une ressource
- Demander la liste et envoyer la présence initiale

Ces différentes étapes sont spécifiées en détail par XMPP (RFC 6120). De plus, les règles spécifiées de 6.2.2 à 6.2.3 doivent s'appliquer.

6.2.2 Utilisation de TLS et de SASL

TLS doit être utilisé dans tous les cas et doit être mis en œuvre comme indiqué dans l'IEC 62351-6. Toutefois, plusieurs cas doivent être pris en considération:

1. XMPP sur TLS où la négociation TLS fait partie de la négociation du flux XMPP;
2. XMPP sur TLS où la session TLS est un tunnel dans lequel le flux XMPP est établi;

3. XMPP sur TLS où la négociation TLS fait partie de la négociation du flux XMPP. Toutefois, dans un tunnel TLS existant dans lequel le flux XMPP est établi (combinaison de 1 et de 2).

SASL doit être utilisé dans tous les cas pour l'authentification des points d'extrémité XMPP. Toutefois, plusieurs cas doivent être pris en considération, selon la manière dont TLS est utilisé:

- le mécanisme SASL EXTERNAL (dans lequel l'authentification est assurée par le certificat échangé pendant la négociation `<stream:feature/>` `<starttls/>`) peut être obtenu pour 1 et 3, dès que le certificat TLS utilisé pendant la négociation inclut l'identité JID.
- le mécanisme SASL PLAIN (un simple mécanisme de mot de passe textuel clair) peut également être déployé pour 1 et 3, et il devient obligatoire dès que le certificat TLS utilisé pendant la négociation n'inclut pas l'identité JID.
- le mécanisme SASL PLAIN doit être utilisé comme étant la seule option pour 2, étant donné que le certificat utilisé dans ce cas par TLS n'est pas disponible au niveau du composant XMPP.

L'une des options proposées pour TLS et SASL doit être choisie et déployée, en fonction de l'infrastructure de transport, de la réglementation locale et/ou de la stratégie locale.

6.2.3 Compression de flux

La compression de flux XMPP peut être utilisée (voir XEP 0138), mais noter qu'elle n'est pas efficace pour les strophes impliquées dans une session de sécurité de bout en bout avec chiffrement.

6.3 Mapping des services ACSI

La plupart des services client/serveur sont des services sollicités définis par trois éléments: la demande, la réponse positive et la réponse négative (voir la Figure 3).

Nom de paramètre
Request
<paramètres de demande>
Response+
<paramètres de réponse>
Response-
<erreurs de service>

Figure 3 – Structure générique des services ACSI client/serveur

Les services sollicités doivent être mis en œuvre par les strophes `<iq>`. Le message Request doit être mis en œuvre par une strophe `<iq>` avec l'attribut "type" défini sur "set" ou "get", et le message Response+ doit être mis en œuvre par une strophe `<iq>` avec l'attribut "type" défini sur "result". Les messages Response- doivent être transmis sous la forme d'une strophe `<iq/>` avec type="result" ou sous la forme `<message/>` avec type="normal".

Les services non sollicités doivent être mis en œuvre sous la forme de strophes `<message/>` de type "normal".

NOTE Les données utiles représentant les services ACSI, c'est-à-dire les PDU de service spécifiées par le présent document, sont imbriquées dans les éléments utilisés pour la gestion de la sécurité de bout en bout et du contexte d'association (voir la Figure 2). Les deux derniers types d'éléments sont spécifiés dans l'IEC 62351-4. Les erreurs rencontrées pendant la gestion de ces éléments sont également spécifiées dans l'IEC 62351-4.

Les erreurs se produisant au niveau XMPP doivent utiliser le mécanisme d'erreur défini par XMPP.

Le Tableau D.1 définit, pour chaque service ACSI, le type de strophes et la valeur de l'attribut "type" qui doivent être utilisés. Noter que le mapping du service Associate (les trois dernières lignes du tableau) est plutôt décrit dans l'IEC 62351-4.

Les différents cas d'erreur (Response- et les erreurs au niveau XMPP) sont décrits en 8.3.3.4 et au Tableau D.2.

6.4 Utilisation de la présence XMPP

Les associations bipartites doivent être surveillées à l'aide des informations de présence des entités IEC 61850 (c'est-à-dire le client IEC 61850 et le serveur IEC 61850).

Lors de la demande Associate, le client IEC 61850 doit s'abonner à la présence du serveur IEC 61850 (si cela n'a pas déjà été fait). Le serveur peut également s'abonner à la présence du client. Une entité qui détecte l'état de présence indisponible de son homologue associé doit générer un abandon (Abort) local.

L'établissement de connexions TCP et de flux XMPP entre une entité de communication IEC 61850 et un serveur XMPP n'est pas dédié à une association et n'en fait donc pas partie. Ils doivent exister avant d'appeler le service Associate.

6.5 Utilisation de la liste

Comme le RFC 6121 le recommande, un client XMPP (puis un client ou serveur IEC 61850) doit demander sa liste à la fin de la création du flux.

Puis, pour des raisons de sécurité, le client XMPP doit vérifier la cohérence de sa liste et éventuellement la mettre à jour en conséquence.

6.6 Extensions XMPP

6.6.1 Utilisation de XMPP PING – XEP 0199

Les serveurs XMPP peuvent utiliser différentes méthodes de vérification de la présence de clients connectés, en général TCP Keepalive, qui détecte uniquement les connexions TCP interrompues ou XMPP PING (XEP 0199), qui détecte les problèmes sur les niveaux TCP et XMPP. Un client XMPP doit au moins pouvoir répondre à des strophes XMPP PING telles que définies dans XEP 0199.

NOTE Les autres utilisations de ce XEP sont hors du domaine d'application de ce SCSM, les entités pouvant, par exemple, envoyer des commandes ping XMPP pour vérifier la disponibilité d'un homologue (une entité IEC 61850 ou un serveur XMPP).

6.6.2 Utilisation de la gestion de flux – XEP 0198

Si les entités XMPP souhaitent améliorer la fiabilité du réseau, elles peuvent activer la gestion de flux conformément au XEP 0198. Le terme "entité XMPP" couvre ici non seulement les clients XMPP, mais également les serveurs XMPP.

Une mise en œuvre déclarant la prise en charge de XEP 0198 doit gérer les accusés de réception définis par ce XEP, pour vérifier que le nombre de strophes envoyées par une entité correspond au nombre de strophes gérées par l'entité de destination.

Il est recommandé qu'une entité reprenne le flux et renvoie les strophes non traitées lorsqu'elle détecte que ces deux nombres sont différents ou lorsque le flux XMPP a été interrompu. Pour un niveau fonctionnel minimal, la reprise peut être remplacée par une erreur généralisée déclenchée auprès des couches de pile supérieures.

NOTE La stratégie d'échange des piles (périodiquement, toutes les strophes ou toutes les N strophes), dépend des scénarii.

Les scénarii de reprise pouvant impliquer de renvoyer les strophes déjà traitées, toutes les entités (même celles qui ne prennent pas en charge XEP 0198) doivent vérifier l'attribut "id" de la strophe afin d'ignorer les doublons.

6.7 Accords de mise en œuvre – XMPP PING – XEP 0199

Le délai entre deux XMPP PING dépend de la configuration du serveur XMPP. La valeur par défaut recommandée est de 60 s.

7 Sécurité de bout en bout

Le mécanisme de sécurité de bout en bout utilisé dans le présent SCSM est spécifié par l'IEC 62351-4. L'IEC 62351-4 s'applique tant au présent SCSM qu'à l'IEC 61850-8-1 et donne les principes et les définitions abstraites des éléments de sécurité échangés entre les points d'extrémité appelants et appelés.

La session de sécurité de bout en bout est créée au moment de la création de l'association ACSI lors de l'échange initial des messages XMPP correspondant à l'Associate ACSI. Dans ces messages, les données utiles correspondant au service ACSI Associate sont complétées par les éléments de sécurité exigés pour établir la session de bout en bout: les jetons contenant les paramètres de chiffrement nécessaires (les certificats numériques, par exemple) et les signatures.

Pendant l'établissement de cette liaison, l'authentification mutuelle des deux points d'extrémité est assurée en fonction des certificats et signatures numériques échangés. Au même moment a lieu un échange de clé Diffie Hellman, fournissant les clés de session nécessaires à la sécurisation des échanges de données subséquents à l'aide d'algorithmes de chiffrement symétriques pour la protection de l'intégrité et la confidentialité des données pendant le transfert de données.

Les détails du mécanisme, y compris en particulier les options disponibles (signature seulement ou également chiffrement des données utiles, par exemple) sont définis dans l'IEC 62351-4.

8 Description des données utiles

8.1 Aperçu XSD

Les charges utiles XML transportées sur XMPP sont composées de plusieurs parties qui ne sont pas toutes définies dans le présent document. Les premiers niveaux des éléments enfants dans les strophes XMPP sont en réalité définis dans l'IEC 62351-4 et sont essentiellement liés à la gestion de la sécurité de bout en bout. Le présent article ne décrit que la partie applicative des données utiles, qui est formellement définie par le schéma XML de l'Annexe G.3.

MMSpdu complexType est le point d'entrée dans le XSD. Il propose un choix de différentes enveloppes pour l'encapsulation des différents services ACSI. Les enveloppes les plus souvent utilisées sont <confirmed-RequestPDU/> et <confirmed-ResponsePDU/> pour le mapping des services ACSI confirmés, et <unconfirmed-PDU/> pour le mapping des services ACSI non confirmés. Dans les enveloppes destinées aux services ACSI confirmés, complexType ConfirmedServiceRequest (et ConfirmedServiceResponse) propose un choix de différents éléments de service (<read/>, <write/>, <getNameList/>, par exemple). Le détail des enveloppes disponibles et des éléments de service est présenté au Tableau D.1. La manière de mapper les différents services ACSI à ces enveloppes et éléments de service est décrite dans le mapping détaillé de chaque service ACSI commençant à l'Article 9.

Noter que le mécanisme de sécurité de bout en bout peut impliquer de chiffrer les données utiles applicatives. Dans ce cas, l'élément défini par l'IEC 62351-4 pour contenir les données utiles applicatives n'est pas de type MMSpdu. Il contient plutôt la forme chiffrée de la valeur équivalente du type MMSpdu qui serait utilisée pour le mapping du même service ACSI lorsqu'aucun chiffrement n'est utilisé.

8.2 Objets de l'IEC 61850

8.2.1 Généralités

Pour le mapping des objets définis dans l'IEC 61850-7-2, le présent SCSM utilise certains concepts déjà utilisés par l'IEC 61850-8-1 et hérités du protocole MMS. Leurs noms apparaissent dans les données utiles XML, mais il n'est pas nécessaire de comprendre ces concepts en détail. En général, il est suffisant de savoir comment les classes ACSI sont mappées à ces concepts (voir le Tableau 2).

Tableau 2 – Mapping des classes ACSI aux concepts MMS

Classes ACSI	Concepts MMS
Serveur	VMD
Dispositif logique	Domaine
Nœud logique	NamedVariable
Objet de données	NamedVariable
Toutes les classes de bloc de commande	NamedVariable
DataSet	NamedVariableList
Journal	Journal
Fichiers	Fichiers

8.2.2 Nœud logique (LN)

La construction du contenu d'un nœud logique est définie dans l'IEC 61850-7-2, l'IEC 61850-7-3 et l'IEC 61850-7-4.

Ce SCSM mappe chaque instance de nœud logique à une variable nommée (concept hérité de MMS), dont le nom est celui de l'instance de nœud logique et dont la structure hiérarchique est complexe, puisque composée de plusieurs niveaux de composants.

La description de type de cette structure peut être créée par l'algorithme présenté à la Figure 4.

Pour chaque contrainte fonctionnelle (c) présente à la Figure 5³

Pour chaque DataObject (d) dans le LN

Si le FCD généré à partir de d,c n'est pas vide, alors ajouter un composant de ComponentName (d) du ComponentType (type de composant) déterminé par le FCD à la structure (s).

Fin de boucle

Si la structure (s) n'est pas vide, alors ajouter un composant de ComponentName (c) du ComponentType (s) au TypeDescription généré.

Fin de boucle

Figure 4 – Algorithme pour le mapping de nœud logique

L'ordre recommandé des composants obtenus est décrit à la Figure 5.

³ L'ordre recommandé assure la compatibilité avec les mises en œuvre UCA 2.0 existantes.

MX
ST
CO
CF
DC
SP
SG
RP
LG
BR
GO
GS (déconseillé)
SV
SE
MS
US (déconseillé)
EX
SR
OR
BL
RG
RS

Figure 5 – Liste ordonnée des contraintes fonctionnelles

NOTE 1 Les FC sont définies comme suit: RP – blocs de commande de rapport non mis en mémoire tampon; LG – blocs de commande de journaux; BR – blocs de commande de rapport mis en mémoire tampon; GO – blocs de commande GOOSE; GS – blocs de commande GSSE déconseillés; SV – valeurs substituées; SE- édition de groupe de réglage; MS – bloc de commande de valeurs échantillonnées multicast (selon l'IEC 61850-9-2); US – bloc de commande de valeurs échantillonnées unicast déconseillé; EX – espace de nommage pour extension de modèle; SR – suivi de service; CO – paramètres des services de commande; OR – réception d'opération; BL pour blocage; RG pour blocs de commande GOOSE acheminables et RS pour blocs de commande de valeurs échantillonnées multicast acheminables.

NOTE 2 Le FC "XX" défini dans l'IEC 61850-7-2 en tant que nom générique FC pour les services uniquement n'est pas mappé au SCSM MMS. Par conséquent, une demande qui utilise FC="XX" est rejetée.

Si aucun DataObject (d) n'instancie une contrainte fonctionnelle spécifique, cette contrainte fonctionnelle ne doit pas apparaître comme faisant partie du TypeDescription de NamedVariable.

Dans un composant, représentant une contrainte fonctionnelle, l'ordre des sous-composants est déterminé par l'ordre des noms des classes de données compatibles instanciées en tant que DO SCL (voir l'IEC 61850-6) dont les attributs de données ont la contrainte fonctionnelle spécifique.

Dans un composant représentant les DataObject de l'IEC 61850-7-4, l'ordre des sous-composants est déterminé par l'ordre des noms des attributs des classes de données communes telles que définies dans l'IEC 61850-7-3 ou les extensions IEC de celles-ci pour les IED revendiquant la conformité à ces normes, et le SCL doit refléter le même ordre.

La Figure 6 est la représentation graphique d'une description de type hiérarchique d'une instance de nœud logique, telle que produite par l'algorithme de la Figure 4.

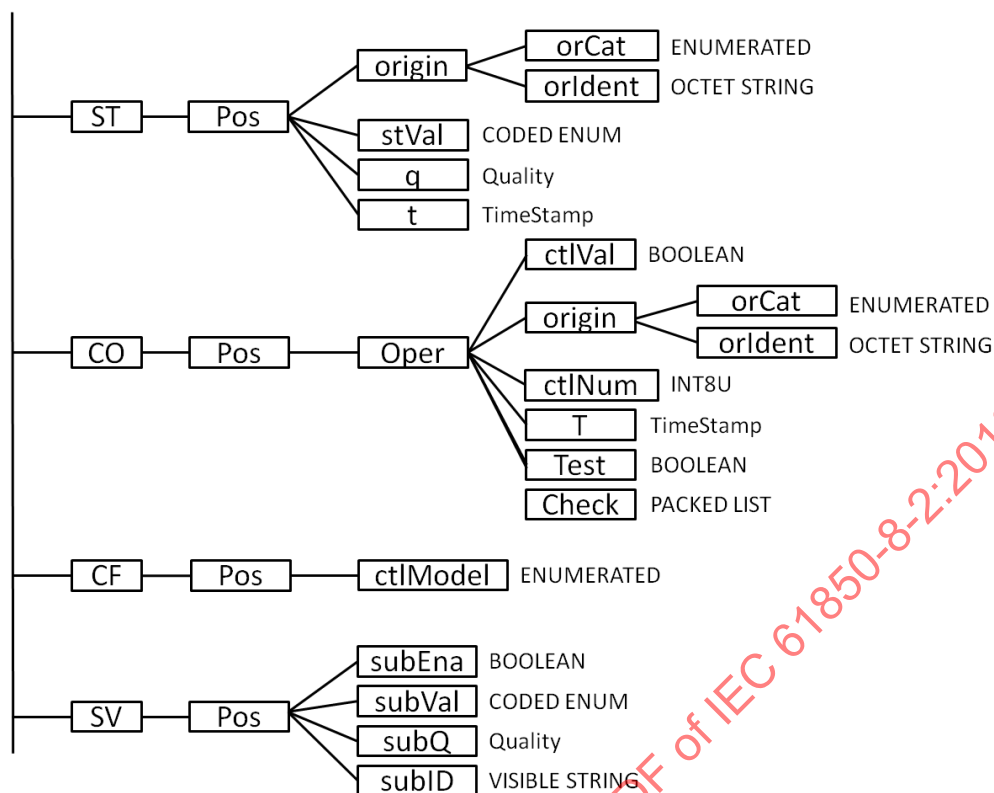


Figure 6 – Exemple de description de type de nœud logique

Outre ce mapping à une variable nommée avec une structure hiérarchique, un nœud logique peut également être perçu comme un ensemble de variables nommées séparées, une pour chaque composant (nœud ou feuille) dans la structure hiérarchique. Le nom de la variable nommée correspondant à un composant donné de l'arborescence est créé par la concaténation des noms de composant présents dans le chemin au composant donné, suivi du nom du composant donné lui-même, tous les noms étant séparés par "\$". L'imbrication de la concaténation doit être contrainte par la taille maximale d'identifiant: 64 caractères. La Figure 7 présente la liste des variables nommées aplanies déduites de la structure LN de la Figure 6.

```

CSWI1
CSWI1$ST
CSWI1$ST$Pos
CSWI1$ST$Pos$origin$orCat
CSWI1$ST$Pos$origin$orIdent
CSWI1$ST$Pos$stVal
CSWI1$ST$Pos$q
CSWI1$ST$Pos$t
CSWI1$CO
CSWI1$CO$Pos
CSWI1$CO$Pos$Oper$ctlVal
CSWI1$CO$Pos$Oper$origin$orCat
CSWI1$CO$Pos$Oper$origin$orIdent
CSWI1$CO$Pos$Oper$ctlNum
CSWI1$CO$Pos$Oper$T
CSWI1$CO$Pos$Oper$Test
CSWI1$CO$Pos$Oper$Check
CSWI1$CF
CSWI1$CF$Pos
CSWI1$CF$Pos$ctlModel
CSWI1$SV
CSWI1$SV$Pos
CSWI1$SV$Pos$subEna
CSWI1$SV$Pos$subVal
CSWI1$SV$Pos$subQ
CSWI1$SV$Pos$subID

```

Figure 7 – Liste des variables nommées aplanies correspondant à un nœud logique

La description de type d'une variable nommée aplanie correspondant à un composant donné est représentée par la sous-arborescence commençant au niveau de ce composant dans la structure globale de nœud logique.

8.2.3 Mapping des références de nœuds logiques à VariableAccessSpecifications

Les paramètres des services ACSI représentant les références à des nœuds logiques sont mappés dans les données utiles XML aux éléments XML de type VariableAccessSpecification. La VariableSpecification de ce type de mapping doit inclure un type ObjectName. Si un nom de LogicalDevice (dispositif logique) est présent dans le LNReference, l'élément enfant de l'ObjectName doit être domain-Specific (spécifique du domaine). Si "@" est présent, l'élément enfant doit être aa-Specific (spécifique à l'application). Si aucun n'est présent, l'élément enfant doit être vmd-Specific.

Dans un élément spécifique au domaine, domainId doit être le nom du LogicalDevice et ItemId celui du LogicalNode. La Figure 8 est un exemple de mapping pour le LDName/LNName = CONTROL/LLN0 de référence.

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL </domainId>
            <itemId> LLN0 </itemId>
          </domain-specific>
        </name>
      </variableSpecification>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 8 – Mapping XML d'une LNReference avec un accès direct

Une autre spécification doit être prise en charge lorsque la LNReference est contrainte à un FC spécifique (dans le cas de GetAllDataValues, voir 12.3.1). La Figure 9 est un exemple de VariableAccessSpecification dans laquelle un élément alternateAccess est utilisé pour coder la contrainte fonctionnelle ST.

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL </domainId>
            <itemId> GGIO1 </itemId>
          </domain-specific>
        </name>
      </variableSpecification>
      <alternateAccess>
        <CHOICE>
          <unnamed>
            <selectAlternateAccess>
              <accessSelection>
                <component> ST </component>
              </accessSelection>
            </selectAlternateAccess>
          </unnamed>
        </CHOICE>
      </alternateAccess>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 9 – Mapping XML d'une LNReference avec un accès alternatif

8.2.4 Mapping d'une référence DataObjects avec VariableAccessSpecifications

Les nœuds logiques, selon l'IEC 61850-7-2, sont constitués d'un ou de plusieurs DataObjects. Les DataObjects eux-mêmes peuvent être construits avec des sous-objets SubDataObject qui peuvent eux-mêmes contenir des SubDataObjects. Les variables nommées aplanies peuvent être associées à des DataObjects et des SubDataObjects dont le nom repose sur le composant nommé de manière hiérarchique des données se trouvant dans le nœud logique. Chaque niveau de la hiérarchie est délimité par un "\$".

EXEMPLES: <LNVariableName>\$<FC>\$<LNDataObjectName1> (XCBR1\$ST\$Pos, par exemple)

<LNVariableName>\$<FC>\$<LNDataObjectName1>\$<SubDataObjectName1> (MMXU1\$MXA\$phsA, par exemple)

Le moyen préférentiel consiste à mapper ce type de références en tant que VariableSpecification dans une VariableAccessSpecification, et de coder la variable nommée aplanie dans l'élément itemId (voir l'exemple de la Figure 10).

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL</domainId>
            <itemId> XCBR1$ST$Pos</itemId>
          </domain-specific>
        </name>
      </variableSpecification>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 10 – Mapping XML direct d'un FCD

Toutes les références d'un SubDataObject qui contient une matrice doivent utiliser une AlternateAccessSpecification dans les variableAccessSpecifications dès que l'un des éléments d'ensemble est adressé (voir 8.2.6).

8.2.5 Mapping de la référence DataAttributes (DataAttr) à VariableAccessSpecifications

Les attributs de données DataAttr des DataObjects correspondent de manière similaire aux DataObjects. Cependant, le nom d'attribut est également inclus dans la hiérarchie (voir les exemples ci-dessous et la Figure 11).

EXEMPLES: <LNVariableName>\${FC}\${LNDDataName1}\${AttributeName1}> (par exemple, XCBR1\$ST\$Pos\$stVal)

<LNVariableName>\${FC}\${LNDDataName1}\${AttributeName1}\${subDataAttributeName1}>

(par exemple, XCBR1\$ST\$Pos\$origin\$orCat)

```

<variableAccessSpecification>
  <listOfVariable>
    <SEQUENCE>
      <variableSpecification>
        <name>
          <domain-specific>
            <domainId> CONTROL</domainId>
            <itemId> XCBR1$ST$Pos$stVal</itemId>
          </domain-specific>
        </name>
      </variableSpecification>
    </SEQUENCE>
  </listOfVariable>
</variableAccessSpecification>

```

Figure 11 – Mapping XML direct d'un FCDA

Toutes les références d'un DataAttribute, SubDataAttribute qui contient une matrice doivent utiliser une AlternateAccessSpecification dans les variableAccessSpecifications dès que l'un des éléments d'ensemble est adressé (voir 8.2.6).

8.2.6 Utilisation de l'accès alternatif pour les références DataObjects et DataAttributes

Une référence doit utiliser un accès alternatif dès qu'un élément d'ensemble est adressé. Toutefois, l'accès alternatif peut être utilisé dans d'autres cas. L'accès alternatif consiste à ajouter un élément `<alternateAccess/>` après l'élément `<variableSpecification/>`. Voir également les articles relatifs à la conformité pour AlternateAccess en 24.2.2.2.1.

Le caractère "\$" n'est pas admis dans les valeurs des éléments `<component/>` dans la partie d'accès alternatif.

Il est recommandé de toujours utiliser le choix `<unnamed/>` dans la partie d'accès alternatif.

Les exemples ci-dessous présentent l'utilisation de ce type d'accès alternatifs dans les cas respectifs suivants:

Figure 12: un composant imbriqué sans matrice.

NOTE Cette référence peut avoir été codée sans accès alternatif avec `itemId= LLN0$ST$Beh$stVal`

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDevice</domainId>
      <itemId>LLN0</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <component>ST</component>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>Beh</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <component>stVal</component>
                        </accessSelection>
                        <alternateAccess/>
                      </selectAlternateAccess>
                    </unnamed>
                  </CHOICE>
                </alternateAccess>
              </selectAlternateAccess>
            </unnamed>
          </CHOICE>
        </selectAlternateAccess>
      </unnamed>
    </CHOICE>
  </alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>

```

Figure 12 – Accès alternatif sans élément d'ensemble

Figure 13. un composant imbriqué dans un élément d'ensemble. Noter que le chemin de composant avant l'index peut avoir été codé en tant que variable aplanie dans l'élément <variableSpecification/> (voir la Figure 14).


```

</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>

```

Figure 13 – Accès alternatif avec élément d'ensemble

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

```

<variableSpecification>
  <name>
    <domain-specific>
      <domainId>LDevice</domainId>
      <itemId>MHAI1$MX$HA$phsAHar</itemId>
    </domain-specific>
  </name>
</variableSpecification>
<alternateAccess>
  <CHOICE>
    <unnamed>
      <selectAlternateAccess>
        <accessSelection>
          <index>7</index>
        </accessSelection>
        <alternateAccess>
          <CHOICE>
            <unnamed>
              <selectAlternateAccess>
                <accessSelection>
                  <component>cVal</component>
                </accessSelection>
                <alternateAccess>
                  <CHOICE>
                    <unnamed>
                      <selectAlternateAccess>
                        <accessSelection>
                          <component>mag</component>
                        </accessSelection>
                        <alternateAccess>
                          <CHOICE>
                            <unnamed>
                              <selectAlternateAccess>
                                <accessSelection>
                                  <component>f</component>
                                </accessSelection>
                                <alternateAccess/>
                              </selectAlternateAccess>
                            </unnamed>
                          </CHOICE>
                        </alternateAccess>
                      </selectAlternateAccess>
                    </unnamed>
                  </CHOICE>
                </alternateAccess>
              </selectAlternateAccess>
            </unnamed>
          </CHOICE>
        </selectAlternateAccess>
      </unnamed>
    </CHOICE>
  </selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>
</selectAlternateAccess>
</unnamed>
</CHOICE>
</alternateAccess>

```

Figure 14 – Accès alternatif avec variable aplanie et élément d'ensemble

8.3 Mapping des attributs de données de l'IEC 61850-7-2

8.3.1 BasicTypes

Dans le XMLpayload, les BasicTypes de l'IEC 61850-7-2 sont mappés aux éléments XML décrits au Tableau 3. Ces éléments sont ceux définis dans le fichier XSD pour les données complexType (voir l'Annexe G).

Tableau 3 – Mapping des BasicTypes ACSI

Nom BasicType IEC 61850-7-2	Nom d'élément XML	Type XML utilisé pour la valeur	Commentaire
BOOLEAN	<boolean/>	BOOLEAN	Exemple: <boolean>true</boolean>
INT8	<integer/>	INTEGER	INTEGER est défini dans XSD en tant que restriction de xsd:integer Exemple: <integer>-99</integer>
INT16	<integer/>		
INT32	<integer/>		
INT64	<integer/>		
INT8U	<unsigned/>		INTEGER est défini dans XSD en tant que restriction de xsd:integer Exemple: <unsigned>99</unsigned>
INT16U	<unsigned/>		
INT24U	<unsigned/>		
INT32U	<unsigned/>		
FLOAT32	<floating-point/>	OCTET_STRING	OCTET_STRING représentant 4 octets. Plage de valeurs et précision comme indiqué par la virgule flottante de précision IEEE 754 NOTE 1 La valeur de Not-A-Number (NaN) est valable. Toutefois, le traitement d'une valeur indiquant Not-A-Number (NaN) est un problème local. NOTE 2 OCTET_STRING est une restriction de xsd:string pour le codage des valeurs hexadécimales
ENUMERATED	<integer/>	INTEGER	INTEGER est défini dans XSD comme une restriction de xsd:integer. Utilisé ici pour un ensemble ordonné de valeurs, défini lorsque le type est utilisé. Voir 8.3.2.2. Exemple: Beh.stVal = Essai (3) serait codé <integer>3</integer>
CODED ENUM	<bit-string/>	BIT_STRING	BIT_STRING est un type xsd:string, restreint à zéro ou plusieurs caractères "0" et "1" (voir 8.3.2.1). Utilisé ici pour un ensemble ordonné de valeurs, défini lorsque le type est utilisé. Voir également 8.3.2.3. Exemple: dbPos est une chaîne binaire de 2 bits <bit-string>01</bit-string>
OCTET STRING	<octet-string/>	OCTET_STRING	OCTET_STRING est une restriction de xsd:string pour le codage des valeurs hexadécimales. La longueur maximale doit être définie lorsque le type est utilisé. Voir également 8.3.2.4. Exemple: DPC ou Ident est une chaîne de 64 octets. Un exemple avec 7 octets serait codé comme suit: <octet-string>53544154494F4E</octet-string>
VISIBLE STRING	<visible-string/>	VisibleString	VisibleString est un xsd:string restreint à des caractères visibles. La longueur maximale doit être définie lorsque le type est utilisé Voir 8.3.2.5. Exemple: RptID d'un ReportControl est un Visible String129 <visible-string>CONTROL/GGIO1\$BR\$simpleBReport05</visible-string>

Nom BasicType IEC 61850-7-2	Nom d'élément XML	Type XML utilisé pour la valeur	Commentaire
UNICODE STRING	<mMString/>	MMString	Défini dans XSD comme un contenu mixte de texte et d'éléments XML codant les caractères de commande. La longueur maximale doit être définie lorsque le type est utilisé. Voir 8.3.2.6. Exemple: <mMString>ābcdēfgh<nul/>ījkℓμñópq<soh/>r\$tùvwxyz</mMString>
Pour les besoins de la suite des articles du présent document, la précision de unsigned/integer doit être déterminée par l'IEC 61850-7-2 ou l'IEC 61850-7-3, sauf indication contraire.			

8.3.2 Définitions additionnelles de BasicType

8.3.2.1 Déclaration générale pour BIT STRING

Dans la présente norme, la référence à un BIT STRING doit indiquer l'utilisation d'un élément avec name="bit-string" et type="BIT_STRING" tel que défini par le XSD normatif (voir l'Annexe G).

BIT_STRING est une restriction du type xsd:string, restreint à zéro ou plusieurs caractères "0" et "1".

NOTE Dans le présent document, les caractères d'un BIT_STRING sont toutefois appelés bits.

Bit(0) du BIT STRING est mappé au caractère le plus à gauche de la chaîne XML.

Le nombre de bits transportés par une chaîne binaire de longueur variable envoyée doit toujours être égal à la longueur maximale de la chaîne binaire de longueur variable définie du côté expéditeur, de manière à assurer le transfert de compatibilité sémantique. Une chaîne binaire de longueur zéro (0) n'est pas admise dans le contexte de l'IEC 61850-8-2.

8.3.2.2 ENUMERATED

L'ENUMERATED de l'IEC 61850-7-2 doit mapper vers un élément <integer/> avec une valeur xsd:integer. Les valeurs qui sont 0 ou plus doivent être réservées pour les valeurs normalisées dans l'IEC 61850. Les valeurs qui sont inférieures à 0 doivent être considérées comme étant des valeurs privées, sauf pour l'énumération Multiplier de l'IEC 61850-7-3 où la valeur de l'énumération donne la valeur du multiplicateur à utiliser. La plage de valeurs autorisées est spécifiée dans l'IEC 61850-7-2, l'IEC 61850-7-3 et l'IEC 61850-7-4.

La première valeur ordonnée des valeurs énumérées normalisées doit être définie à la valeur zéro (0).

Les valeurs énumérées à l'extérieur de la plage de valeurs normalisée doivent être considérées comme des extensions de valeurs et ne doivent pas causer une erreur de protocole.

Pour les valeurs énumérées de l'IEC 61850, où les valeurs numériques effectives sont spécifiées, les valeurs entières dans le contexte de l'IEC 61850-8-2 doivent être les mêmes que les valeurs numériques spécifiées.

8.3.2.3 CODED ENUM

Le CODED ENUM de l'IEC 61850-7-2 et mappé à un élément <bit-string/> avec un type de valeur BIT_STRING, comme défini dans le XSD normatif (voir 8.3.2.1). La taille de la chaîne doit être le nombre de bits exigé pour représenter la valeur énumérée maximale définie dans

l'IEC 61850-7-2. Elle doit représenter le codage d'entier non signé des valeurs énumérées dans l'ordre spécifié dans l'IEC 61850-7-2.

Pour les besoins de l'IEC 61850-8-2, les CODED ENUM qui apparaissent dans un objet PACKED LIST ou un ensemble d'autres objets CODED ENUM sont traités comme des cas particuliers.

8.3.2.4 OCTET STRING

L'OCTET STRING de l'IEC 61850-7-2 doit mapper vers un élément <octet-string/> avec une valeur de type OCTET_STRING. OCTET_STRING est une restriction de xsd:string pour le codage de valeurs hexadécimales, c'est-à-dire qu'il repose sur le jeu de caractères [0-9][A-F][a-f], mais en incluant les éventuels espaces supplémentaires. La taille maximale doit être celle spécifiée dans l'IEC 61850-7-2 et l'IEC 61850-7-3 et n'inclut pas les espaces supplémentaires ajoutés pour la lisibilité. La longueur de la chaîne NULL est 0.

NOTE Le terme "octet" est utilisé dans le présent document pour désigner les deux caractères d'un OCTET_STRING représentant la valeur hexadécimale d'un octet binaire.

8.3.2.5 VISIBLE STRING

Le VISIBLE STRING de l'IEC 61850-7-2 doit mapper vers un élément <visible-string/> avec une valeur de type VisibleString. VisibleString est défini dans le XSD en tant que xsd:string restreint aux caractères visibles, c'est-à-dire à la plage comprise entre Espace (code ascii 32) et Tilde (code ascii 126). La taille maximale doit être comme indiquée dans l'IEC 61850-7-2 et l'IEC 61850-7-3. La longueur de la chaîne NULL est 0.

Le jeu de caractères doit être restreint à ISO646String.

8.3.2.6 UNICODE STRING

L'UNICODE STRING de l'IEC 61850-7-2 doit être mappé vers un élément <mMSString/> avec une valeur mMSString complexType. mMSString complexType peut contenir un mélange de texte (codage UTF8) et des caractères de commande codés en tant qu'éléments XML explicites tels que <nul/> ou <sch/>. La taille maximale doit être le nombre de caractères Unicode, comme indiqué dans l'IEC 61850-7-2 et l'IEC 61850-7-3. La sérialisation d'un caractère Unicode a lieu comme indiqué dans le RFC 3629, et peut impliquer jusqu'à 4 octets. La longueur de la chaîne NULL est 0.

8.3.2.7 Matrice

Une matrice doit mapper vers un élément XML <array/> avec au moins un élément XML enfant.

Si les éléments de la matrice sont des éléments BasicType, les éléments XML enfants doivent être directement ceux décrits au Tableau 3 avec un élément XML correspondant à chaque élément d'ensemble.

Sinon, les éléments enfants doivent être des éléments <structure/> qui peuvent inclure de manière récursive des éléments BasicType et/ou <structure/>.

Voir dans le fichier XSD de l'Annexe G la définition des éléments "array" dans le complexType "Data".

Exemple: codage d'une matrice de 2 points

```
<array>
  <structure>
    <floating-point>3f800000</floating-point>
```

```

        <floating-point>40000000</floating-point>
        <floating-point>40400000</floating-point>
    </structure>
    <structure>
        <floating-point>40800000</floating-point>
        <floating-point>40a00000</floating-point>
        <floating-point>40c00000</floating-point>
    </structure>
</array>

```

8.3.3 ACSITypes communs

8.3.3.1 ObjectName

Ce type apparaît uniquement dans le contexte d'ObjectReference. Le jeu de caractères autorisé dans les instances d'ObjectName est défini dans l'IEC 61850-7-2.

8.3.3.2 ObjectReference

8.3.3.2.1 Généralités

Le mapping de référence d'objet de l'IEC 61850-7-2 dépend du contexte dans lequel la référence d'objet est définie:

- Dans le domaine d'application des attributs de bloc de commande: voir 8.3.3.2.2.
- Dans le domaine d'application des attributs de données (Tracking CDC,ORG): voir 8.3.3.2.3.
- Dans le domaine d'application du DataRef des rapports: voir 8.3.3.2.4.

Dans tous les cas, ObjectReference est mappé en tant que VISIBLE STRING de longueur maximale de 129 octets.

8.3.3.2.2 Référence d'objet dans des blocs de commande

La valeur doit être construite comme suit:

- Objets de domaine d'application spécifiques AA: @<Nom d'objet>
- Spécifique au domaine: <Nom de domaine>/<Nom d'objet>
- Spécifique VMD: /<Nom d'objet>

La taille maximale de Nom d'objet et de Nom de domaine doit être de 64.

Le jeu de caractères admis est défini et est le même qu'ObjectName plus les caractères "\$", "/" et "@".

8.3.3.2.3 Référence d'objet dans des CDC (Tracking (suivi), ORG).

La valeur doit être construite comme indiqué dans l'IEC 61850-7-2.

Le jeu de caractères admis est défini dans l'IEC 61850-7-2. La valeur doit être construite comme suit:

- Référence à un dispositif logique:

<LDName>

- Référence à un nœud logique:
<LDName>/<LNName>
- Référence à un DataSet:
<LDName>/<LNName>.<DataSetName>
- Référence à un DataObject:
<LDName>/<LNName>.<DataObjectName>[.<SubDataObjectName>[. ...]]
- Référence à un DataAttribute:
<LDName>/<LNName>.<DataObjectName>[.<SubDataObjectName>[. ...]].<DataAttributeName>[(.<NumArrayElement>)]<SubDataAttributeName>[. ...]]
- Référence à un bloc de commande
<LDName>/<LNName>.<CBName>

8.3.3.2.4 Référence d'objet dans le DataRef des rapports.

La valeur doit être construite comme suit:

<Nom de domaine>/<Nom d'objet>

Le jeu de caractères admis est défini dans l'IEC 61850-7-2.

La valeur doit être construite comme suit:

- Référence à un FCD DataObject:
<LDName>/<LNName>\$<FC>\$<DataObjectName>[\$<SubDataObjectName>[(.<NumArrayElement>)]\$...]]

Exemple: le datRef représentant la référence ACSI LDevice/MHA11.HA.phsAHar(7) contrainte de manière fonctionnelle MX est LD1/MMHA1\$MX\$HA\$phsAHar(7)

- Référence à un FCDA:
<LDName>/<LNName>\$<FC>\$<DataObjectName>[\$<SubDataObjectName>[(.<NumArrayElement>)]\$...]]\$<DataAttributeName>[(.<NumArrayElement>)]\$<SubDataAttributeName>[\$...]]

Exemple: le datRef représentant la référence ACSI LDevice/MHA11.HA.phsAHar(7).cVal.mag.f est LD1/MMHA1\$MX\$HA\$phsAHar(7)\$cVal\$mag\$f

8.3.3.3 PhyComAddr

8.3.3.3.1 PhyComAddr pour la communication Couche 2

Le PhyComAddr, utilisé dans le contexte de la communication Couche 2 par différents blocs de commande, est mappé vers la structure définie au Tableau 4.

Tableau 4 – Structure PhyComAddr pour la communication Couche 2

Nom de composant	Type de données	o/f	Commentaires
Addr	OCTET STRING	o	La longueur est de 6 octets et contient la valeur de l'adresse Media Access Control (MAC) de destination à laquelle le message GOOSE est à envoyer. Il doit s'agir d'une adresse Ethernet dont le bit multicast est défini sur TRUE.
PRIORITY	INT8U	o	La plage de valeurs doit être limitée de 0 à 7.
VID	INT16U	o	La plage de valeurs doit être limitée de 0 à 4 095.
APPID	INT16U	o	Comme défini dans l'IEC 61850-8-1:2011+AMD1:2018 CSV, Annexe C

8.3.3.3.2 PhyComAddr pour la communication UDP/IP

Le PhyComAddr, utilisé dans le contexte de la communication UDP/IP par différents blocs de commande, est mappé vers la structure définie au Tableau 5.

Tableau 5 – PhyComAddr pour la communication UDP/IP

Nom de composant	Type de données	o/f	Commentaires
Addr	OCTET STRING	o	La longueur variable de la chaîne OCTET STRING est 16 octets. Elle doit utiliser une longueur de 4 octets pour coder une adresse IPv4, alors qu'elle doit utiliser une longueur de 16 octets pour coder une adresse IPv6. Addr représente l'adresse de destination UDP/IP
isIPv6	BOOLEAN	o	Doit être TRUE si Addr est une adresse IPv6. Doit être FALSE si un IPv4 est spécifié
PRIORITY	INT8U	o	La plage de valeurs doit être limitée de 0 à 7. Cette valeur concerne la priorité Layer-2 (par des commutateurs locaux sur le LAN de l'éditeur, par exemple). La valeur n'est pas garantie être acheminée par un routeur.
VID	INT16U	o	La plage de valeurs doit être limitée de 0 à 4 095. La valeur zéro (0) est la valeur par défaut et indique qu'aucun VLAN 802.1 n'est spécifié. Cette valeur concerne l'ID Layer-2 (par des commutateurs locaux sur le LAN de l'éditeur, par exemple). La valeur n'est pas garantie être acheminée par un routeur.
APPID	Unsigned16	o	Comme défini dans l'IEC 61850-8-1:2011+AMD1:2018 CSV, Annexe C
gwAddr	OCTET-STRING	o	La longueur variable de la chaîne OCTET STRING est 16 octets. Elle doit utiliser une longueur de 4 octets pour coder une adresse IPv4, alors qu'elle doit utiliser une longueur de 16 octets pour coder une adresse IPv6. GwAddr représente l'adresse de la première passerelle de saut.
tOS	INT8U	o	Spécifie le champ TOS de l'en-tête IPv4 ou le champ Traffic Class de l'en-tête IPv6 La plage de valeurs doit être limitée de – 0 à 2 ⁸ -1 si IPv6 est FALSE. – 0 à 2 ⁸ -1 si IPv6 est TRUE

Nom de composant	Type de données	o/f	Commentaires
IGMPSrc	OCTET-STRING		La longueur variable de la chaîne OCTET STRING est 16 octets. Elle doit utiliser une longueur de 4 octets pour coder une adresse IPv4, alors qu'elle doit utiliser une longueur de 16 octets pour coder une adresse IPv6. IGMPSrc représente l'adresse src unicast à utiliser pour l'abonnement IGMPv3.

8.3.3.4 ServiceError

8.3.3.4.1 Généralités

Il existe plusieurs possibilités de réponses de service négatives ACSI, en fonction de la couche dans laquelle se produit l'erreur. Le Tableau D.2 synthétise les différents types d'erreurs.

8.3.3.4.2 Erreurs XMPP

Selon XMPP, les erreurs irrécupérables se produisant pendant la configuration du flux ou la transmission de strophe sont indiquées par un message `<stream:error/>` contenant un élément enfant qualifié par l'espace de nommage `urn:ietf:params:xml:ns:xmpp-streams`. Cet espace de nommage et la liste complète des conditions d'erreur sont spécifiés dans le RFC 6120.

Selon XMPP, les erreurs irrécupérables se produisant pendant la transmission ou le traitement des strophes (iq ou message) sont indiquées par une strophe de même type (iq, message) `type="error"` contenant un élément enfant `<error/>`, qualifié par l'espace de nommage `urn:ietf:params:xml:ns:xmpp-stanzas`. Cet espace de nommage et la liste complète des conditions d'erreur sont spécifiés dans le RFC 6120.

8.3.3.4.3 Erreurs de contexte d'association

Les erreurs qui se sont produites pendant la gestion du contexte d'association doivent être mappées vers les strophes XMPP comme indiqué dans l'IEC 62351-4.

8.3.3.4.4 Erreurs de sécurité

Les erreurs déclenchées par le mécanisme de sécurité de bout en bout doivent être mappées vers les strophes XMPP comme indiqué dans l'IEC 62351-4.

8.3.3.4.5 Erreurs de mapping

Lorsqu'un client XMPP reçoit une strophe `<iq/>` ou `<message/>` contenant des données utiles conformes, mais que le mapping sur XMPP n'est pas conforme au Tableau D.1 (il n'utilise pas la bonne strophe ou le bon type), il doit répondre par une strophe `<message/>` de `type="normal"` contenant un `rejectPDU` avec `rejectReason= unrecognized-service`.

8.3.3.4.6 Erreurs de rejet

Si un client formule une demande de service ACSI sur un élément de service (voir le Tableau D.1) qui n'est pas prise en charge ou si le serveur ne peut pas reconnaître la requête et le `confirmed-RequestPDU` formé, le serveur doit renvoyer un `<rejectPDU/>`.

L'indication de prise en charge d'un élément de service doit être déclarée dans les PIC/PIXIT et dans l'élément `<servicesSupportedCalled/>` de la réponse Associate (mappée vers un `initiate-ResponsePDU`).

Si le service n'est pas pris en charge (l'élément approprié dans le `servicesSupportedCalled` est FALSE, par exemple) et que le client émet un `confirmed-RequestPDU` bien formé de ce type de service, le serveur doit renvoyer un rejet, comme spécifié ci-dessous. Un rejet similaire doit être envoyé lorsqu'un client ou un serveur détecte qu'une PDU de service ne respecte pas les règles de compatibilité définies à l'Article 10.2.2.1.4:

`originalInvokeID`: doit être présent
`rejectReason`: doit être une raison de `confirmed-requestPDU` avec une valeur de `unrecognized-service`

Si un PDU incorrectement formé est reçu, le contenu du rejet doit être défini localement.

8.3.3.4.7 Erreurs de service ACSI

Les erreurs se produisant dans le traitement d'un service pris en charge sont renvoyées au client par l'intermédiaire d'un type `<confirmed-ErrorPDU/>` ou d'un autre type de PDU dédié aux erreurs.

Ces erreurs sont définies dans l'article relatif au mapping de chaque service. Toutefois, les articles suivants donnent certaines définitions partagées par plusieurs services.

8.3.3.4.8 Services ACSI mappés sur `GetNameList`

Si une demande de service mappée sur `<getNameList/>` ou `<extendedGetNameList/>` indique des spécifications `<objectClass/>` et `<objectScope/>` conflictuelles (voir le Tableau 6), un `<confirmed-ErrorPDU/>` doit être renvoyé. Le `<serviceError/>`, dans le `<confirmed-ErrorPDU/>`, doit être `<errorClass/><service/>` avec un `errorCode=<object-constraint-conflict/>`.

Tableau 6 – Conflit de `GetNameList` pour un `objectClass` et un `objectScope` IEC 61850

ObjectClass	objectScope
	Valeurs de paramètre
domain	domainSpecific
journal	aaSpecific
domain	aaSpecific

Le Tableau 7 définit le mapping des `ServiceError` ACSI sur l'élément `<serviceError/>` pour les services ACSI utilisant `GetNameList`.

Tableau 7 – Mappings d'erreur de service pour les services ACSI utilisant `GetNameList`

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
parameter-value-inconsistent	service	object-constraint-conflict	
failed-due-to-communications-constraint	Toutes les classes	Tous les codes d'erreur non mappés	

Pour les autres demandes qui sont traitées et pour lesquelles aucun objet correspondant n'est localisé, une réponse+ doit être renvoyée avec le `<listOfIdentifiers/>` égal à NULL et `moreFollows` défini sur FALSE.

NOTE Cette response+ vide peut se produire, par exemple, lorsque <extendedGetNameList/> est utilisé et que l'élément <domainSpecific/> contient une référence, dans un dispositif logique existant, à un objet qui n'existe pas ou pour lequel le client ne dispose d'aucun privilège d'accès.

8.3.3.4.9 Services ACSI mappés sur un service de lecture

Le présent paragraphe porte sur les erreurs communes partagées par les services lorsque le message de demande est un <confirmed-RequestPDU/> contenant un élément <read/>.

Toutefois, la référence de l'objet à lire peut être incluse dans la demande sous la forme d'un élément <variableSpecification/> (cas de GetDataValues, par exemple) ou peut être un ensemble de données côté serveur, référencé sous la forme d'un élément <variableListName/> (cas de GetDataSetvalues). Dans ce cas, les objets dont l'accès peut générer les erreurs définies dans le présent article sont ceux référencés en tant que membres de l'ensemble de données.

Les erreurs peuvent être renvoyées en tant que Response+ ou qu'élément <confirmed-ErrorPDU/> contenant un <serviceError/>.

Les définitions des erreurs possibles suivantes qui peuvent être retournées sont:

- Si la demande génère une réponse qui dépasse la taille de PDU négociée, un <confirmed-ErrorPDU/> doit être renvoyé et le <serviceError/> doit être <errorClass/><service/> avec un errorCode=<pdu-size/>. Celui-ci doit être mappé au ServiceError ACSI failed-due-to-communications-constraint.
- Si le client ne dispose des privilèges d'accès à un objet à lire, une Response+ doit être renvoyée en indiquant un <failure/> avec DataAccessError=<object-access-denied/> ou DataAccessError=<object-non-existent/>.
- Si un objet à lire n'existe pas, une Response+ doit être renvoyée en indiquant un <failure/> avec DataAccessError=<object-non-existent/>.
- Si l'accessibilité d'un objet à lire dépend de la valeur d'un autre objet (valeurs du groupe de réglage, par exemple), une Response+ doit être renvoyée en indiquant un <failure/> avec DataAccessError=<temporarily-unavailable/>.

Voir le Tableau 8 pour les mappings d'erreurs de service de lecture.

Tableau 8 – Mappings d'erreurs de service de lecture

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
failed-due-to-communications-constraint	service	pdu-size	.
failed-due-to-communications-constraint	Toutes les classes	Tous les codes d'erreur non mappés	
Erreur de service ACSI	Response+		Commentaires
	DataAccessError		
access-violation	object-access-denied		
instance-not-available	object-non-existent		
failed-due-to-communications-constraint	Tous les autres DataAccessErrors		
access-not-allowed-in-current-state	temporarily-unavailable		

8.3.3.4.10 Services ACSI mappés sur un service d'écriture

Le présent paragraphe porte sur les erreurs communes partagées par les services lorsque le message de demande est un <confirmed-RequestPDU/> contenant un élément <write/>.

Toutefois, la référence de l'objet à écrire peut être incluse dans la demande sous la forme d'un élément `<variableSpecification/>` (cas de `SetDataValues`, par exemple) ou peut être un ensemble de données côté serveur, référencé sous la forme d'un élément `<variableListName/>` (cas de `SetDataSetValues`). Dans ce cas, les objets dont l'accès peut générer les erreurs définies dans le présent article sont ceux référencés en tant que membres de l'ensemble de données.

Les erreurs peuvent être renvoyées en tant que `Response+` ou qu'élément `<rejectPDU/>`. Une `Response+` contient une séquence de résultats qui indiquent le succès ou l'échec correspondant aux objets individuels à lire.

Les définitions des erreurs possibles suivantes qui peuvent être retournées sont:

- Une demande qui dépasse la taille de PDU négociée doit générer un `<rejectPDU/>` avec un `rejectPDUType=<confirmed-requestPDU/>` et un `rejectCode=<value-out-of-range/>`. Celui-ci doit être mappé au `ServiceError ACSI failed-due-to-communications-constraint`.
- Si le client ne dispose des privilèges d'accès à un objet à écrire, une `Response+` doit être renvoyée en indiquant un `<failure/>` avec `DataAccessError=<object-access-denied/>` ou `DataAccessError=<object-non-existent/>`.
- Si un objet à écrire n'existe pas, une `Response+` doit être renvoyée en indiquant un `<failure/>` avec `DataAccessError=<object-non-existent/>`.
- Si un objet à écrire a été réservé par un autre client (un `ControlBlock` ou une structure de commande, par exemple) ou si l'accès en écriture/lecture de l'objet dépend de la valeur d'un autre objet (valeurs du groupe de réglage, par exemple), une `Response+` doit renvoyer un `<failure/>` en indiquant `dataAccessError=<temporarily-unavailable/>`.
- Plusieurs questions sont liées aux données fournies dans la demande comparées aux définitions de type de données prévues côté serveur:
 - Le type de données de taille fixe est prévu et les données fournies ne correspondent pas à la taille définie: un `<failure/>` dans la `Response+` indique `dataAccessError=<type-inconsistent/>`.
 - Les types de données de taille de longueur variable prévus et les données fournies ne dépassent pas la taille maximale définie: un `<failure/>` dans la `Response+` indique `dataAccessError=<type-inconsistent/>`.
 - Les types de données de taille de longueur variable, et extensible, prévus (`BITSTRINGS`, par exemple), mais dont les données fournies ne dépassent pas la taille maximale définie: aucun `<failure/>` ne peut être généré.
- Une tentative d'écriture d'un objet totalement en lecture seule ou si l'un au moins de ces sous-composants est en lecture seule (un composant de structure ou un élément d'ensemble, par exemple) doit résulter en une `Response+` indiquant un `dataAccessError=<object-access-denied/>`.
- Les `ENUMERATIONS` sont mappées à la même taille d'un `Integer DataType`. Si la valeur est au-delà de l'ensemble de valeurs défini admis pour `ENUMERATION`, l'application IEC 61850 doit renvoyer `DataAccessError=<object-value-invalid/>`.

NOTE Cela s'applique aux plages de valeurs énumérées positives définies et négatives prises en charge.

Voir le Tableau 9 pour les mappings d'erreurs de service d'écriture.

Tableau 9 – Mappings des erreurs de service d'écriture

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
failed-due-to-communications-constraint	Toutes les classes	Tous les codes d'erreur non mappés	La liste exhaustive des classes et codes d'erreur est définie par le type complexe ServiceError dans le XSD de l'Annexe G
Erreur de service ACSI	Response+ MMS		Commentaires
	DataAccessError		
access-violation	object-access-denied		
instance-not-available	object-non-existent		
instance-locked-by-other-client	temporarily-unavailable		
type-conflict	type-inconsistent		
parameter-value-inconsistent	object-value-invalid		
failed-due-to-communications-constraint	Tous les autres DataAccessErrors		
access-not-allowed-in-current-state	temporarily-unavailable		
Erreur de service ACSI	Rejet		Commentaires
	Type PDU du rejet	Code de rejet	
parameter-value-inconsistent	pdu-error	Invalid-pdu	

8.3.3.4.11 Services ACSI mappés sur FileDirectory

Le présent article porte sur les erreurs communes partagées par les services dont le message de demande est un <confirmed-RequestPDU/> incluant un élément <FileDirectory/>, c'est-à-dire les services ACSI GetServerDirectory (lors de la demande de FILE) et les GetFileAttributeValues.

- Une demande qui spécifie un répertoire qui n'existe pas doit renvoyer un <confirmed-ErrorPDU/>. Le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><file/> avec un errorCode=<file-non-existent/>.
- Une demande qui spécifie un FileDirectory pour lequel le client ne dispose pas des privilèges d'accès doit renvoyer un <confirmed-ErrorPDU/>. Le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><file/> avec un errorCode=<file-access-denied/> ou <errorClass/><file/> avec un errorCode=<file-non-existent/>.
- Une demande qui spécifie un FileSpecification qui ne peut pas être interprété doit renvoyer un <confirmed-ErrorPDU/>. Le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><file/> avec un errorCode=<filename-syntax-error/>.
- Une demande qui spécifie un <continueAfter/> avec un Filename qui n'existe pas doit renvoyer un <confirmed-ErrorPDU/>. Le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><file/> avec un errorCode=<file-non-existent/>.

Voir le Tableau 10 pour les mappings d'erreurs de service GetFileAttributeValues.

Tableau 10 – Mappings des erreurs de service GetFileAttributeValues

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	file	file-non-existent	
access-violation	file	file-access-denied	
parameter-value-inappropriate	file	filename-syntax-error	
failed-due-to-communications-constraint	Toutes les classes	Tous les codes d'erreur non mappés	

8.3.3.5 EntryID

Le type EntryID ACSI doit être mappé en type ACSI OCTET STRING de longueur fixe de 8 octets (voir 8.3.2.4). Le contenu est spécifique au serveur IEC 61850. Il est recommandé que le format du contenu soit documenté comme faisant partie de la déclaration PIXIT de la mise en œuvre.

La valeur réservée d'EntryID est zéro (0), c'est-à-dire la valeur `<octet-string>0000000000000000</octet-string>`.

8.3.3.6 PACKED LIST

Un type PACKED LIST doit être mappé en tant que type ACSI BIT STRING (voir 8.3.2.1). La taille minimale du `xsd:string` doit être déterminée par le nombre de bits exigé pour coder les membres du PACKED LIST. L'ordre des bits, dans le `xsd:string`, doit être tel que le premier membre du PACKED LIST doit être mappé à Bit(0). Les autres membres doivent être mappés aux bits consécutifs dans l'ordre spécifié par l'IEC 61850-7-2.

Exemple: `quality` est une chaîne binaire de longueur de 13 bits `<bit-string>0000000000000</bit-string>`

Le PACKED LIST doit prendre en charge au moins le nombre de bits exigé pour représenter la valeur maximale définie énumérée. Les bits qui sont retournés en excès de ce nombre de bits peuvent être ignorés.

Le bit 0 doit être le bit le plus à gauche (le plus significatif) du premier octet. Le bit 7 doit être le bit le plus à droite (le moins significatif) du premier octet. Le bit 8 doit être le bit le plus à gauche (le plus significatif) du deuxième octet. Le bit 15 doit être le bit le plus à droite (le moins significatif) du deuxième octet. Cela doit être appliqué de cette façon dans les octets suivants.

Certains cas spéciaux sont mappés individuellement et ne sont pas conformes à la règle générale. Il s'agit du type Timestamp (horodatage) (spécifié en 8.3.3.7), du type Quality (spécifié en 8.3.4), du type TriggerConditions (spécifié en 8.3.3.9) et du type ReasonForInclusion (spécifié en 8.3.3.10 et en 8.3.3.11).

8.3.3.7 Timestamp

Timestamp doit être mappé à un élément avec `name="utc-time"` et `type="OCTET_STRING"`, avec une taille fixe de 8 octets (sauf si un autre mapping pour timestamp est spécifié dans le présent document dans un cas dédié).

Exemple: `<utc-time>017215E00000001F</utc-time>`

Les quatre premiers octets codent l'attribut `SecondSinceEpoch`.

Les octets du cinquième au septième codent l'attribut `FractionOfSecond`.

Les indicateurs de qualité sont situés dans le huitième octet. Ces indicateurs sont définis au Tableau 11.

Tableau 11 – Codage de TimeQuality selon l'IEC 61850-7-2

Numéro de bit	Valeur	Signification
0		Seconde intercalaire connue
1		Défaillance d'horloge
2		Horloge non synchronisée
3-7		Précision du temps des fractions de seconde
	00000	0 bit de précision
	00001	1 bit de précision
	00010	2 bits de précision
	00011	3 bits de précision
	00100 - 11000	Valeur entière du nombre de bits de précision
	11001- 11110	Non valide
	11111	non spécifié

Le bit 0 doit être le bit le plus significatif de l'octet 7. Le bit 7 doit être le bit le moins significatif de l'octet 7.

Exemple: dans `<utc-time>017215E00000001F</utc-time>`, la valeur 1F hexadécimale de l'octet 7 (représentant 00011111 binaire) signifie: pas de Seconde intercalaire connue, pas de Défaillance d'horloge, Horloge synchronisée, Précision du temps non spécifiée.

8.3.3.8 EntryTime

EntryTime doit être mappé à un élément avec name="binary-time" et type = "TimeOfDay".

- TimeOfDay est une restriction d'OCTET_STRING. La longueur de la valeur doit être de six (6) octets.
- Les quatre premiers octets doivent contenir une valeur indiquant le nombre de millisecondes depuis minuit de la date actuelle. Les deux derniers octets contiennent le jour relatif depuis le 1 janvier 1984.

Exemple: `<binary-time>0316E6032DD2</ binary-time >` est le codage de 14 h 23 min 52 s et 232 ms le 12 février 2016. "0316E603" est le nombre de millisecondes depuis minuit, 2DD2 est le nombre de jours depuis le 1 janvier 1984.

8.3.3.9 TriggerConditions

Les valeurs pour TriggerConditions sont codées sous forme de PACKED LIST, c'est-à-dire en tant que BIT STRING.

Bit(0) est réservé, de sorte que la taille minimale doit être de six (6) bits. Les bits supplémentaires sont réservés pour des versions futures de la norme. Les bits sont assignés comme indiqué au Tableau 12.

Tableau 12 – Codage de TriggerConditions

Condition de déclenchement	Position de bit MMS
Réservé (réservé pour assurer la compatibilité ascendante avec UCA 2.0)	0
data-change	1
quality-change	2
data-update	3
integrity	4
general-interrogation	5

NOTE Une extension future des TriggerConditions prend en compte la définition du Bit 6 comme étant réservée afin de maintenir un certain alignement entre le type TriggerConditions, le type ReasonForInclusionInReport et le type ReasonForInclusionInLog.

8.3.3.10 Raison de l'inclusion dans des rapports (ReasonForInclusionInReport)

Les valeurs pour ReasonForInclusionInReport sont codées sous forme de PACKED LIST, c'est-à-dire en tant que BIT STRING.

Bit(0) est réservé, de sorte que la taille minimale doit être de six (6) bits. Les bits supplémentaires sont réservés pour des versions futures de la norme. Les bits sont assignés comme indiqué au Tableau 13.

Tableau 13 – Codage de ReasonForInclusionInReport

Raison de l'inclusion	Position de bit MMS
Réservé (réservé pour assurer la compatibilité ascendante avec UCA 2.0)	0
data-change	1
quality-change	2
data-update	3
integrity	4
general-interrogation	5

NOTE Une extension future de ReasonForInclusionInReport prend en compte la définition du Bit 6 comme étant réservée afin de maintenir un certain alignement entre le type TriggerConditions, le type ReasonForInclusionInReport et le type ReasonForInclusionInLog.

8.3.3.11 Raison de l'inclusion dans des journaux (ReasonForInclusionInLog)

Les valeurs pour ReasonCode sont codées sous forme de PACKED LIST, c'est-à-dire en tant que BIT STRING.

Bit(0) est réservé, de sorte que la taille minimale doit être de sept (7) bits. Les bits supplémentaires sont réservés pour des versions futures de la norme. Les bits sont assignés comme indiqué au Tableau 14.

Tableau 14 – Codage de ReasonForInclusionInLog

Raison de l'inclusion	Position de bit MMS
Réservé (réservé pour assurer la compatibilité ascendante avec UCA 2.0)	0
data-change	1
quality-change	2
data-update	3
integrity	4
general-interrogation	5
application-trigger	6

8.3.3.12 RCBReportOptions

Les valeurs pour RCBReportOptions sont codées sous forme de PACKED LIST, c'est-à-dire en tant que BIT STRING. La taille minimale du BITSTRING doit être de dix (10) bits. Les bits supplémentaires sont réservés pour des versions futures de la norme. Les bits sont assignés comme indiqué au Tableau 15.

Tableau 15 – Codage de RCBReportOptions

Valeur ACSI de BRCState	Position de bit MMS
Réservée	0
sequence-number	1
report-time-stamp	2
reason-for-inclusion	3
data-set-name	4
data-reference	5
buffer-overflow	6
entryID	7
conf-revision	8
segmentation	9

8.3.3.13 LCBLogEntryOptions

Non mappé. Le présent Article oblige les reasonCodes à toujours apparaître dans le journal.

8.3.3.14 SVMessageOptions

Les valeurs pour SVMessageOptions sont codées sous forme de PACKED LIST, c'est-à-dire en tant que BIT STRING. La taille minimale du BITSTRING doit être de sept (7) bits. Les bits supplémentaires sont réservés pour des versions futures de la norme. Les bits sont assignés comme indiqué au Tableau 16.

Tableau 16 – Codage de SVMessageOptions

Valeur ACSI de SVMessageOptions	Position de bit MMS
refresh-time	0
réservée	1
sample-rate	2
data-set-name	3
security	4
sample-mode	5
synch-source-identity	6

8.3.3.15 CheckConditions

Les valeurs pour CheckConditions sont codées sous forme de PACKED LIST, c'est-à-dire en tant que BIT STRING. La taille minimale du BITSTRING doit être de deux (2) bits. Les bits supplémentaires sont réservés pour des versions futures de la norme. Les bits sont assignés comme indiqué au Tableau 17.

Tableau 17 – Codage de CheckConditions

Valeur ACSI de CheckConditions	Position de bit MMS
synchrocheck	0
interlockCheck	1

8.3.3.16 Originator

Le type Originator doit être mappé à un élément <structure> représentant l'Originator.

Exemple:

```
<structure>
  <integer>2</integer>
  <octet-string>53544154494F4E</octet-string>
</structure>
```

8.3.3.17 Currency

Le type Currency doit être mappé à une chaîne visible dont la longueur maximale est de trois (3) caractères. Le contenu de la chaîne doit être le code de currency en caractères ISO 4217 3.

8.3.4 Mapping de type d'attribut de données de qualité communes spécifié dans l'IEC 61850-7-2

La définition, présentée dans l'IEC 61850-7-2, de l'attribut de données communes Quality, spécifie une combinaison de valeurs PACKED LIST, CODED ENUM et BOOLEAN. Le mapping de cet attribut de données doit être comme indiqué dans le présent Article.

Une valeur IEC 61850-7-2 du type quality (qualité) doit être représentée par un BIT STRING.

Les valeurs de bit doivent être telles que définies au Tableau 18. Le Bit(0) est le bit le plus à gauche, et tous les bits doivent être acheminés.

Des versions futures de la norme peuvent ajouter des bits supplémentaires. Le nombre maximal autorisé de bits doit être de 15.

Tableau 18 – Codage de quality (qualité) de l'IEC 61850-7-2

Bit(s)	IEC 61850-7-3		Bit-String	
	Attribute name	Attribute value	Value	Default
0-1	Validity	Good	0 0	0 0
		Invalid	0 1	
		Réservée	1 0	
		Questionable	1 1	
2	Overflow		TRUE	FALSE
3	OutofRange		TRUE	FALSE
4	BadReference		TRUE	FALSE
5	Oscillatory		TRUE	FALSE
6	Failure		TRUE	FALSE
7	OldData		TRUE	FALSE
8	Inconsistent		TRUE	FALSE
9	Inaccurate		TRUE	FALSE
10	Source	Process	0	0
		Substituted	1	
11	Test		TRUE	FALSE
12	OperatorBlocked		TRUE	FALSE

8.4 Mapping général des valeurs de données dans des données utiles XML

Le présent article s'applique aux données définies dans l'IEC 61850-7-3 et aux blocs de commande définis dans l'IEC 61850-7-2.

Tous les services ACSI utilisés pour acheminer les valeurs de données (pour le réglage, l'obtention et la génération de rapport) sont mappés sur des éléments XML à l'aide de complexType Data. Le présent article explique comment instancier ce type Data dans de véritables données utiles.

Si les données acheminées sont un élément BasicType, la valeur est mappée vers un seul élément, comme indiqué au Tableau 3 (<bit-string>0111110010</bit-string>, par exemple).

Si les données acheminées sont un composant contenant potentiellement (et de manière récursive) des sous-composants, le composant et les sous-composants doivent être mappés vers un élément <structure/> contenant (potentiellement et de manière récursive également) des éléments BasicType et/ou des éléments <structure/>. De plus, tous les éléments inclus dans le composant doivent être individuellement encapsulés dans un élément <Data/>.

Le Tableau 19 donne des exemples de codage de valeurs renvoyées par un service ACSI GetDataValues appliqué respectivement sur le FCD "CONTROL/GGIO1\$ST\$Ind3" et le FCDA "CONTROL/GGIO1\$ST\$Ind3\$stVal".

Tableau 19 – Exemples de codage de valeurs de données

Paramètre d'entrée de la demande GetDataValues	Valeur de l'élément <success/> dans la réponse GetDataValues
FCD CONTROL/GGIO1\$ST\$Ind3	<pre> <structure> <Data> <boolean>false</boolean> </Data> <Data> <bit-string>000000000000</bit-string> </Data> <Data> <utc-time>017215E30000001F</utc-time> </Data> </structure> </pre>
FCDA CONTROL/GGIO1\$ST\$Ind3\$stVal	<pre> <boolean>true</boolean> </pre>

8.5 Comportement étendu pour l'optimisation de la bande passante

Comme indiqué à l'Article 1, le présent SCSM utilise généralement les mêmes définitions MMS abstraites pour les messages que celles de l'IEC 61850-8-1. Toutefois, les définitions MMS comportent certains inconvénients bien connus que le présent SCSM peut surmonter.

Les services ACSI concernés sont GetLogicalDeviceDirectory, GetLogicalNodeDirectory et GetDataDirectory. Pour ces services, un codage XML direct des messages MMS abstraits génère des messages Response+ volumineux dont la plupart des informations ne sont pas exigées, et doivent donc être filtrés côté client.

L'idée consiste à maintenir ce fonctionnement comme étant celui par défaut, mais d'en proposer éventuellement un étendu, dans lequel Response+ contient uniquement les informations obligatoires. Le choix entre le comportement par défaut et le comportement étendu est fait au moment de l'association, au moyen de l'échange d'une liste de services pris en charge contenant éventuellement la valeur extendedGetNameList. Si le comportement étendu est choisi, les services ACSI concernés doivent utiliser le choix extendedGetNameList plutôt que leur choix normal (getNameList pour GetLogicalDeviceDirectory et GetLogicalNodeDirectory, et getVariableAccessAttributes pour GetDataDirectory, respectivement) dans leurs éléments ConfirmedServiceRequest et ConfirmedServiceResponse. La structure des éléments getNameList et celle d'extendedGetNameList sont strictement identiques.

Les mappings par défaut et étendus détaillés sont spécifiés dans l'article relatif à chacun des services ACSI affectés.

9 Modèle de la classe Server

9.1 Généralités

NOTE 1 Dans le présent article, et dans les articles suivants définissant la manière dont les services ACSI sont mappés sur les messages XML transmis sur XMPP, la partie de contexte d'association et les éléments sécurisés sont ignorés. Pour bien comprendre la présentation complète des messages transmis dans les flux XMPP, voir 5.2.2, 7 et 8.

NOTE 2 Dans le présent document, le terme "taille de PDU" fait référence à la taille de la PDU de service telle que définie en 5.2.2. Cette taille est négociée pendant le service Associate (voir 10.2.2.1).

Une mise en œuvre IEC 61850-8-2 doit prendre en charge un ou plusieurs ServiceAccessPoints dédié(s) à la communication XMPP. Chaque point d'accès doit être défini par un ou plusieurs JID.

9.2 GetServerDirectory

9.2.1 Généralités

Le service GetServerDirectory doit être mappé différemment selon la classe ACSI demandée: LOGICAL-DEVICE et FILE.

Classe LOGICAL-DEVICE

La demande GetServerDirectory mappe vers un élément XML <confirmed-RequestPDU/> contenant un élément <getNameList/> dans lequel la valeur de l'élément <objectClass/> doit être <domain/>. La Figure 15 donne la structure du message XML correspondant aux données utilisateur (c'est-à-dire sans élément de sécurité qui va encapsuler le message utilisateur dans le message réel sur le câble).

La réponse GetServerDirectory mappe vers un élément XML <confirmed-ResponsePDU/> contenant un élément <GetNameList/>. La Figure 16 donne la structure du message XML correspondant aux données utilisateur.

Les erreurs de service (ServiceError) ACSI sont détaillées en 8.3.3.4.8.

Le Tableau 20 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés dans les structures de demande et de réponse, respectivement.

Le service GetServerDirectory peut être restreint par la limitation de taille de PDU négociée. Dans ce cas, la réponse peut renvoyer uniquement un sous-ensemble de noms de dispositif logique et un indicateur (moreFollows) indiquant la nécessité de demander le sous-ensemble suivant. Le client doit ensuite transmettre une autre demande, spécifiant cette fois-ci la position dans la liste à laquelle il convient que le serveur continue la recherche de nom. En raison de cette contrainte, le mapping du service ACSI est en fait une séquence de demandes/réponses, jusqu'à moreFollows=FALSE dans la réponse.

Tableau 20 – Mapping de GetServerDirectory (DISPOSITIF LOGIQUE) ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNameList/>		Voir la structure de la demande XML à la Figure 15
<objectClass/> = LOGICAL-DEVICE	<objectClass/>	<domain/>	Valeur fixe
	<continueAfter/>	<LDNameX>	Facultatif: nom du dernier LD renvoyé dans la précédente Response si elle contient moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNameList/>		Voir la structure de la réponse XML à la Figure 16

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Reference[0..n]	<listOfIdentifier/>	<pre> <Identifier>LDName1</Identifier> <Identifier>LDName2</Identifier> ... </pre>	Response+ contient uniquement les dispositifs logiques qui peuvent être affichés par l'application client en fonction de ses privilèges d'accès. Pour les mises en œuvre revendiquant la conformité à l'IEC 61850, Response+ ne doit pas être NULL. La liste des identifiants doit être ordonnée d'un point de vue lexicographique (selon le codage ASCII des caractères)
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Voir 8.3.3.4.8	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>78531</invokeID>
  <ConfirmedServiceRequest>
    <getNameList>
      <extendedObjectClass>
        <objectClass><domain/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <vmdSpecific/>
      </objectScope>
    </getNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 15 – Structure XML de GetServerDirectory-Request (LD)

```

<confirmed-ResponsePDU>
  <invokeID>78531</invokeID>
  <ConfirmedServiceResponse>
    <getNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>CONTROL</Identifier>
        <Identifier>TEST</Identifier>
      </listOfIdentifier>
    </getNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 16 – Structure XML de GetServerDirectory-Response (LD)

Classe FILE

La demande GetServerDirectory mappe vers un élément XML <confirmed-RequestPDU/> contenant un élément <fileDirectory/>. La Figure 17 donne la structure du message XML correspondant.

La réponse GetServerDirectory mappe vers un élément XML <confirmed-ResponsePDU/> contenant un élément <fileDirectory/>. La Figure 18 donne la structure du message XML correspondant.

Les erreurs de service (ServiceError) ACSI sont détaillées en 8.3.3.4.11

Le Tableau 21 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés dans les structures de demande et de réponse, respectivement.

Le service GetServerDirectory, s'il est restreint par la limitation de taille de PDU négociée, peut segmenter le processus en retournant un sous-ensemble de noms de fichier et un indicateur (moreFollows) indiquant qu'il est nécessaire de demander le sous-ensemble suivant. Le client doit ensuite transmettre une autre demande, spécifiant cette fois-ci la position dans la liste à laquelle il convient que le serveur continue la recherche de nom. En raison de cette contrainte, le mapping du service ACS I est en fait une séquence de demandes et réponses FileDirectory (jusqu'à moreFollows=FALSE).

Si le nom de fichier (élément <fileSpecification/>) n'est pas présent dans la demande ou si le caractère générique "*" est utilisé, le serveur répondeur doit retourner les noms de fichier présents dans le répertoire racine (des noms de fichier supplémentaires peuvent aussi être retournés). Le caractère générique "*" ne doit pas être combiné dans une chaîne en tant que caractère générique pour l'extension de nom, mais toujours comme un caractère unique et, par conséquent, est équivalent à une demande dans laquelle le nom de fichier est vide, "/" ou "\".

Pour les serveurs qui ont des dispositifs logiques, 23.1 spécifie le répertoire racine. Les noms des fichiers retournés doivent être les noms des fichiers présents dans le répertoire LD. Pour les serveurs qui n'ont pas de dispositifs logiques (par exemple, éventuellement un serveur de fichiers), le répertoire racine doit être défini localement.

Tableau 21 – Mapping de GetServerDirectory (FICHIER) ACS I

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <fileDirectory/>		Voir la structure de la demande XML à la Figure 17
<objectClass/> = FILE	<fileSpecification/>	<GraphicString>*</GraphicString>	Facultatif
	<continueAfter/>	FileName	Facultatif (voir le commentaire pour le paramètre Reference[0..n])
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <fileDirectory/>		Voir la structure de la réponse XML à la Figure 18
Reference[0..n]	<listOfDirectoryEntry/>	Chaque DirectoryEntry donne le FileName et les attributs d'un fichier	Le FileName contient un élément <GraphicString/> si sa taille est inférieure à 64 caractères. Sinon, d'autres éléments <GraphicString/> de taille maximale de 64 caractères sont ajoutés
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Voir 8.3.3.4.11	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>3819321</invokeID>
  <ConfirmedServiceRequest>
    <fileDirectory>
      <fileSpecification>
        <GraphicStringMMS>*</GraphicStringMMS>
      </fileSpecification>
    </fileDirectory>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 17 – Structure XML de GetServerDirectory-Request (FICHIER)

```

<confirmed-ResponsePDU>
  <invokeID>3819321</invokeID>
  <ConfirmedServiceResponse>
    <fileDirectory moreFollows="true">
      <listOfDirectoryEntry>
        <DirectoryEntry>
          <fileName>
            <GraphicStringMMS>FileName1</GraphicStringMMS>
          </fileName>
          <fileAttributes>
            <sizeOfFile>383</sizeOfFile>
            <lastModified>201512160532-0500</lastModified>
          </fileAttributes>
        </DirectoryEntry>
        <DirectoryEntry>
          <fileName>
            <GraphicStringMMS>FileName2</GraphicStringMMS>
          </fileName>
          <fileAttributes>
            <sizeOfFile>6271</sizeOfFile>
            <lastModified>20160115092132,2-0500</lastModified>
          </fileAttributes>
        </DirectoryEntry>
      </listOfDirectoryEntry>
    </fileDirectory>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 18 – Structure XML de GetServerDirectory-Response (FICHIER)

10 Modèle d'association

10.1 Relation d'association avec les profils de communication

Les deux profils spécifiés dans ce SCSM utilisent l'association bipartite spécifiée dans l'IEC 61850-7-2 (voir le Tableau 22).

Tableau 22 – Modèle d'association/profils de communication

Profil de communication	Modèle(s) d'association ACSI pris en charge
Client/serveur	Bipartite
Synchronisation temporelle	Bipartite

10.2 Modèle d'association bipartite pour le profil de communication client/serveur

10.2.1 Établissement d'une association de bout en bout sécurisée

Le service Associate décrit à l'Article 10.2.2.1 se déroule dans le cadre de la liaison assurée en parallèle pour deux mécanismes supplémentaires: la création du contexte d'association et l'établissement d'une session sécurisée de bout en bout. Ces deux mécanismes sont définis dans l'IEC 62351-4 et ne sont donc pas décrits dans les paragraphes ci-après.

NOTE La partie de contexte d'association et les éléments de sécurité de bout en bout sont ignorés dans les messages XML décrits dans les articles suivants et dans tous les exemples du présent document.

Avant d'établir une association entre deux entités d'application, une connexion de transport sous-jacente doit être établie, c'est-à-dire que chaque point d'extrémité doit être connecté à son serveur XMPP. Si les deux points d'extrémité n'appartiennent pas au même domaine XMPP, les deux serveurs XMPP associés doivent être fédérés.

Dès que l'association a été établie, les deux points d'extrémité (l'appelant et l'appelé) doivent surveiller l'état de la communication. Si un point d'extrémité détecte une indisponibilité de communication, un abandon local doit être déclenché jusqu'au niveau d'application IEC 61850:

- Si une perte de liaison est détectée par un point d'extrémité au niveau du client XMPP (connexion TCP interrompue entre le client XMPP et le serveur XMPP, par exemple), l'abandon local doit être déclenché pour toutes les Associations utilisant cette liaison, c'est-à-dire toutes les Associations dans lesquelles ce point d'extrémité est impliqué avec le même JID.
- Si le client XMPP d'un point d'extrémité est informé par le serveur XMPP qu'un JID impliqué dans une association est "indisponible" (par l'intermédiaire du mécanisme de présence, voir 6.4), un abandon local doit être déclenché, mais uniquement pour les Associations impliquant ce JID donné.

10.2.2 Services d'association

10.2.2.1 Associate

10.2.2.1.1 Généralités

La demande, la réponse+ et la réponse- d'un service Associate ACSI doivent respectivement mapper vers les éléments <initiate-RequestPDU/>, <initiate-ResponsePDU/> et <initiate-ErrorPDU/>.

La Figure 19 et la Figure 20 donnent des exemples de structure de messages XML correspondant à la demande et à la réponse. Le Tableau 23 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés globalement sur ces messages de demande et de réponse. Toutefois, les éléments XML de ces messages ne correspondent à aucun paramètre de niveau ASCL. Ils sont plutôt décrits au Tableau 24 et au Tableau 25.

NOTE Pendant ce service, les entités appelantes et appelées échangent la liste de leurs services pris en charge dans les éléments <servicesSupportedCalling/> et <servicesSupportedCalled/>, respectivement. La négociation du comportement étendu (voir 8.5) exige que la demande et la réponse contiennent <extendedGetNameList/> comme service pris en charge.

Le mapping de la réponse- est détaillé en 10.2.2.1.2.

Tableau 23 – Mapping du service Associate ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<initiate-RequestPDU>		Voir la structure de la demande XML à la Figure 19
ServerAccessPointReference			Le JID de l'entité de destination est mappé dans l'attribut "to" de la strophe XMPP <iq/> (voir 6.3)
Paramètre d'authentification			Mappé dans les éléments de sécurité de bout en bout (voir l'Article 7)
Response+	<initiate-ResponsePDU>		Voir la structure de la réponse XML à la Figure 20
AssociationId			
Result			
Response–	<initiate-ErrorPDU>	Voir 10.2.2.1.2	
ServiceError			

Tableau 24 – Description des éléments de demande Associate

Éléments XML Initiate Request	Valeur/Plage	Description
<localDetailCalling/>		Taille de PDU proposée
<proposedMaxServOutstandingCalling/>	1 ou plus	Nombre maximal de demandes en attente envoyées
<proposedMaxServOutstandingCalled/>	1 ou plus	Nombre maximal de demandes en attente reçues
<proposedDataStructureNestingLevel/>	c1	Nombre maximal de caractères "\$" dans une référence
<proposedVersionNumber/>	doit être égal à 1	
<proposedParameterCBB/>		Ensemble de paramètres proposés. Doit être cohérent avec les fonctionnalités prises en charge par la mise en œuvre (voir les détails en 24.2.2.1.3)
<servicesSupportedCalling/>		Ensemble de services proposés. Doit être cohérent avec les fonctionnalités prises en charge et les services ACSI revendiqués par la mise en œuvre (voir les détails en 24.2.2.1.2)
c1 Doit être cinq (5) ou plus si la prise en charge de modèle de nœud logique ACSI est déclarée.		

Tableau 25 – Description des éléments de réponse Associate

Éléments XML Initiate Response	Valeur/Plage	Description
<localDetailCalled/>		Taille de PDU négociée
<negotiatedMaxServOutstandingCalling/>	1 ou plus	Nombre maximal de demandes en attente envoyées
<negotiatedMaxServOutstandingCalled/>	1 ou plus	Nombre maximal de demandes en attente reçues
<negotiatedDataStructureNestingLevel/>	c1	Nombre maximal de caractères "\$" dans une référence
<negotiatedVersionNumber/>	doit être égal à 1	
<negotiatedParameterCBB/>		Ensemble de paramètres négociés. Doit être cohérent avec les fonctionnalités prises en charge par la mise en œuvre (voir les détails en 24.2.2.1.3)

Éléments XML Initiate Response	Valeur/Plage	Description
<servicesSupportedCalled/>		Ensemble de services négociés. Doit être cohérent avec les fonctionnalités prises en charge et les services ACSI revendiqués par la mise en œuvre (voir les détails en 24.2.2.1.2)
c1 Doit être cinq (5) ou plus si la prise en charge de modèle de nœud logique ACSI est déclarée.		

```

<initiate-RequestPDU>
  <localDetailCalling>65536</localDetailCalling>
  <proposedMaxServOutstandingCalling>5</proposedMaxServOutstandingCalling>
  <proposedMaxServOutstandingCalled>5</proposedMaxServOutstandingCalled>
  <proposedDataStructureNestingLevel>7</proposedDataStructureNestingLevel>
  <mmsInitRequestDetail>
    <proposedVersionNumber>1</proposedVersionNumber>
    <proposedParameterCBB>
      str1 str2 vnam valt vlis
    </proposedParameterCBB>
    <servicesSupportedCalling>
      getNameList informationReport
    </servicesSupportedCalling>
  </mmsInitRequestDetail>
</initiate-RequestPDU>

```

Figure 19 – Structure XML d'Associate-Request

```

<initiate-ResponsePDU>
  <localDetailCalled>65536</localDetailCalled>
  <negotiatedMaxServOutstandingCalling>5</negotiatedMaxServOutstandingCalling>
  <negotiatedMaxServOutstandingCalled>5</negotiatedMaxServOutstandingCalled>
  <negotiatedDataStructureNestingLevel>7</negotiatedDataStructureNestingLevel>
  <mmsInitResponseDetail>
    <negotiatedVersionNumber>1</negotiatedVersionNumber>
    <negotiatedParameterCBB>
      str1 str2 vnam valt vlis
    </negotiatedParameterCBB>
    <servicesSupportedCalled>
      status getNameList identify read write getVariableAccessAttributes
      defineNamedVariableList getNamedVariableListAttributes deleteNamedVariableList
      obtainFile readJournal getCapabilityList fileOpen fileRead fileClose fileDelete
      fileDirectory conclude cancel extendedGetNameList
    </servicesSupportedCalled>
  </mmsInitResponseDetail>
</initiate-ResponsePDU>

```

Figure 20 – Structure XML d'Associate-Response

10.2.2.1.2 Response-

Un Associate Response- est mappé sur un <initiate-ErrorPDU> qui contient un élément <errorClass/> avec les informations suivantes:

- En cas de discordances de paramètre pour l'application prévue lorsque le parameterCBB proposé ou le serviceSupported ne permet pas à l'application IEC 61850 prévue de fonctionner correctement: l'élément <errorClass/> doit indiquer <initiate/> avec une valeur errorCode appropriée (voir le Tableau 26).
- Si le demandeur du service Associate n'est pas autorisé à établir une association en raison de privilèges de sécurité, le serveur doit émettre un élément <errorClass/><access/> avec un errorCode = <object-access-denied/>.

- Toutes les autres erreurs appartenant à un service Associate doivent être mappées au ServiceError ACSI de failed-due-to-server-constraint

Le Tableau 26 donne des mappings d'erreurs de service ACSI d'association. Les valeurs de ServiceError ACSI qui n'apparaissent pas dans le tableau ne sont pas mappées.

Tableau 26 – Mappings des erreurs de service ACSI d'association

Erreur de service ACSI	ServiceError		Commentaire
	Classe d'erreur	Code d'erreur	
parameter-value-inconsistent	initiate	version-incompatible	
failed-due-to-communications-constraint	initiate	parameter-CBB-insufficient nesting-level-insufficient max-services-outstanding-called-insufficient max-services-outstanding-calling-insufficient max-services-outstanding-calling-insufficient service-CCB-insufficient	
access-violation	access	object-access-denied	Result::=Failure
failed-due-to-server-constraint	Toutes les classes	Tous les codes d'erreur non mappés	

10.2.2.1.3 Accord de mise en œuvre

- Le numéro de version MMS doit être tel que spécifié au Tableau 96 et au Tableau 97.
- Le max_mms_pdu_size est défini comme étant le nombre maximal d'octets dans un MMSpdu codé à l'aide de la syntaxe de transfert négociée. Cette taille doit s'appliquer à tous les MMSpdu, sauf à initiate-Request PDU, initiate-Response PDU et initiate-Error PDU. Le max_mms_pdu_size doit être négocié pendant l'initiation de la connexion à l'aide des paramètres Local Detail Calling et Local Detail Called du service Associate.
- Le paramètre Local Detail Calling de la réponse Associate doit spécifier le max_mms_pdu_size négocié pour l'association d'application. Il est recommandé qu'au moins 8 192 octets soient pris en charge. Sa valeur doit être inférieure ou égale à celle du paramètre Local Detail Calling de la demande.
- Le max_mms_pdu_size négocié doit être appliqué comme suit:
 - a) Tous les MMSpdu reçus dont la longueur est inférieure ou égale au max_mms_pdu_size négocié doivent faire l'objet d'une analyse syntaxique et d'un traitement corrects.
 - b) Il convient qu'une mise en œuvre n'envoie pas de MMSpdu dont la taille dépasse le max_mms_pdu_size négocié. Si une mise en œuvre envoie un MMSpdu dont la taille dépasse le max_mms_pdu_size négocié, elle doit être préparée à recevoir un PDU de rejet. Si une mise en œuvre reçoit un MMSpdu qui dépasse le max_mms_pdu_size négocié, elle doit rejeter le MMSpdu ou accepter le MMSpdu comme si aucune violation de taille ne s'était produite, puis procéder au traitement prévu.
 - c) Si une mise en œuvre n'est pas en mesure d'envoyer une réponse de service qui serait susceptible de dépasser le max_mms_pdu_size, elle doit renvoyer un Service response- avec un <serviceError/> de <errorClass/><service/> avec un code= <pdu-size/>.
 - d) Si un MMSpdu est rejeté car il dépasse le max_mms_pdu_size négocié, une mise en œuvre doit utiliser un <rejectPDU/> de type <confirmed-requestPDU/> avec un code de rejet = <value-out-of-range/>.

10.2.2.1.4 Versionnage et compatibilité

Lors de l'établissement d'association, une mise en œuvre doit indiquer la dernière version du schéma XSD auquel elle se conforme, respectivement dans la demande, la réponse+ et la réponse- du service Associate. Les informations de versionnage sont composées de trois attributs facultatifs: version, revision et release, dont les définitions et les valeurs doivent être cohérentes avec l'Article 1.2. Ces attributs doivent être pris en charge par les éléments <initiate-RequestPDU/>, <initiate-ResponsePDU/> et <initiate-ErrorPDU/>.

NOTE Les valeurs par défaut des trois attributs de versionnage correspondent à la première version publiée, de sorte que ces trois attributs peuvent être omis dans cette première version.

Si, lors de la liaison d'association, un point d'extrémité détecte que la version de l'autre point d'extrémité de communication n'est pas conforme à ses propres capacités, la liaison doit se terminer de l'une des façons suivantes:

- Si le problème est détecté par le point d'extrémité qui reçoit la demande Associate, un <initiate-ErrorPDU/> est renvoyé avec <errorClass/><initiate/> et un errorCode = <version-incompatible/>.
- Si le problème est détecté par le point d'extrémité qui a initié l'association, il doit simplement le résoudre.

Une fois l'association établie, chaque point d'extrémité de communication a connaissance des capacités du point d'extrémité associé, et doit ainsi faire en sorte de n'envoyer que les informations conformes à la version prise en charge par son homologue. Toutefois, un point d'extrémité doit être préparé à:

- Accepter les messages XML dans lesquels un nouvel élément inconnu a été ajouté à la fin de la liste des éléments connus, à condition qu'il ne soit pas marqué d'un attribut mustUnderstand. Dans ce cas, ignorer le nouvel élément.
- Si un élément connu contient un élément enfant inconnu avec mustUnderstand=true comme dernier élément enfant, l'ensemble de l'élément parent connu doit être ignoré, mais uniquement si, en outre, l'élément contenant est facultatif dans le XSD du point d'extrémité destinataire. Si l'élément parent n'est pas facultatif, le point d'extrémité doit rejeter le message en envoyant un RejectPDU tel que défini en 8.3.3.4.6.
- Ignorer les nouveaux attributs des éléments connus.

NOTE Cela implique que les nouveaux attributs peuvent être définis dans les versions futures, mais uniquement de manière facultative avec une valeur par défaut, cette valeur devant en outre être rétrocompatible vers "attribut inconnu".

10.2.2.2 Abort

Les indications Abort ACSI sont émises en local par les couches sous-jacentes et n'ont aucune représentation dédiée sur le câble. Les cas d'utilisation et les raisons qui justifient les indications Abort sont détaillés dans les articles ci-dessus.

10.2.2.3 Release

10.2.2.3.1 Généralités

La demande et la réponse+ du service Release ACSI doivent mapper vers les éléments vides <conclude-RequestPDU/> et <conclude-ResponsePDU/>.

La réponse- doit mapper vers un élément <conclude-ErrorPDU> contenant un élément <errorClass/> (voir 10.2.2.3.2).

Le Tableau 27 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés globalement sur ces messages XML de demande et de réponse.

Tableau 27 – Mapping du service Release ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<conclude-RequestPDU/>	Aucun	
AssociationId			
Response+	<conclude-ResponsePDU/>	Aucun	
AssociationId			
Result			
Response–		Voir 10.2.2.3.2	
ServiceError	<conclude-ErrorPDU/>		

10.2.2.3.2 Response–

Le Tableau 28 détaille les mappings des erreurs ACSI vers les valeurs possibles de <errorClass/> inclus dans un <conclude-ErrorPDU/>. Les valeurs de ServiceError ACSI qui n'apparaissent pas dans le tableau ne sont pas mappées.

Tableau 28 – Mappings des erreurs de service Release

Erreur de service ACSI	ServiceError	
	Classe d'erreur	Code d'erreur
instance-in-use	conclude	further-communication-required
failed-due-to-server-constraint	conclude	other

11 Modèle de dispositif logique**11.1 Généralités**

La demande GetLogicalDeviceDirectory mappe vers un élément XML <confirmed-RequestPDU/> contenant un élément <getNameList/> dans lequel la valeur de l'élément <objectClass/> doit être <namedVariable/> (NamedVariable est le mapping MMS des nœuds logiques. Voir le Tableau 2). La Figure 21 donne la structure du message XML correspondant aux données utilisateur (c'est-à-dire sans élément de sécurité qui va encapsuler le message utilisateur dans le message réel sur le câble).

La réponse GetLogicalDeviceDirectory mappe vers un élément XML <confirmed-ResponsePDU/> contenant un élément <GetNameList/>. La Figure 22 donne la structure du message XML correspondant aux données utilisateur.

Le Tableau 29 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés dans ces structures de demande et de réponse, respectivement.

Le service GetLogicalDeviceDirectory peut être restreint par la limitation de taille de PDU négociée. Dans ce cas, la réponse peut renvoyer uniquement un sous-ensemble de références et un indicateur (moreFollows) indiquant la nécessité de demander le sous-ensemble suivant. Le client doit ensuite transmettre une autre demande, spécifiant cette fois-ci la position dans la liste à laquelle il convient que le serveur continue la recherche de nom. En raison de cette contrainte, le mapping de ce service ACSI est en fait une séquence de demandes/réponses, jusqu'à moreFollows=FALSE dans la réponse.

Tableau 29 – Mapping de GetLogicalDeviceDirectory ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNameList/>		Voir la structure de la demande XML à la Figure 21
	<objectClass/>	<namedVariable/>	Valeur fixe
LDName	<domainSpecific/>	<LDName>	
	<continueAfter/>	<LNReferenceX>	Facultatif: nom de la dernière référence renvoyée dans la précédente réponse si elle contient moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNameList/>		Voir la structure de la réponse XML à la Figure 22
LNReference[0..n]	<listOfIdentifier/>	<Identifier>Reference1</Identifier> <Identifier>Reference2</Identifier> ...	Les identifiants renvoyés doivent être les variables nommées aplanies composant les LN du dispositif logique, dont le <objectClass/> est <namedVariable/>. 8.2.2 explique comment construire les variables nommées aplanies d'un LN. Dans le présent article, la liste des identifiants doit être ordonnée de manière lexicographique. Afin de faire correspondre les définitions de l'IEC 61850-7-2, cette liste doit être filtrée par l'application client afin de ne conserver que les nœuds logiques (sans le caractère "\$"). La liste peut éventuellement être vide
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Voir 11.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2329013</invokeID>
  <ConfirmedServiceRequest>
    <getNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL</domainSpecific>
      </objectScope>
    </getNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 21 – Structure XML de GetLogicalDeviceDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>2329013</invokeID>
  <ConfirmedServiceResponse>
    <getNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>LLN0</Identifier>
        <Identifier>LLN0$CF</Identifier>
        <Identifier>LLN0$CF$Mod</Identifier>
        <Identifier>LLN0$CF$Mod$ctlModel</Identifier>
        <Identifier>LLN0$DC</Identifier>
        <Identifier>LLN0$DC$NamPlt</Identifier>
        <Identifier>LLN0$DC$NamPlt$vendor</Identifier>
        <Identifier>LLN0$DC$NamPlt$swRev</Identifier>
        <Identifier>LLN0$DC$NamPlt$d</Identifier>
        .....
        <Identifier>CSWI1$ST$Pos$stVal</Identifier>
        <Identifier>CSWI1$ST$Pos$q</Identifier>
        <Identifier>CSWI1$ST$Pos$st</Identifier>
      </listOfIdentifier>
    </getNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 22 – Structure XML de GetLogicalDeviceDirectory-Response

Si le LDName demandé existe, mais qu'il ne contient aucun nœud logique, un GetLogicalDeviceDirectory-Response doit être envoyé avec MoreFollows=FALSE et un NULL <listOfIdentifier/>.

Le <listOfIdentifier/> renvoyé dans la réponse peut contenir plus de références que des nœuds logiques seuls.. Les identifiants doivent être filtrés par l'application client afin de ne conserver que les références LN, c'est-à-dire les références sans le caractère "\$".

11.2 Response-

La Response- d'un GetLogicalDeviceDirectory est mappée sur un <confirmed-ErrorPDU/> contenant un <serviceError/>:

- Si le LDName demandé contenu dans l'élément <domainSpecific/> de la demande correspondante n'existe pas, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Si le client ne dispose pas des privilèges d'accès pour le LDName demandé contenu dans l'élément <domainSpecific/> de la demande correspondante, le serveur doit renvoyer un <serviceError/> avec <errorClass/><access/> et errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Voir également 8.3.3.4.8.

11.3 Comportement étendu

Le comportement décrit dans le présent paragraphe doit être négocié pendant le processus d'association. L'objectif est qu'il convient que l'application serveur ne renvoie que les informations nécessaires strictement exigées par l'application client et pas la totalité des informations renvoyées dans l'élément <getNameList/> présentées ci-dessus, ce qui permet d'éviter le filtrage au niveau de l'application.

Ce comportement spécifique est indiqué dans la demande en choisissant <extendedGetNameList/> en lieu et place de l'élément <getNameList/> (voir la Figure 23). Ensuite, le message de réponse correspondant contient uniquement la liste des LNName (voir la Figure 24).

Le Tableau 30 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés si le comportement étendu est sélectionné.

Tableau 30 – Mapping étendu de GetLogicalDeviceDirectory ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <extendedGetNameList/>		Voir la structure de la demande XML à la Figure 23
	<objectClass/>	<namedVariable/>	Valeur fixe
LDName	<domainSpecific/>	<LDName>	
	<continueAfter/>	<LNReferenceX>	Facultatif: nom de la dernière référence renvoyée dans la précédente réponse si elle contient moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <extendedGetNameList//>		Voir la structure de la réponse XML à la Figure 24
LNReference[0..n]	<listOfIdentifier/>	<Identifier>LNName1</Identifier> <Identifier>LNName2</Identifier> ...	Les identifiants renvoyés doivent être les noms de tous les LN. La liste peut éventuellement être vide.
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Similaire à 11.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>5440259</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 23 – Structure XML de GetLogicalDeviceDirectory-Request étendu

```

<confirmed-ResponsePDU>
  <invokeID>5440259</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>LLN0</Identifier>
        <Identifier>LPHD1</Identifier>
        <Identifier>GGIO1</Identifier>
        <Identifier>CSWI1</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 24 – Structure XML de GetLogicalDeviceDirectory-Response étendu

12 Modèle de nœud logique

12.1 Généralités

Un nœud logique est composé de différents attributs de classes ACSI différentes (voir le Tableau 31). Tous ces types d'objets peuvent être extraits par le service GetLogicalNodeDirectory.

Tableau 31 – Classes d'objets pour le service GetLogicalNodeDirectory

Classe ACSI demandée dans GetLogicalNodeDirectory	Valeur d'objectClass
DataObject	NamedVariable
DataSet	NamedVariableList
BufferedReportControlBlock	NamedVariable
UnbufferedReportControlBlock	NamedVariable
LogControlBlock	NamedVariable
SettingGroupControlBlock	NamedVariable
Journal	Journal
GooseControlBlock	NamedVariable
GSSEControlBlock (déconseillé)	NamedVariable
MulticastSampledValueControlBlock	NamedVariable
UnicastSampledValueControlBlock (déconseillé)	NamedVariable

12.2 GetLogicalNodeDirectory

12.2.1 Généralités

La demande GetLogicalNodeDirectory mappe vers un élément XML <confirmed-RequestPDU/> contenant un élément <getNameList/> dans lequel la valeur de l'élément <objectClass/> doit être une valeur spécifiée au Tableau 31. La Figure 25 donne la structure du message XML correspondant aux données utilisateur (c'est-à-dire sans élément de sécurité qui va encapsuler le message utilisateur dans le message réel sur le câble).

La réponse GetLogicalNodeDirectory mappe vers un élément XML <confirmed-ResponsePDU/> contenant un élément <GetNameList/>. La Figure 26 donne la structure du message XML correspondant aux données utilisateur.

Le Tableau 32 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés dans ces structures de demande et de réponse, respectivement.

Le service GetLogicalNodeDirectory peut être restreint par la limitation de taille de PDU négociée. Dans ce cas, la réponse peut renvoyer uniquement un sous-ensemble de références et un indicateur (moreFollows) indiquant la nécessité de demander le sous-ensemble suivant. Le client doit ensuite transmettre une autre demande, spécifiant cette fois-ci la position dans la liste à laquelle il convient que le serveur continue la recherche de nom. En raison de cette contrainte, le mapping de ce service ACSI est en fait une séquence de demandes/réponses, jusqu'à moreFollows=FALSE dans la réponse.

Tableau 32 – Mapping de GetLogicalNodeDirectory ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribution XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNameList/>		Voir la structure de la demande XML à la Figure 25
LNReference	<domainSpecific/>	<LDName>	Nom du dispositif logique parent
ACSIClass	<objectClass/>	Voir Tableau 31	
	<continueAfter/>	<ReferenceX>	Facultatif: nom de la dernière référence renvoyée dans la précédente réponse si elle contient moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNameList/>		Voir la structure de la réponse XML à la Figure 26
InstanceName[0..n]	<listOfIdentifier/>	<Identifier>Reference1</Identifier> <Identifier>Reference2</Identifier> ...	Les identifiants renvoyés doivent être les variables nommées aplanies composant les LN du dispositif logique, dont le <objectClass/> est celui spécifié dans l'entrée. 8.2.2 explique comment construire les variables nommées aplanies d'un LN. Dans le présent article, la liste des identifiants doit être ordonnée de manière lexicographique. Afin de faire correspondre les définitions de l'IEC 61850-7-2, cette liste doit être filtrée par l'application client afin de ne conserver que les objets de niveau le plus élevé du seul nœud logique demandé. La liste peut éventuellement être vide.
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Voir 12.2.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceRequest>
    <getNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL</domainSpecific>
      </objectScope>
    </getNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 25 – Structure XML de GetLogicalNodeDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceResponse>
    <getNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>LLN0</Identifier>
        <Identifier>LLN0$CF</Identifier>
        <Identifier>LLN0$CF$Mod</Identifier>
        <Identifier>LLN0$CF$Mod$ctlModel</Identifier>
        <Identifier>LLN0$DC</Identifier>
        <Identifier>LLN0$DC$NamPlt</Identifier>
        <Identifier>LLN0$DC$NamPlt$vendor</Identifier>
        <Identifier>LLN0$DC$NamPlt$swRev</Identifier>
      </listOfIdentifier>
      .....
      <Identifier>LLN0$DC$NamPlt$d</Identifier>
      <Identifier>CSWI1$ST$Pos$stVal</Identifier>
      <Identifier>CSWI1$ST$Pos$st</Identifier>
      <Identifier>CSWI1$ST$Pos$st</Identifier>
    </listOfIdentifier>
  </getNameList>
</ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 26 – Structure XML de GetLogicalNodeDirectory-Response

Le <listOfIdentifier/> renvoyé dans la réponse peut contenir plus de références que seulement les objets demandés du nœud logique demandé. Il est nécessaire que l'application client filtre les identifiants afin de ne conserver que les objets du nœud logique demandé. Si la classe ACSI demandée est DataObject, seuls les noms du niveau le plus élevé doivent être conservés.

Si aucun objet ne correspond à la demande GetLogicalNodeDirectory, une Response+ doit être renvoyée avec le <listOfIdentifiers/> égal à NULL et moreFollows défini sur FALSE.

12.2.2 Response-

La Response- d'un GetLogicalNodeDirectory est mappé sur un <confirmed-ErrorPDU/> contenant un <serviceError/>:

- Si le LDName demandé contenu dans l'élément <domainSpecific/> de la demande correspondante n'existe pas, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Si le client ne dispose pas des privilèges d'accès pour le LDName demandé contenu dans l'élément <domainSpecific/> de la demande correspondante, le serveur doit renvoyer un <serviceError/> avec <errorClass/><access/> et errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existent/>.

- Voir également 8.3.3.4.8.

12.2.3 Comportement étendu

Le comportement décrit dans le présent paragraphe doit être négocié pendant le processus d'association. L'objectif est qu'il convient que l'application serveur ne renvoie que les informations nécessaires strictement exigées par l'application client et pas la totalité des informations renvoyées dans l'élément <getNameList/> présentées ci-dessus, ce qui permet d'éviter le filtrage au niveau de l'application.

Ce comportement spécifique est indiqué dans la demande en choisissant <extendedGetNameList/> en lieu et place de l'élément <getNameList/>.

Le mapping de ce service ACSI exige un processus en deux étapes. Dans une première demande (étape 1), le paramètre LNReference doit être <LDName/LNName> (voir la Figure 27), la réponse devant renvoyer le nom de toutes les contraintes fonctionnelles présentes dans le LN spécifié (voir la Figure 28). Ensuite, pour chaque contrainte fonctionnelle renvoyée, une seconde demande (étape 2) doit être émise avec LNReference= <LDName/LNName\$FC> (voir la Figure 29), le message de réponse devant contenir les noms des objets de premier niveau se trouvant dans le LN spécifié et disposant d'au moins un sous-composant de la contrainte fonctionnelle spécifiée (voir la Figure 30).

Le Tableau 33 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés si le comportement étendu est sélectionné.

Tableau 33 – Mapping étendu de GetLogicalNodeDirectory ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <extendedGetNameList/>		Voir la structure de la demande XML de la Figure 27 (étape 1) et de la Figure 29 (étape 2)
LNReference	<domainSpecific/>	<LDName/LNName> ou <LDName/LNName\$FC>	Étape 1: une demande avec LNReference Étape 2: une demande pour chaque FC
ACSIClass	<objectClass/>	Voir Tableau 31	
	<continueAfter/>	<ReferenceX>	Facultatif: nom de la dernière référence renvoyée dans la précédente réponse si elle contient moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <extendedGetNameList/>		Voir la structure de la réponse XML de la Figure 28 (étape 1) et de la Figure 30 (étape 2)

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
InstanceName [0..n]	<listOfIdentifier/>	<pre> <Identifier>Name1</Identifier> <Identifier>Name2</Identifier> ... </pre>	Les identifiants renvoyés doivent être: Étape 1: les noms des FC présentes pour l'objectClass spécifié Étape 2: les noms des objets de premier niveau dont l'objectClass est celui spécifié dans l'entrée et ayant au moins un sous-composant de la FC spécifiée La liste est ordonnée de manière lexicographique. Elle peut éventuellement être vide.
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Similaire à 12.2.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL/LLN0</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 27 – Structure XML de GetLogicalNodeDirectory-Request étendu (étape 1)

```

<confirmed-ResponsePDU>
  <invokeID>2329054</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>CF</Identifier>
        <Identifier>DC</Identifier>
        <Identifier>EX</Identifier>
        <Identifier>ST</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 28 – Structure XML de GetLogicalNodeDirectory-Response étendu (étape 1)

```

<confirmed-RequestPDU>
  <invokeID>2329058</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass>
          <namedVariable/>
        </objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL/LLN0$ST</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 29 – Structure XML de GetLogicalNodeDirectory-Request étendu (étape 2)

```

<confirmed-ResponsePDU>
  <invokeID>2329058</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>Beh</Identifier>
        <Identifier>Health</Identifier>
        <Identifier>Loc</Identifier>
        <Identifier>Mod</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 30 – Structure XML de GetLogicalNodeDirectory-Response étendu (étape 2)

12.3 GetAllDataValues

12.3.1 Généralités

La demande ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> contenant un élément <read/>. Dans tous les cas, l'élément <variableAccessSpecification/> doit inclure un <variableSpecification/> pour le mapping du paramètre ACSI LNReference (voir 8.2.3).

Toutefois, il est recommandé que la demande ACSI contienne un paramètre de contrainte fonctionnelle (voir l'exemple de la Figure 31). Lorsqu'une contrainte fonctionnelle est spécifiée, l'élément <variableAccessSpecification/> de la demande doit être un <listOfVariable/> limité à un seul <variableSpecification/> et doit également spécifier un <alternateAccess/> en choisissant l'élément <unnamed/>. Dans l'élément <unnamed/>, l'élément <accessSelection/> doit inclure le choix <component/> avec la valeur de la contrainte fonctionnelle spécifiée. Tous les autres mécanismes d'accès sont hors du domaine d'application du présent paragraphe.

NOTE L'accès à toutes les valeurs d'un nœud logique particulier (sans contrainte fonctionnelle SV, CO et sans bloc de commande) peut également être obtenu en utilisant une seule variable nommée, sans accès alternatif. Le nom de la variable serait <LogicalNodeName>.

La Response+ de GetAllDataValues ACSI mappe vers un <confirmed-ResponsePDU/> contenant un élément <read/> avec un élément <success> renvoyant les données en tant qu'élément <structure/> contenant un <structure/> par DataObject et éventuellement des éléments <structure/> imbriqués pour le sous-DO ou le sous-DA imbriqué (voir 8.4).

Le paramètre Response+ de GetAllDataValues ACSI de DataAttributeReference n'est ni mappé ni pris en charge dans ce SCSM.

Le Tableau 34 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés dans les structures de demande et de réponse, respectivement.

Tableau 34 – Mapping de GetAllDataValues ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		Voir la structure de la demande XML à la Figure 31
LNReference	<variableSpecification/>	Si spécifique au domaine, domainId=<LDName> et itemId=<LNName>	Voir d'autres cas en 8.2.3
FunctionalConstraint[0..1]	<alternateAccess/>	Voir l'exemple de la Figure 31	Facultatif. Si l'élément alternateAccess existe, il doit être présent dans la variableAccessSpecification
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		Voir la structure de la réponse XML à la Figure 32
LNReference	<variableSpecification/>		Comme Request, mais facultatif dans Response
DataAttributeReference[1..n]	N/A	N/A	
DataAttributeValue[1..n]	<listOfAccessResult/>	Structure des structures imbriquées	Un niveau imbriqué par sous-DO ou sous-DA supplémentaire (voir 8.4)
Response–	<confirmed-RequestPDU/> <ConfirmedServiceResponse/> <read/> ou <confirmed-ErrorPDU/>	Voir 8.3.3.4.9	

```

<confirmed-RequestPDU>
  <invokeID>121013</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
            <alternateAccess>
              <CHOICE>
                <unnamed>
                  <selectAlternateAccess>
                    <accessSelection>
                      <component>ST</component>
                    </accessSelection>
                    <alternateAccess/>
                  </selectAlternateAccess>
                </unnamed>
              </CHOICE>
            </alternateAccess>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 31 – Structure XML de GetAllDataValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>121013</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <structure>
                  <Data>
                    <integer>1</integer>
                  </Data>
                  <Data>
                    <bit-string>0000000000000</bit-string>
                  </Data>
                  <Data>
                    <utc-time>017215E30000001F</utc-time>
                  </Data>
                </structure>
              </Data>
            </Data>
          </Data>
          [...]
          <Data>
            <structure>
              <Data>
                <bit-string>10</bit-string>
              </Data>
              <Data>
                <bit-string>0000000000000</bit-string>
              </Data>
              <Data>
                <utc-time>017215E30000001F</utc-time>
              </Data>
            </structure>
          </Data>
        </structure>
      </success>
    </AccessResult>
  </listOfAccessResult>
</read>
</ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 32 – Structure XML de GetAllDataValues-Response

13 Modèle DataObject, DataAttribute, SubDataAttribute

13.1 Généralités

Voir 8.2.4 et 8.2.5 pour des informations relatives à la manière de mapper FCD et FCDA (c'est-à-dire respectivement Functionally Constrained Data (données sous contrainte fonctionnelle) et Functionally Constrained Data Attribute (attribut de données sous contrainte fonctionnelle)) vers les éléments <variableAccessSpecification/>.

13.2 GetDataValues

La demande GetDataValues ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> comprenant un élément <read/> dans lequel l'élément

<variableAccessSpecification/> contient un <VariableSpecification/> tel que défini en 8.2.4 et 8.2.5.

La Response+ de GetDataValues ACSI mappe vers un <confirmed-ResponsePDU/> comprenant un élément <read/> avec un élément <success/> renvoyant des données en tant qu'élément de type de base si le paramètre Reference est un DA final (voir la Figure 34, par exemple) ou un élément <structure/> contenant éventuellement et de manière récursive des éléments <structure/> imbriqués dans le cas d'un DO ou DA imbriqué (voir 8.4).

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 35.

Tableau 35 – Mapping des paramètres de service GetDataValues

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		Voir la structure de la demande XML à la Figure 33
Reference	<variableSpecification/>	Voir 8.2.4 et 8.2.5	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		Voir la structure de la réponse XML à la Figure 34
DataAttributeValue[1..n]	<listOfAccessResult/>	Une valeur de type de base ou un élément <structure/> avec un élément <structure/> éventuellement imbriqué de manière récursive (voir 8.4)	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/> ou <confirmed-ErrorPDU/>	Voir 8.3.3.4.9	

```

<confirmed-RequestPDU>
  <invokeID>2676121</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>IEDNameLDINst</domainId>
                  <itemId>LLN0$ST$Beh$stVal</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 33 – Structure XML de GetDataValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>2676121</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <integer>1</integer>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 34 – Structure XML de GetDataValues-Response

13.3 SetDataValues

La demande SetDataValues ACSI doit être mappée vers un élément XML <confirmed-RequestPDU/> comprenant un élément <write/> dans lequel l'élément <variableAccessSpecification/> contient un <VariableSpecification/> tel que défini en 8.2.4 et en 8.2.5, et l'élément <listOfData/> contient un élément <Data/> contenant lui-même soit un élément de type de base (voir la Figure 35, par exemple) si le paramètre Reference est un DA final soit un élément <structure/> contenant des éléments <structure/> éventuellement imbriqués de manière récursive si Reference dispose d'un DO ou d'un DA imbriqué (voir 8.4).

La Response+ de GetDataValues ACSI mappe vers un <confirmed-ResponsePDU/> contenant un élément <write/> avec un élément <success/> vide.

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 36.

Tableau 36 – Mapping des paramètres de service SetDataValues

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		Voir la structure de la demande XML à la Figure 35
Reference	<variableSpecification/>	Voir 8.2.4 et 8.2.5	
DataAttributeValue[1..n]	<listOfData/><Data/>	Une valeur de type de base ou un élément <structure/> avec un élément <structure/> éventuellement imbriqué de manière réursive (voir 8.4)	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/>		Voir la structure de la réponse XML à la Figure 36
	<success/>	Vide	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/> ou <rejectPDU/>	Voir 8.3.3.4.10	

```

<confirmed-RequestPDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>TEST</domainId>
                  <itemId>GGIO1$CF$Mod$ctlModel</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <integer>1</integer>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 35 – Structure XML de SetDataValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 36 – Structure XML de SetDataValues-Response

13.4 GetDataDirectory

13.4.1 Généralités

Le GetDataDirectory ACSI doit mapper vers une séquence de demandes et de réponses.

Les demandes doivent mapper vers un élément XML <confirmed-RequestPDU/> contenant un élément <getVariableAccessAttributes/> dans lequel le paramètre Reference est mappé sur un élément <name/> contenant un élément <domain-specific/> (voir l'exemple de la Figure 37). Dans cette séquence de demandes, l'élément <itemId/> doit parcourir la liste des contraintes fonctionnelles établie par le client.

Chaque Response+ de GetDataDirectory ACSI mappe vers un <confirmed-ResponsePDU/> contenant un élément <getVariableAccessAttributes/> contenant lui-même un élément <typeSpecification> qui donne de manière récursive le nom et le type de tous les composants, niveau par niveau, de tous les DO ou DA imbriqués.

Au niveau d'application IEC 61850, les informations renvoyées dans la réponse doivent être filtrées afin de conserver uniquement les noms des composants sur le niveau sous la référence, sans type.

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 37.

Tableau 37 – Mapping des paramètres de service GetDataDirectory

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getVariableAccessAttributes/>		Une demande et une réponse pour chaque FC. Voir la structure de la demande XML à la Figure 37
DataObjectReference	<name/>	Même structure que l'élément <name/> de 8.2.4	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getVariableAccessAttributes/>		Voir la structure de la réponse XML à la Figure 38 NOTE La valeur de mmsDeletable est toujours "false"
SubDataObjectName[0..n] or DataAttributeName[1..n]	<typeSpecification>	Voir l'exemple de la Figure 38	Filtré au niveau de l'application pour ne conserver que les noms de composants un niveau au-dessous
Response–	<confirmed-ErrorPDU/>	Voir 13.4.2	

```

<confirmed-RequestPDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceRequest>
    <getVariableAccessAttributes>
      <name>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId> LLN0$ST$Mod </itemId>
        </domain-specific>
      </getVariableAccessAttributes>
    </ConfirmedServiceRequest>
  </confirmed-RequestPDU>

```

Figure 37 – Structure XML de GetDataDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceResponse>
    <getVariableAccessAttributes>
      <mmsDeletable>
        false
      </mmsDeletable>
      <typeSpecification>
        <structure packed="true">
          <components>
            <SEQUENCE>
              <componentName>stVal</componentName>
              <componentType>
                <integer>1</integer>
              </componentType>
            </SEQUENCE>
            <SEQUENCE>
              <componentName>q</componentName>
              <componentType>
                <bit-string>13</bit-string>
              </componentType>
            </SEQUENCE>
            <SEQUENCE>
              <componentName>t</componentName>
              <componentType>
                <utc-time/>
              </componentType>
            </SEQUENCE>
          </components>
        </structure>
      </typeSpecification>
    </getVariableAccessAttributes>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 38 – Structure XML de GetDataDirectory-Response

13.4.2 Response-

La Response- de GetDataDirectoryACSI mappe sur un <confirmed-ErrorPDU/> contenant un <serviceError/>.

- Si le <domainId/> ou le <itemId/> spécifié dans la Request n'existe pas, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existent/>.

- Si le client ne dispose d'aucun privilège d'accès au <domainId/> spécifié dans la Request, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Si la demande génère une Response+ qui dépasse la taille de PDU négociée, un <confirmed-ErrorPDU/> doit être renvoyé et le <serviceError/> doit être <errorClass/><service/> avec un errorCode=<pdu-size/>. Celui-ci doit être mappé au ServiceError ACSI failed-due-to-communications-constraint.

Voir le Tableau 38 pour les mappings d'erreurs de service GetDataDirectory.

Tableau 38 – Mappings des erreurs de service GetDataDirectory

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	service	pdu-size	

13.4.3 Comportement étendu

13.4.3.1 Généralités

Le comportement décrit dans le présent paragraphe doit être négocié pendant le processus d'association. L'objectif est qu'il convient que l'application serveur ne renvoie que les informations nécessaires strictement exigées par l'application client et pas la totalité des informations renvoyées dans l'élément <getVariableAccessAttributes/> présentées ci-dessus, ce qui permet d'éviter le filtrage au niveau de l'application.

Pour ce comportement étendu, la demande GetDataDirectory ACSI doit également mapper vers une séquence de demandes. Chaque demande doit être mappée vers un élément XML <confirmed-RequestPDU/> contenant un élément <extendedGetNameList/> (voir l'exemple de la Figure 39) et chaque Response+ doit être mappée vers un élément XML <confirmed-ResponsePDU/> contenant un élément <extendedGetNameList/> (voir l'exemple de la Figure 40). Dans cette séquence de demandes, l'élément <domainSpecific/> doit parcourir la liste des contraintes fonctionnelles établie par le client.

Noter que le choix <extendedGetNameList/> doit être utilisé en lieu et place de l'élément <getNameList/> dans les éléments <confirmed-RequestPDU/> et <confirmed-ResponsePDU/>. Ensuite, le message de réponse correspondant contient uniquement la liste des composants un niveau au-dessous de la référence demandée (voir la Figure 40).

Avec ce comportement étendu, le service GetDataDirectory peut être restreint par la limitation de taille de PDU négociée. Dans ce cas, la réponse peut renvoyer uniquement un sous-ensemble de noms et un indicateur (moreFollows) indiquant la nécessité de demander le sous-ensemble suivant. Le client doit ensuite transmettre une autre demande, spécifiant cette fois-ci la position dans la liste à laquelle il convient que le serveur continue la recherche de nom. En raison de cette contrainte, le mapping de ce service ACSI est en fait une séquence de demandes/réponses, jusqu'à moreFollows=FALSE dans la réponse.

Le Tableau 39 explique comment les paramètres de demande et de réponse définis dans l'IEC 61850-7-2 doivent être mappés si le comportement étendu est sélectionné.

Tableau 39 – Mapping étendu de GetDataDirectory ACSI

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <extendedGetNameList/>		Une demande et une réponse pour chaque FC. Voir la structure de la demande XML à la Figure 39
	<objectClass/>	<namedVariable/>	
DataObjectReference	<domainSpecific/>	<LDName/Reference>	Reference est le nom de la variable nommée aplanie décrite en 8.2.2
	<continueAfter/>	<LNReferenceX>	Facultatif: nom de la dernière référence renvoyée dans la précédente réponse si elle contient moreFollows= true
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <extendedGetNameList/>		Voir la structure de la réponse XML à la Figure 40
SubDataObjectName[0..n] or DataAttributeName[1..n]	<listOfIdentifier/>	<Identifier>Name1</Identifier> <Identifier>Name2</Identifier> ...	Noms de composants un niveau au-dessous de la référence demandée. La liste peut éventuellement être vide
	moreFollows	true false	
Response–	<confirmed-ErrorPDU/>	Voir 13.4.3.2	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceRequest>
    <extendedGetNameList>
      <extendedObjectClass>
        <objectClass><namedVariable/></objectClass>
      </extendedObjectClass>
      <objectScope>
        <domainSpecific>CONTROL/LLN0$ST$Mod</domainSpecific>
      </objectScope>
    </extendedGetNameList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 39 – Structure XML de GetDataDirectory-Request étendu

```

<confirmed-ResponsePDU>
  <invokeID>2424030</invokeID>
  <ConfirmedServiceResponse>
    <extendedGetNameList moreFollows="false">
      <listOfIdentifier>
        <Identifier>stVal</Identifier>
        <Identifier>q</Identifier>
        <Identifier>t</Identifier>
      </listOfIdentifier>
    </extendedGetNameList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 40 – Structure XML de GetDataDirectory-Response étendu

13.4.3.2 Response-

La Response- de GetDataDirectory étendu est mappée sur un <confirmed-ErrorPDU/> contenant un <serviceError/>:

- Si le LDName demandé contenu dans l'élément <domainSpecific/> de la demande correspondante n'existe pas, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existant/>.
- Si le client ne dispose pas des privilèges d'accès pour le LDName demandé contenu dans l'élément <domainSpecific/> de la demande correspondante, le serveur doit renvoyer un <serviceError/> avec <errorClass/><access/> et errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existant/>.
- Voir également 8.3.3.4.8.

13.5 GetDataDefinition

Le mapping de ce service doit être le même que le mapping par défaut de GetDataDirectory au niveau du message XML (voir 13.4.1 et 13.4.2). Toutefois, aucun filtrage ne doit être appliqué au niveau de l'application IEC 61850, de sorte que les noms et les types de tous les composants soient renvoyés, quel que soit leur niveau.

14 Modèle de classe d'ensemble de données

14.1 Généralités

Il existe plusieurs types d'ensembles de données spécifiés dans l'IEC 61850-7-2, persistant et non-persistant. Dans les services suivants, les paramètres contenant une référence d'ensemble de données sont tous mappés sur un <variableAccessSpecification/> contenant un élément <variableListName/>. Toutefois, en fonction du type d'ensembles de données, différents choix doivent être utilisés dans cet élément <variableListName/>:

- Ensemble de données persistant dans un dispositif logique: choice= <domain-specific/> (voir la Figure 41)
- Ensemble de données persistant à l'extérieur d'un dispositif logique: choice=<vmd-specific/> (voir la Figure 42)
- Ensemble de données non persistant: choice=<aa-specific/> (voir la Figure 43)

```

<domain-specific>
  <domainId>CONTROL</domainId>
  <itemId>GGIO1$myDS</itemId>
</domain-specific>

```

Figure 41 – Mapping de la référence à un ensemble de données persistant dans un dispositif logique

```

<vmd-specific>myPersistentDS</vmd-specific>

```

Figure 42 – Mapping de la référence à un ensemble de données persistant à l'extérieur d'un dispositif logique

```

<aa-specific>@myNonPersistentDS</aa-specific>

```

Figure 43 – Mapping de la référence à un ensemble de données non persistant

14.2 GetDataSetValues

14.2.1 Généralités

La demande GetDataSetValues ACSI doit être mappée vers un élément XML <confirmed-RequestPDU/> contenant un élément <read/> (voir l'exemple de la Figure 44).

La Response+ de GetDataSetValues ACSI doit être mappée vers un élément XML <confirmed-RequestPDU/> contenant un élément <read/> (voir l'exemple de la Figure 45).

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 40.

Tableau 40 – Mapping des paramètres de service GetDataSetValues

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		Voir la structure de la demande XML à la Figure 44
	specificationWithResult	true ou false	
DataSetReference	<DomainId/>	<LDName>	Cas d'un ensemble de données persistant dans un dispositif logique. Pour d'autres cas, voir 14.1
	<ItemId/>	<LNName>\$<DSName>	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		Voir la structure de la réponse XML à la Figure 45
DataSetReference	<variableAccessSpecification/>		Facultatif. Doit être restreint par la demande utilisateur.
DataAttributeValue[1..n]	<listOfAccessResult/>		Contient une séquence d'éléments <success/> et <failure/>

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
	<success/>	Une valeur de type de base ou un élément <structure/> avec un élément <structure/> éventuellement imbriqué de manière récursive (voir 8.4)	
	<failure/>	Une valeur énumérée telle que définie par XML complexType DataAccessError	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/> ou <confirmed-ErrorPDU/>	Voir 14.2.2	

La Response+ doit retourner un <listOfAccessResults/>. Les valeurs contenues dans cette liste doivent indiquer le succès ou l'échec. Cela signifie que certains objets AccessResult peuvent retourner <success/> tandis que d'autres retournent <failure/>.

```

<confirmed-RequestPDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <variableListName>
          <domain-specific>
            <domainId>CONTROL</domainId>
            <itemId>GGIO1$myNewDS</itemId>
          </domain-specific>
        </variableListName>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 44 – Structure XML de GetDataSetValues-Request


```

<confirmed-ResponsePDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <bit-string>0000000000000</bit-string>
              </Data>
              <Data>
                <utc-time>017215E00000001F</utc-time>
              </Data>
            </structure>
          </success>
        </AccessResult>
        <AccessResult>
          <failure>
            <temporarily-unavailable/>
          </failure>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 45 – Structure XML de GetDataSetValues-Response

14.2.2 Response–

Les erreurs peuvent être renvoyées en tant que Response+ ou que <confirmed-ErrorPDU/> contenant un <serviceError>.

Le présent article, conjointement avec 8.3.3.4.9, détaille les erreurs qui doivent être retournées.

Le traitement suivant doit survenir avant le traitement conformément à 8.3.3.4.9.

- Si la demande spécifie un dispositif logique (c'est-à-dire <domainId/>) pour lequel le client ne dispose d'aucun privilège d'accès, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existant/>.
- Si l'ensemble de données n'existe pas, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existant/>.
- Une demande dont le code de réponse dépasse la taille de PDU négociée doit retourner un <serviceError/> avec un <errorClass/>=<service/> avec un errorCode=<pdu-size/>.

L'ensemble des autres erreurs/traitements doit être conforme à 8.3.3.4.9.

Voir le Tableau 41 pour les mappings d'erreurs GetDataSetValues.

Tableau 41 – Mappings des erreurs GetDataSetValues

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	service	pdu-size	
NOTE Pour d'autres codes d'erreur, voir 8.3.3.4.9.			

14.3 SetDataSetValues

14.3.1 Généralités

La demande SetDataSetValues ACSI doit être mappée vers un élément XML <confirmed-RequestPDU/> contenant un élément <write/> (voir l'exemple de la Figure 46). L'élément <listOfData/> doit contenir un élément <Data/> pour chaque référence de membre DataSet. Chaque élément <Data/> contient soit un élément de type de base si la référence de membre DataSet est un DA final, soit un élément <structure/> contenant des éléments <structure/> éventuellement imbriqués de manière récursive si la référence de membre DataSet contient un DO ou DA imbriqué (voir 8.4).

La Response+ de SetDataSetValues ACSI doit être mappée vers un élément XML <confirmed-ResponsePDU/> contenant un élément <write/> (voir l'exemple de la Figure 47).

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 42.

Tableau 42 – Mapping des paramètres de service SetDataSetValues

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest> <write>		Voir la structure de la demande XML à la Figure 46
DataSetReference	<DomainId/>	<LDName>	Cas d'un ensemble de données persistant dans un dispositif logique. Pour d'autres cas, voir 14.1
	<IdemId/>	<LNName>\${DSName}>	
DataAttributeValue[1..n]	<listOfData/>	Liste des éléments </Data> contenant une valeur de type de base ou un élément <structure/> avec un élément <structure/> éventuellement imbriqué de manière récursive (voir 8.4)	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse> <write>		Voir la structure de la réponse XML à la Figure 47

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Result	<success/>		
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse> <write> ou <confirmed-ErrorPDU/> ou <rejectPDU/>	Voir 14.3.2	

```

<confirmed-RequestPDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <variableListName>
          <domain-specific>
            <domainId>CONTROL</domainId>
            <itemId>GGIO1$myDS</itemId>
          </domain-specific>
        </variableListName>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <integer>1</integer>
        </Data>
        <Data>
          <integer>2</integer>
        </Data>
        <Data>
          <integer>3</integer>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 46 – Structure XML de SetDataSetValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>1322032</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 47 – Structure XML de SetDataSetValues-Response

14.3.2 Response–

Les erreurs peuvent être renvoyées en tant que Response+, qu'élément <rejectPDU/> ou que <confirmed-ErrorPDU/> contenant un <serviceError>.

Le présent article, conjointement avec 8.3.3.4.10, détaille les erreurs qui doivent être retournées.

Le traitement suivant doit survenir avant le traitement conformément à 8.3.3.4.10.

- Une demande SetDataSetValues qui dépasse la taille de PDU négociée doit générer un <rejectPDU/> avec un rejectPDUType=<confirmed-requestPDU/> et un rejectCode=<value-out-of-range/>. Celui-ci doit être mappé au ServiceError ACSI "failed-due-to-communications-constraint".
- Si la demande spécifie un dispositif logique (c'est-à-dire <domainId/>) pour lequel le client ne dispose d'aucun privilège d'accès, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Si l'ensemble de données n'existe pas, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existent/>.

L'ensemble des autres erreurs/traitements doit être conforme à 8.3.3.4.10.

Voir le Tableau 43 pour les mappings d'erreurs SetDataSetValues.

Tableau 43 – Mappings des erreurs SetDataSetValues

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
Erreur de service ACSI	Rejet		Commentaires
	Type PDU du rejet	Code de rejet	
failed-due-to-communications-constraint	Confirmed-requestPDU	Value-out-of-range	
NOTE Pour d'autres codes d'erreur, voir 8.3.3.4.10.			

14.4 CreateDataSet

14.4.1 Généralités

La demande CreateDataSet ACSI doit être mappée vers un élément XML <confirmed-RequestPDU/> contenant un élément <defineNamedVariableList/> (voir l'exemple de la Figure 48).

La Response+ de CreateDataSet ACSI doit être mappée vers un élément XML <confirmed-ResponsePDU/> contenant un élément <defineNamedVariableList/> (voir l'exemple de la Figure 49).

Le mapping des paramètres de service ACSI doit se conformer aux définitions du Tableau 44 et le mapping d'erreur de service doit se conformer aux définitions de 14.4.2.

Tableau 44 – Mapping des paramètres de service CreateDataSet

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <defineNamedVariableList/>		Voir la structure de la demande XML à la Figure 48
DataSetReference	<domainId/>	<LDName>	Cas d'un ensemble de données persistant dans un dispositif logique. Pour d'autres cas, voir 14.1
	<idemId/>	<LNName>\${DSName}>	
DSMemberRef[1..n]	<listOfVariable/>		Liste de FCD ou de FCDA. Chaque FCD ou FCDA est mappé comme indiqué en 8.2.4 et en 8.2.5
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <defineNamedVariableList/>		Uniquement un élément vide: <defineNamedVariableList/>. Voir la structure de la réponse XML à la Figure 49
Response–	<confirmed-ErrorPDU/>	Voir 14.4.2	

```

<confirmed-RequestPDU>
  <invokeID>36113013</invokeID>
  <ConfirmedServiceRequest>
    <DefineNamedVariableList>
      <variableListName>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId>GGIO1$myNewDS</itemId>
        </domain-specific>
      </variableListName>
      <listOfVariable>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$ST$Ind1</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$ST$Ind3</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
      </listOfVariable>
    </DefineNamedVariableList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 48 – Structure XML de CreateDataSet-Request

```

<confirmed-ResponsePDU>
  <invokeID>36113013</invokeID>
  <ConfirmedServiceResponse>
    <DefineNamedVariableList/>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 49 – Structure XML de CreateDataSet-Response

14.4.2 Response–

La Response- de CreateDataSet est mappée sur un <confirmed-ErrorPDU/> contenant un élément <serviceError/>:

- Si la demande CreateDataSet spécifie un dispositif logique (c'est-à-dire <domainId>) qui n'existe pas, le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><access/> avec un errorCode=<object-non-existant/>.
- Si la demande spécifie un dispositif logique (c'est-à-dire <domainId/>) pour lequel le client ne dispose d'aucun privilège d'accès, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existant/>.
- Si l'un des <variableSpecification/> inclus dans l'élément <listOfVariable/> de la demande spécifie un objet non existant, le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><definition/> avec un errorCode=<object-undefined/>.

- Si le nom d'objet contenu dans l'élément <variableListName/> existe déjà, le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><definition/> avec un errorCode=<object-exists/>.
- Si le nombre de <variableSpecification/> dans <listOfVariable/> dépasse la valeur maximale admise, le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><resource/> avec un errorCode=<capability-unavailable/>. Celui-ci doit être mappé au ServiceError ACSI de "failed-due-to-server-constraint".
- Une demande CreateDataSet qui dépasse la taille de PDU négociée doit générer un <rejectPDU/> avec un rejectPDUType=<confirmed-requestPDU/> et un rejectCode=<value-out-of-range/>. Celui-ci doit être mappé au ServiceError ACSI "failed-due-to-communications-constraint".
- Une demande CreateDataSet qui spécifie un dispositif logique (c'est-à-dire <domainId>) et spécifie un <variableListName/> qui ne contient pas le nom d'un nœud logique existant, doit générer un <confirmed-ErrorPDU/> avec un <serviceError/><errorClass/><access/> et un errorCode=<object-non-existent/>. Celui-ci doit être mappé au ServiceError ACSI de "parameter-value-inconsistent".

Voir le Tableau 45 pour les mappings d'erreurs de service CreateDataSet.

Tableau 45 – Mappings des erreurs de service CreateDataSet

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
instance-in-use	definition	object-exists	
parameter-value-inconsistent	definition	object-undefined	
failed-due-to-server-constraint	resource	capability-unavailable	
failed-due-to-communications-constraint	Toutes les classes	Tous les codes d'erreur non mappés	
Erreur de service ACSI	Rejet		Commentaires
	Type PDU du rejet	Code de rejet	
failed-due-to-communications-constraint	Confirmed-requestPDU	Value-out-of-range	

14.5 DeleteDataSet

14.5.1 Généralités

La demande DeleteDataSet ACSI doit être mappée vers un élément XML <confirmed-RequestPDU/> contenant un élément <deleteNamedVariableList/> (voir l'exemple de la Figure 50).

La Response+ de DeleteDataSet ACSI doit être mappée vers un élément XML <confirmed-ResponsePDU/> contenant un élément <deleteNamedVariableList/> (voir l'exemple de la Figure 51).

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 46.

Tableau 46 – Mapping des paramètres de service DeleteDataSet

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément/attribut XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <deleteNamedVariableList/>		Voir la structure de la demande XML à la Figure 50
	scopeOfDelete	specific	Attribut avec valeur fixe
DataSetReference	<domainId/>	<LDName>	Cas d'un ensemble de données persistant dans un dispositif logique. Pour d'autres cas, voir 14.1
	<itemId/>	<LNName>\$<DSName>	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <deleteNamedVariableList/>		Voir la structure de la réponse XML à la Figure 51
	<numberMatched/>	1 (en cas de succès)	Pour les valeurs en cas d'erreur, voir 14.5.2
	<numberDeleted/>	1 (en cas de succès)	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <deleteNamedVariableList/> ou <confirmed-ErrorPDU/>	Voir 14.5.2	

```

<confirmed-RequestPDU>
  <invokeID>1322432</invokeID>
  <ConfirmedServiceRequest>
    <deleteNamedVariableList scopeOfDelete="specific">
      <listOfVariableListName>
        <ObjectName>
          <domain-specific>
            <domainId>CONTROL</domainId>
            <itemId>GGIO1$myDS</itemId>
          </domain-specific>
        </ObjectName>
      </listOfVariableListName>
    </deleteNamedVariableList>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 50 – Structure XML de DeleteDataSet-Request


```
<confirmed-ResponsePDU>
  <invokeID>1322432</invokeID>
  <ConfirmedServiceResponse>
    <deleteNamedVariableList>
      <numberMatched>1</numberMatched>
      <numberDeleted>1</numberDeleted>
    </deleteNamedVariableList>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>
```

Figure 51 – Structure XML de DeleteDataSet-Response

14.5.2 Erreurs

Le service DeleteDataSet ACSI peut renvoyer des erreurs en tant que Response+ ou que <confirmed-ErrorPDU/> contenant un <serviceError/> ou que <confirmed-ErrorPDU/> contenant un <serviceError/> avec un élément <serviceSpecificInformation/>.

- Les demandes spécifiant un élément <scopeOfDelete/> avec des valeurs <aa-specific/>, <domain/> ou <vmd/> sont hors du domaine d'application de ce SCSM.
- Si l'ensemble de données spécifié n'existe pas, une Response+ avec <numberMatched/>=0 doit être renvoyée
- Si l'ensemble de données spécifié existe, mais qu'il n'est pas supprimable, une Response+ avec <numberMatched/>=1 et <numberDeleted/>=0 doit être renvoyée.
- Si l'ensemble de données spécifié est supprimable, mais qu'il ne peut pas être supprimé (s'il est utilisé dans un bloc de commande de rapport, par exemple), une Response- (c'est-à-dire <confirmed-ErrorPDU/> contenant un <serviceSpecificInformation/>) avec le nombre de <deleteNamedVariableList/> défini sur 0 doit être renvoyée.

Voir le Tableau 47 pour les mappings d'erreurs de service DeleteDataSet.

Tableau 47 – Mappings des erreurs de service DeleteDataSet

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	Toutes les classes	Tous les codes d'erreur non mappés	
Erreur de service ACSI	Response+		Commentaires
	Valeur de	paramètre	
instance-not-available	numberMatched	0	
	numberDeleted	0	
failed-due-to-server-constraint	numberMatched	1	
	numberDeleted	0	
Erreur de service ACSI	Response-		Commentaires
	Valeur de	paramètre	
failed-due-to-server-constraint	numberDeleted	0	Restreint au type d'erreur: Classe d'erreur = Service code d'erreur = object- constraint-conflict

14.6 GetDataSetDirectory

14.6.1 Généralités

La demande GetDataSetDirectory ACSI doit être mappée vers un élément XML <confirmed-RequestPDU/> contenant un élément <getNamedVariableListAttributes/> (voir l'exemple de la Figure 52).

La Response+ de GetDataSetDirectory ACSI doit être mappée vers un élément XML <confirmed-ResponsePDU/> contenant un élément <getNamedVariableListAttributes/> (voir l'exemple de la Figure 53).

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 48.

Tableau 48 – Mapping des paramètres de service GetDataSetDirectory

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <getNamedVariableListAttributes/>		Voir la structure de la demande XML à la Figure 52
DataSetReference	<domainId/>	<LDName>	Cas d'un ensemble de données persistant dans un dispositif logique. Pour d'autres cas, voir 14.1
	<itemId/>	<LNName>\${DSName}</LNName>	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <getNamedVariableListAttributes/>		Voir la structure de la réponse XML à la Figure 53
DSMemberRef[1..n]	<listOfVariable/>		Liste de FCD ou de FCDA. Chaque FCD ou FCDA est mappé comme indiqué en 8.2.4 et en 8.2.5
Response–	<confirmed-ErrorPDU/>	Voir 14.6.2	

```

<confirmed-RequestPDU>
  <invokeID>1322763</invokeID>
  <ConfirmedServiceRequest>
    <getNamedVariableListAttributes>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$myDS</itemId>
      </domain-specific>
    </getNamedVariableListAttributes>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 52 – Structure XML de GetDataSetDirectory-Request

```

<confirmed-ResponsePDU>
  <invokeID>1322763</invokeID>
  <ConfirmedServiceResponse>
    <getNamedVariableListAttributes>
      <mmsDeletable>true</mmsDeletable>
      <listOfVariable>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CF$SPCSO1$ctlModel</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CF$SPCSO2$ctlModel</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CF$DPCSO1$ctlModel</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
      </listOfVariable>
    </getNamedVariableListAttributes>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 53 – Structure XML de GetDataSetDirectory-Response

14.6.2 Response-

La Response- de GetDataSetDirectory est mappée sur un <confirmed-ErrorPDU/> contenant un <serviceError/>:

- Si la demande GetDataSetDirectory spécifie un dispositif logique (c'est-à-dire <domainId>) qui n'existe pas, le <serviceError/> dans le <confirmed-ErrorPDU/> doit être <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Si l'ensemble de données n'existe pas, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Si la demande spécifie un dispositif logique (c'est-à-dire <domainId/>) pour lequel le client ne dispose d'aucun privilège d'accès, le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-access-denied/> ou un <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Une demande dont le code de réponse dépasse la taille de PDU négociée doit retourner un <serviceError/><errorClass/><service/> avec un errorCode=<pdu-size/>. Celui-ci doit être mappé au ServiceError ACSI failed-due-to-communications-constraint.

Voir le Tableau 49 pour les mappings d'erreurs de service GetDataSetDirectory.

Tableau 49 – Mappings des erreurs de service GetDataSetDirectory

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-communications-constraint	service	pdu-size	
failed-due-to-communications-constraint	Toutes les classes	Tous les codes d'erreur non mappés	

15 Modèle ServiceTracking

Le modèle service tracking repose sur des classes de données communes (CDC) spécifiques définies dans l'IEC 61850-7-3. Le rôle de ces CDC consiste à stocker les paramètres utilisés par les services ACSI afin de les consigner dans un journal ou un rapport à chaque appel d'un service. Les attributs de ces CDC utilisent les types de données déjà mappés en 8.3.1 de sorte que le présent article ne décrit pas le mapping détaillé pour toutes les CDC. Les seules informations présentées ici sont:

- Les valeurs possibles des attributs définis comme ENUMERATED par le CST de classe de données communes: serviceType (voir le Tableau 50) et errorCode (Tableau 51).
- Le mapping des CDC dont la définition est modifiée dans ce SCSM comparée à l'IEC 61850-7-3: classe LTS (voir le Tableau 52) et GTS (voir le Tableau 53).

Tableau 50 – Mapping des valeurs de ServiceType ACSI

Service ACSI	Valeur
Unknown	0
Associate	1
Abort	2
Release	3
GetServerDirectory	4
GetLogicalDeviceDirectory	5
GetAllDataValues	6
GetDataValues	7
SetDataValues	8
GetDataDirectory	9
GetDataDefinition	10
GetDataSetValues	11
SetDataSetValues	12
CreateDataSet	13
DeleteDataSet	14
GetDataSetDirectory	15
SelectActiveSG	16
SelectEditSG	17
SetEditSGValue	18

Service ACSI	Valeur
ConfirmEditSGValues	19
GetEditSGValue	20
GetSGCBValues	21
Rapport	22
GetBRCBValues	23
SetBRCBValues	24
GetURCBValues	25
SetURCBValues	26
GetLCBValues	27
SetLCBValues	28
QueryLogByTime	29
QueryLogAfter	30
GetLogStatusValues	31
SendGOOSEMessage	32 ^a
GetGoCBValues	33
SetGoCBValues	34
GetGoReference	35 ^a
GetGOOSEElementNumber	36 ^a
SendMSVMessage	37 ^a
GetMSVCBValues	38
SetMSVCBValues	39
SendUSVMessage	40 ^a
GetUSVCBValues	41
SetUSVCBValues	42
Select	43
SelectWithValue	44
Cancel	45
Operate	46
CommandTermination	47
TimeActivatedOperate	48
GetFile	49
SetFile	50
DeleteFile	51
GetFileAttributeValues	52
TimeSynchronization	53
InternalChangeUnknown	54
GetLogicalNodeDirectory	55
^a Ces services ne sont pas mappés dans l'IEC 61850-8-2. Toutefois, leur exécution dans le contexte d'un autre SCSM peut être consignée dans un journal ou un rapport sur la base de l'IEC 61850-8-2.	

Tableau 51 – Mapping des valeurs d'errorCode ACSI

ServiceError ACSI	Valeur
no-error	0
instance-not-available	1
instance-in-use	2
access-violation	3
access-not-allowed-in-current-state	4
parameter-value-inappropriate	5
parameter-value-inconsistent	6
class-not-supported	7
instance-locked-by-other-client	8
control-must-be-selected	9
type-conflict	10
failed-due-to-communications-constraint	11
failed-due-to-server-constraint	12

Tableau 52 – Mapping de CDC LTS

Nom d'attribut ACSI	Nom de composant	Type ACSI	Commentaire
Doit hériter de tous les composants MMS du CDC CST			
Spécifique au LTS			
logEna	logEna	BOOLEAN	Voir 8.3.1
logRef	logRef	ObjectReference	Voir 8.3.3.2.3
datSet	datSet	ObjectReference	Voir 8.3.3.2.3
	oldEntrTm	EntryTime	Voir 8.3.3.8. Présenté avec le mapping du LCB dans l'IEC 61850-8-2
	newEntrTm	EntryTime	Voir 8.3.3.8. Présenté avec le mapping du LCB dans l'IEC 61850-8-2
	oldEnt	EntryID	Voir 8.3.3.5. Présenté avec le mapping du LCB dans l'IEC 61850-8-2
	newEnt	EntryID	Voir 8.3.3.5. Présenté avec le mapping du LCB dans l'IEC 61850-8-2
optFlds			Non mappé. Le présent Article oblige les reasonCodes à toujours apparaître dans le journal.
trgOps	trgOps	TriggerConditions	Voir 8.3.3.9
intgPd	intgPd	INT32U	Voir 8.3.1

Tableau 53 – Mapping de CDC GTS

Nom d'attribut ACSI	Nom de composant	Type ACSI	Commentaire
Doit hériter de tous les composants MMS du CDC CST			
Spécifique au GTS			
goEna	goEna	BOOLEAN	Voir 8.3.1
goID	goID	VISIBLE STRING	Voir 8.3.2.5
datSet	datSet	ObjectReference	La valeur de ce composant doit être au format de l'ObjectReference et doit être limitée au VMD ou au domaine couvert par NamedVariableLists
confRev	confRev	INT32U	Voir 8.3.1
ndsCom	ndsCom	BOOLEAN	Voir 8.3.1
dstAddress	dstAddress	PhyComAddr	Restreint au transport utilisé. Voir 8.3.3.3.1 et 8.3.3.3.2
	minTime	INT32U	Voir 8.3.1. Présenté avec le mapping du GoCB dans l'IEC 61850-8-2
	maxTime	INT32U	Voir 8.3.1. Présenté avec le mapping du GoCB dans l'IEC 61850-8-2
	fixedOffs	BOOLEAN	Voir 8.3.1. Présenté avec le mapping du GoCB dans l'IEC 61850-8-2

16 Modèle de classe de commande de groupe de réglage

16.1 Définition de bloc de commande de groupe de réglage (SGCB)

Si le SGCB existe dans un dispositif logique, il doit être considéré comme un objet spécifique du nœud logique LLN0, dans la contrainte fonctionnelle SP.

La structure représentant le SGCB doit être du type défini au Tableau 54. Les conditions de présence des composants ACSI ne sont pas différentes de celles indiquées dans la spécification IEC 61850-7-2.

Tableau 54 – Mapping de SGCB

Nom de composant ACSI	Type ACSI	Commentaire
NumOfSG	INT8U	Mappé selon 8.3.1
ActSG	INT8U	Mappé selon 8.3.1
EditSG	INT8U	Mappé selon 8.3.1
CnfEdit	BOOLEAN	Mappé selon 8.3.1
LActTm	Timestamp	Mappé selon 8.3.1
ResvTms	INT16U	Mappé selon 8.3.1

16.2 Services de classe de commande de groupe de réglage

16.2.1 SelectActiveSG

Ce service est une spécialisation du service SetDataValues (voir 13.3) dans lequel la valeur de ActSG doit être définie. Voir la Figure 54 pour un exemple de demande SelectActiveSG pour l'activation du premier groupe de réglage d'un dispositif logique nommé CONTROL.

La Figure 55 et la Figure 56 donnent des exemples, respectivement, d'une réponse positive et d'une réponse négative si le SGCb référencé n'existe pas. Noter que ces exemples de réponses positives et négatives peuvent s'appliquer à d'autres services de la présente classe, tant qu'ils sont mappés sur un élément XML <confirmed-RequestPDU/> contenant un élément <write/> (c'est-à-dire SelectEditSG, SetEditSGValue et ConfirmEditSGValues). Pour les différentes erreurs possibles incluses dans la réponse négative, voir 8.3.3.4.10.

```
<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB$ActSG</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <unsigned>1</unsigned>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>
```

Figure 54 – Structure XML de SelectActiveSG-Request

```
<confirmed-ResponsePDU>
  <invokeID>2018020</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>
```

Figure 55 – Structure XML de SelectActiveSG-Response+

```
<confirmed-ResponsePDU>
  <invokeID>2018020</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <failure>
          <object-non-existent/>
        </failure>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>
```

Figure 56 – Structure XML de SelectActiveSG-Response-

16.2.2 SelectEditSG

Ce service est une spécialisation du service SetDataValues (voir 13.3) dans lequel la valeur d'EditSG doit être définie. Voir la Figure 57 pour un exemple de demande SelectEditSG pour la sélection du second groupe de réglage d'un dispositif logique, pour l'édition.

```

<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB$EditSG</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <unsigned>2</unsigned>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 57 – Structure XML de SelectEditSG-Request

16.2.3 SetEditSGValue

Ce service est une spécialisation du service SetDataValues (voir 13.3) dans lequel la contrainte fonctionnelle de FCD ou de FCDA doit être SE. Voir la Figure 58 pour un exemple de demande SetEditSGValue.

Avant l'exécution de ce service, il est nécessaire de sélectionner le groupe d'édition approprié en utilisant le service SelectEditSG.

```

<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>PVOC1$SE$MinOpTmms$setVal</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <integer>21</integer>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 58 – Structure XML de SetEditSGValue-Request

16.2.4 ConfirmEditSGValues

Ce service est une spécialisation du service SetDataValues (voir 13.3). La valeur de CnfEdit doit être définie sur TRUE. Voir la Figure 59 pour un exemple de demande ConfirmEditSGValues.

L'effet du temps et de la méthode de mise à jour du stockage non volatil est défini localement. La méthodologie doit être documentée dans le PIXIT. La valeur de CnfEdit doit retourner à FALSE à l'issue du stockage.

```

<confirmed-RequestPDU>
  <invokeId>2018020</invokeId>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB$CnfEdit</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <boolean>true</boolean>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 59 – Structure XML de ConfirmEditSGValues

16.2.5 GetEditSGValue

Ce service est une spécialisation du service GetDataValues (voir 13.2) dans lequel la contrainte fonctionnelle de FCD ou de FCDA doit être SE ou SG. Voir la Figure 60 et la Figure 61 pour des exemples de demande et de réponse GetEditSGValue.

Avant l'exécution de ce service, il est nécessaire de sélectionner le groupe d'édition et/ou groupe actif approprié en utilisant le service SelectEditSG ou SelectActiveSG.

```

<confirmed-RequestPDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>PVOC1$SE$MinOpTmms$setVal</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 60 – Structure XML de GetEditSGValue-Request

```

<confirmed-ResponsePDU>
  <invokeID>263031</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <integer>123</integer>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 61 – Structure XML de GetEditSGValue-Response

16.2.6 GetSGCBValues

Ce service est une spécialisation du service GetDataValues (voir 13.2) dans lequel la référence à lire est l'objet SGCB (avec FC=SP). Voir la Figure 62 et la Figure 63 pour des exemples de demande et de réponse GetSGCBValues, respectivement.

```

<confirmed-RequestPDU>
  <invokeID>2637821</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>LLN0$SP$SGCB</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 62 – Structure XML de GetSGCBValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>2637821</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <unsigned>3</unsigned>
              </Data>
              <Data>
                <unsigned>1</unsigned>
              </Data>
              <Data>
                <unsigned>2</unsigned>
              </Data>
              <Data>
                <boolean>>false</boolean>
              </Data>
              <Data>
                <utc-time>017215E00000001F</utc-time>
              </Data>
              <Data>
                <unsigned>600</unsigned>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 63 – Structure XML de GetSGCBValues-Response

17 Modèle de classe de reporting et de journalisation

17.1 Modèle de rapport – Blocs de commande de rapport

17.1.1 Bloc de commande de rapport mis en mémoire tampon

Un BRCB doit être un objet spécifique dans une instance d'un nœud logique, sous la contrainte fonctionnelle BR. La structure du BRCB doit être telle que définie au Tableau 55. Les types des attributs BRCB doivent être tels que définis en 8.3.

Tableau 55 – Structure BRCB

Attributs BRCB	Type ACSI	Commentaire
RptID	VISIBLE STRING	Peut ne pas être écrit lorsque RptEna=TRUE
RptEna	BOOLEAN	
DatSet	ObjectReference	Peut ne pas être écrit lorsque RptEna=TRUE
ConfRev	INT32U	Peut ne pas être écrit
OptFlds	RCBReportOptions	Peut ne pas être écrit lorsque RptEna=TRUE
BufTm	INT32U	Peut ne pas être écrit lorsque RptEna=TRUE
SqNum	INT16U	Peut ne pas être écrit
TrgOps	TriggerConditions	Peut ne pas être écrit lorsque RptEna=TRUE
IntgPd	INT32U	Peut ne pas être écrit lorsque RptEna=TRUE
GI	BOOLEAN	
PurgeBuf	BOOLEAN	Peut ne pas être écrit lorsque RptEna=TRUE
EntryID	EntryID	Peut ne pas être écrit lorsque RptEna=TRUE
TimeOfEntry	EntryTime	Peut ne pas être écrit
ResvTms	INT16	Peut ne pas être écrit lorsque RptEna=TRUE
Owner	OCTET STRING 64	

Une écriture de modification de valeur de RptID, DatSet, BufTm, TrgOps, IntgPd, ou PurgeBuf doit exécuter la procédure PurgeBuf décrite dans l'IEC 61850-7-2.

La dénomination des instances d'un BRCB doit être contrôlée par l'attribut de l'IEC 61850-6 RCB "indexed":

- Si l'attribut de l'IEC 61850-6 RCB "indexed" est TRUE, la numérotation d'instance de BRCB doit commencer à 01 et progresser jusqu'à 99. La numérotation des instances à deux chiffres est obligatoire. Par exemple, le SCL <RCB name="myRcb" indexed="true"> génère les composants nommés MMS myRCB01, myRCB02, ..., myRcbXX où XX représente l'attribut max du subElement de SCL RptEnabled du RCB en question.
- Si l'attribut de l'IEC 61850-6 RCB "indexed" est FALSE, l'instance de BRCB doit être sans numéro d'instance additionnel. Par exemple, le SCL <RCB name="myRcb" indexed="false"> conduit à un composant nommé MMS myRCB.

RptEna

Une tentative de définition de la valeur de RptEna doit échouer si la valeur de DatSet est NULL.

DatSet

Une tentative de définition d'une référence à un DataSet qui n'existe pas doit échouer sauf si la valeur est NULL. La valeur NULL doit indiquer qu'il n'y a pas de DataSet.

OptFlds

OptFlds doit être mappé vers un BITSTRING conformément à 8.3.3.12.

Le bit de segmentation est réservé afin de maintenir la correspondance entre OptFlds et ReportedOptFlds.

Owner

La lecture d'Owner (propriétaire) doit retourner le JID de l'entité d'application qui a réservé le bloc de commande de rapport. L'adresse IP doit être codée comme suit: exactement 4 octets pour une adresse IPv4 (par exemple: C0A80017 sont les 4 octets à utiliser pour coder

l'adresse IPv4 "192.168.0.23") et 16 octets pour une adresse IPv6. Il existe deux types de propriétaires de bloc de commande de rapport: le propriétaire préconfiguré et le propriétaire en ligne. Si le bloc de commande de rapport a été préconfiguré dans le fichier de station SCL et associé à un élément SCL ClientLN dédié, la lecture d'Owner doit retourner l'adresse JID du client pour lequel il a été configuré, même si le ClientLN ne se trouve pas dans le même domaine XMPP et se connecte au serveur par l'intermédiaire de plusieurs serveurs XMPP (fédération).

Si le bloc de commande de rapport a été préconfiguré sans association à un élément SCL ClientLN dédié ou que le BRCB a été instancié pour un client de reporting dynamique, la lecture d'Owner retourne une chaîne vide.

L'attribut Owner est destiné au diagnostic uniquement et ne doit pas être utilisé pour refuser une demande de réservation à un client. L'IEC 62351-6 qui admet l'authentification d'un utilisateur et de ses droits définit la valeur qui fournit un mécanisme d'authentification pour traiter le refus de réservation.

NOTE L'IEC 61850-7-2 définit une association d'un bloc de commande de rapport avec un ensemble de clients en coopération. Cependant, à ce jour, le SCL IEC 61850-6 ne prend pas en charge le concept d'ensemble de clients en coopération. Par conséquent, la propriété préconfigurée statique est limitée à l'élément SCL ClientLN.

17.1.2 Bloc de commande de rapport non mis en mémoire tampon

Un URCB doit être un objet spécifique dans une instance d'un nœud logique, sous la contrainte fonctionnelle RP. La structure de l'URCB doit être telle que définie au Tableau 56. Les types d'attributs URCB doivent être mappés comme indiqué en 8.3.

Tableau 56 – Structure URCB

Attributs URCB	Type ACSI	Commentaire
RptID	VISIBLE STRING	Peut ne pas être écrit lorsque RptEna=TRUE
RptEna	BOOLEAN	
Resv	BOOLEAN	Peut ne pas être écrit lorsque RptEna=TRUE
DatSet	ObjectReference	Peut ne pas être écrit lorsque RptEna=TRUE
ConfRev	INT32U	Peut ne pas être écrit
OptFlds	RCBReportOptions	Peut ne pas être écrit lorsque RptEna=TRUE. Les valeurs de buffer-overflow et d'entryID doivent être ignorées
BufTm	INT32U	Peut ne pas être écrit lorsque RptEna=TRUE
SqNum	INT16U	Peut ne pas être écrit
TrgOps	TriggerConditions	Peut ne pas être écrit lorsque RptEna=TRUE
IntgPd	INT32U	Peut ne pas être écrit lorsque RptEna=TRUE
GI	BOOLEAN	
Owner	OCTET STRING	Voir 17.1.1

OptFlds

La valeur retournée pour OptFlds.buffer-overflow et OptFlds.entryID doit toujours être False.

Les valeurs d'OptFlds doivent refléter la/les valeur(s) par défaut si aucune écriture d'OptFlds n'a eu lieu alors que l'URCB est réservé (Resv=TRUE, par exemple).

Si une écriture a lieu alors que Resv=FALSE, la valeur retournée lors de la lecture d'OptFlds est définie localement.

Une transition de RptEna de FALSE à TRUE doit amener la valeur de SqNum à être définie à zéro (0).

La dénomination des instances d'un URCB doit être contrôlée par l'attribut de l'IEC 61850-6 RCB "indexed":

- Si l'attribut de l'IEC 61850-6 RCB "indexed" est TRUE, la numérotation d'instance d'URCB doit commencer à 01 et progresser jusqu'à 99. La numérotation des instances à deux chiffres est obligatoire. Par exemple, le SCL <RCB name="myRcb" indexed="true"> génère les composants nommés MMS myRCB01, myRCB02, ..., myRcbXX où XX représente l'attribut max du subElement de SCL RptEnabled du RCB en question.
- Si l'attribut de l'IEC 61850-6 RCB "indexed" est FALSE, l'instance URCB doit être sans numéro d'instance. Par exemple, le SCL <RCB name="myRcb" indexed="false"> conduit à un composant nommé MMS myRCB.

RptEna

Une tentative de définition de la valeur de RptEna doit échouer si la valeur de DataSet est NULL.

DatSet

Une tentative de définition d'une référence à un DataSet qui n'existe pas doit échouer sauf si la valeur est NULL. La valeur NULL doit indiquer qu'il n'y a pas de DataSet.

17.2 Services de reporting

17.2.1 Service de rapport

Un rapport doit être mappé à un élément XML <unconfirmed-PDU/> contenant un élément <informationReport/>. La Figure 64 donne un exemple de message Report.

L'élément <variableAccessSpecification/> inclus dans ce <informationReport/> doit utiliser le choix <variableListName/> et <vmd-specific/> avec la valeur "RPT".

Outre <variableAccessSpecification/>, l'objet <informationReport/> doit être composé d'une séquence d'AccessResult. Les valeurs d'AccessResult doivent être comme décrites au Tableau 57.

La valeur des AccessResults correspondant à un attribut dans le bloc de commande doit être mappée comme dans l'attribut correspondant dans le bloc de commande: voir les types ACSI au Tableau 55 et au Tableau 56 pour, respectivement, les attributs de BRCB et d'URCB, et voir 8.3 pour savoir comment mapper les valeurs du présent SCSM.

Le mapping des AccessResults qui ne correspondent pas à un attribut dans les blocs de commande est expliqué ci-dessous.

Tableau 57 – Ordre des AccessResults pour Report

Nom de paramètre du format de rapport IEC 61850-7-2	Condition
RptID	Doit toujours être présent
Reported OptFlds	Doit toujours être présent
SeqNum	Doit être présent si OptFlds.sequence-number est TRUE
TimeOfEntry	Doit être présent si OptFlds.report-time-stamp est TRUE
DatSet	Doit être présent si OptFlds.data-set-name est TRUE
BufOvfl	Doit être présent si OptFlds.buffer-overflow est TRUE
EntryID	Doit être présent si OptFlds.entryID est TRUE
ConfRev	Doit être présent si OptFlds.conf-rev est TRUE
SubSeqNum	Doit être présent si OptFlds.segmentation est TRUE
MoreSegmentsFollow	Doit être présent si OptFlds.segmentation est TRUE
Inclusion-bitstring	Doit être présent
data-reference(s)	Doit être présent si OptFlds.data-reference est TRUE
value(s)	Voir AccessResult pour la/les valeur(s)
ReasonCode(s)	Doit être présent si OptFlds.reason-for-inclusion est TRUE

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <variableListName>
          <vmd-specific>RPT</vmd-specific>
        </variableListName>
      </variableAccessSpecification>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$BR$simpleBReport05
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <bit-string>0111110010</bit-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <unsigned>2312</unsigned>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <binary-time>031DC5B02DD2</binary-time>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$myNewDS
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <unsigned>1</unsigned>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <bit-string>11</bit-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$ST$Ind1
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <visible-string>CONTROL/GGIO1$ST$Ind3
          </visible-string>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <bit-string>000000000000</bit-string>
              </Data>
              <Data>
                <utc-time>017215E00000001F</utc-time>
              </Data>
            </structure>
          </success>
        </AccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>

```

← RptID
 ← OptFlds
 ← SeqNum
 ← TimeOfEntry
 ← DataSet
 ← ConfRev
 ← Inclusion-bitstring
 ← data-reference(s)
 ← value(s)
 ← CONTROL/GGIO1\$ST\$Ind1\$stVal
 ← CONTROL/GGIO1\$ST\$Ind1\$q (= 0)
 ← CONTROL/GGIO1\$ST\$Ind1\$t

<boolean>>false</boolean>	← CONTROL/GGIO1\$ST\$Ind3\$stVal
</Data>	
<Data>	
<bit-string>000000000000</bit-string>	← CONTROL/GGIO1\$ST\$Ind3\$q (= 0)
</Data>	
<Data>	
<utc-time>017215E30000001F</utc-time>	← CONTROL/GGIO1\$ST\$Ind3\$t
</Data>	
</structure>	
</success>	
</AccessResult>	
<AccessResult>	
<success>	← ReasonCode(s)
<bit-string>000010</bit-string>	
</success>	
</AccessResult>	
<AccessResult>	
<success>	
<bit-string>000010</bit-string>	
</success>	
</AccessResult>	
</listOfAccessResult>	
</informationReport>	
</UnconfirmedService>	
</unconfirmed-PDU>	

Figure 64 – Structure XML de Report

AccessResult pour RptID

Si la valeur de RptID du bloc de commande est NULL, l'AccessResult doit contenir la référence d'objet spécifiant le bloc de commande qui a causé la génération du rapport.

AccessResult pour les OptFlds rapportés

Les valeurs pour ce résultat sont définies en 8.3.3.12.

Le bit de segmentation doit être utilisé pour indiquer la présence ou l'absence des résultats d'accès SubSeqNum et MoreSegmentsFollow.

Si le bit de segmentation est TRUE, les AccessResult SubSeqNum et MoreSegmentsFollow doivent être présents. Si le bit de segmentation est FALSE, les AccessResult SubSeqNum et MoreSegmentsFollow ne doivent pas être présents.

AccessResult pour BufOvfl

L'AccessResult contenant la valeur de BufOvfl doit être mappé comme un type ACSI BOOLEAN (voir le mapping en 8.3). La valeur FALSE indique qu'il n'y a pas de dépassement de capacité de mémoire tampon.

AccessResult pour SubSeqNum

La valeur acheminée dans cet AccessResult doit être mappée comme un type ACSI INT16U (voir le mapping en 8.3).

AccessResult pour MoreSegmentsFollow

L'AccessResult contenant la valeur de MoreSegmentsFollow doit être mappé comme un type ACSI BOOLEAN (voir le mapping en 8.3). La valeur TRUE indique que d'autres sous-séquences de rapport sont exigées pour compléter l'ensemble du rapport.

AccessResult pour inclusion-bitstring

L'AccessResult contenant la valeur du bitstring d'inclusion doit être mappé en tant que BITSTRING (voir le mapping en 8.3.2.1). Le nombre de bits utilisés dans le bitstring doit être égal au nombre de membres de DataSet. Si une valeur de bit est TRUE, une valeur du membre d'ensemble de données correspondant est présente. Si OptFld.data-reference et/ou OptFlds.reason-for-inclusion sont TRUE, ces éléments AccessResult doivent également être présents. L'ordre des bits dans le bitstring doit être l'ordre des membres de l'ensemble de données.

AccessResult pour data-reference(s)

L'AccessResult contenant la valeur de data-reference doit être mappé en tant que type ACSI ObjectReference (voir le mapping en 8.3.3.2.4). La valeur de l'AccessResult doit être l'ObjectReference pour le membre de l'ensemble de données dont la valeur est incluse. L'ordre du/des AccessResult(s) doit être l'ordre des membres de l'ensemble de données.

AccessResult pour la/les valeur(s)

Cette liste d'AccessResult doit contenir les éléments <success/> avec les données correspondant aux membres du DataSet qui sont rapportées. Chaque élément <success/> doit être une valeur de type de base ou un élément <structure/> contenant plusieurs types de base, avec un élément <structure/> éventuellement imbriqué de manière récursive (voir 8.4). L'ordre des éléments <success/> doit être l'ordre des membres de l'ensemble de données.

AccessResult pour le(s) ReasonCode(s)

Le paramètre Reason-for-inclusion doit être codé sous forme d'AccessResult multiple. Chaque AccessResult doit être mappé en tant que type ACSI ReasonForInclusionInReport. La taille de l'AccessResult et la définition binaire doivent correspondre à la définition de ReasonForInclusionInReport (voir 8.3.3.10). Le bit general-interrogation (Bit 5) doit être utilisé pour indiquer que la raison est liée à une interrogation générale. Le nombre d'AccessResults doit être le même que le nombre d'AccessResults de l'attribut de valeur(s) dans l'InformationReport. L'ordre du/des AccessResult(s) doit être l'ordre des membres de l'ensemble de données.

17.2.2 GetBRCBValues

La demande GetBRCBValues ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> comprenant un élément <read/> dans lequel l'élément <variableAccessSpecification/> contient un <VariableSpecification/> tel que défini en 8.2.4 et où la référence à lire est le BRCB (avec FC=BR, voir la Figure 65).

La Response+ du GetBRCBValues ACSI mappe vers un <confirmed-ResponsePDU/> contenant un élément <read/> avec un élément <success/> renvoyant un élément <structure/> contenant tous les attributs du BRCB (voir la Figure 66).

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 58.

Tableau 58 – Mapping des paramètres de service GetBRCBValues

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		Voir la structure de la demande XML à la Figure 65
BRCBReference	<variableSpecification/>	Voir 8.2.4	
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		Voir la structure de la réponse XML à la Figure 66

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
ReportIdentifier	<listOfAccessResult/>	Un élément <structure/> contenant une valeur de type de base pour chaque attribut du BRCB (voir 8.4 et la Figure 66)	
ReportEnable			
DataSetReference			
ConfigurationRevision			
OptionalFields			
BufferTime			
SequenceNumber			
TriggerOptionsEnabled			
IntegrityPeriod			
GeneralInterrogation			
PurgeBuf			
EntryIdentifier			
TimeOfEntry			
ReserveTimeSecond			
Owner			
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/> ou <confirmed-ErrorPDU/>	Voir 8.3.3.4.9	
ServiceError			

```

<confirmed-RequestPDU>
  <invokeID>195015</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1$BR$simpleBReport05</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 65 – Structure XML de GetBRCBValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>195015</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <visible-string>
                  CONTROL/GGIO1$BR$simpleBReport05
                </visible-string>
              </Data>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <visible-string>
                  CONTROL/GGIO1$myNewDS
                </visible-string>
              </Data>
              <Data>
                <unsigned>1</unsigned>
              </Data>
              <Data>
                <bit-string>0111110010</bit-string>
              </Data>
              <Data>
                <unsigned>0</unsigned>
              </Data>
              <Data>
                <unsigned>0</unsigned>
              </Data>
              <Data>
                <bit-string>000010</bit-string>
              </Data>
              <Data>
                <unsigned>0</unsigned>
              </Data>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <boolean>>false</boolean>
              </Data>
              <Data>
                <octet-string>
                  0000000000000F40
                </octet-string>
              </Data>
              <Data>
                <binary-time>
                  031DC5B02DD2
                </binary-time>
              </Data>
              <Data>
                <integer>0</integer>
              </Data>
              <Data>
                <octet-string/>
              </Data>
            </structure>

```

←RptID
 ←RptEna
 ←DatSet
 ←ConfRev
 ←OptFlds
 ←BufTm
 ←SqNum
 ←TrgOps
 ←IntgPd
 ←GI
 ←PurgeBuf
 ←EntryID
 ←TimeOfEntry
 ←ResvTms
 ←Owner

```

</success>
</AccessResult>
</listOfAccessResult>
</read>
</ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 66 – Structure XML de GetBRCBValues-Response

17.2.3 SetBRCBValues

La demande SetBRCBValues ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> comprenant un élément <write/> dans lequel l'élément <variableAccessSpecification/> contient au moins un élément <variableSpecification/> tel que défini en 8.2.5. Un élément <variableSpecification/> doit être présent par attribut à définir dans le BRCB (avec FC=BR). L'élément <listOfData/> doit inclure au moins une valeur de type de base, une pour chaque attribut à définir dans le BRCB. Si la demande spécifie plusieurs <variableSpecification/>, chacun est traité de manière individuelle, traité dans l'ordre de la liste et peut individuellement échouer, alors que les autres aboutissent. La même demande ne peut pas être utilisée pour définir plusieurs fois le même élément BRCB.

La Response+ de SetBRCBValues ACSI mappe vers un <confirmed-ResponsePDU/> contenant un élément <write/> avec au moins un élément <success/> ou <failure/>, indiquant que l'écriture des différents attributs a abouti ou échoué. L'échec de l'écriture d'un attribut n'implique pas celui des autres opérations.

L'erreur de service de Response– doit être mappée comme indiqué au 8.3.3.4.10.

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 59.

Une mise en œuvre peut décider qu'un seul attribut de bloc de commande peut être écrit par appel de service. Cela doit être précisé dans son PICS/PIXIT.

Tableau 59 – Mapping des paramètres de service SetBRCBValues

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		Voir la structure de la demande XML à la Figure 67
BRCBReference	<variableAccessSpecification/>	Un élément <variableSpecification/> pour chaque attribut à définir dans le BRCB (voir 8.2.5)	
ReserveTimeSecond [0..1]	<listOfData/><Data/>	Une valeur de type de base pour chaque attribut décrit par un élément <variableSpecification/> ci-dessus	
ReportIdentifier [0..1]			
DataSetReference [0..1]			
OptionalFields [0..1]			
BufferTime [0..1]			
TriggerOptionsEnabled [0..1]			
IntegrityPeriod [0..1]			
PurgeBuffer [0..1]			
EntryIdentifier [0..1]			
ReportEnable [0..1]			
GeneralInterrogation [0..1]			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/>		Voir la structure de la réponse XML à la Figure 68
	<success/> ou <failure/>	Les éléments <success/> sont vides. Pour les valeurs <failure/>, voir 8.3.3.4.10	
Response–	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <write/> ou <rejectPDU/>	Voir 8.3.3.4.10	
ServiceError			

```

<confirmed-RequestPDU>
<invokeID>3726014</invokeID>
<ConfirmedServiceRequest>
  <write>
    <variableAccessSpecification>
      <listOfVariable>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$BR$simpleBReport05$ResvTms</itemId>
              </domain-specific>

```



```

</name>
</variableSpecification>
</SEQUENCE>
<SEQUENCE>
  <variableSpecification>
    <name>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$BR$simpleBReport05$DatSet</itemId>
      </domain-specific>
    </name>
  </variableSpecification>
</SEQUENCE>
<SEQUENCE>
  <variableSpecification>
    <name>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$BR$simpleBReport05$OptFlds</itemId>
      </domain-specific>
    </name>
  </variableSpecification>
</SEQUENCE>
<SEQUENCE>
  <variableSpecification>
    <name>
      <domain-specific>
        <domainId>CONTROL</domainId>
        <itemId>GGIO1$BR$simpleBReport05$TrgOps</itemId>
      </domain-specific>
    </name>
  </variableSpecification>
</SEQUENCE>
</listOfVariable>
</variableAccessSpecification>
<listOfData>
  <Data>
    <integer>600</integer>
  </Data>
  <Data>
    <visible-string>CONTROL/GGIO1$UnkownDS</visible-string>
  </Data>
  <Data>
    <bit-string>0111010000</bit-string>
  </Data>
  <Data>
    <bit-string>011100</bit-string>
  </Data>
</listOfData>
</write>
</ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 67 – Structure XML de SetBRCBValues-Request

```

<confirmed-ResponsePDU>
  <invokeID>3726014</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
      <CHOICE>
        <failure>
          <object-undefined/>
        </failure>
      </CHOICE>
    </CHOICE>
    <CHOICE>
      <success/>
    </CHOICE>
    <CHOICE>
      <success/>
    </CHOICE>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 68 – Structure XML de SetBRCBValues-Response

17.2.4 GetURCBValues

Doit être mappé conformément à 17.2.2.

17.2.5 SetURCBValues

Doit être mappé conformément à 17.2.3.

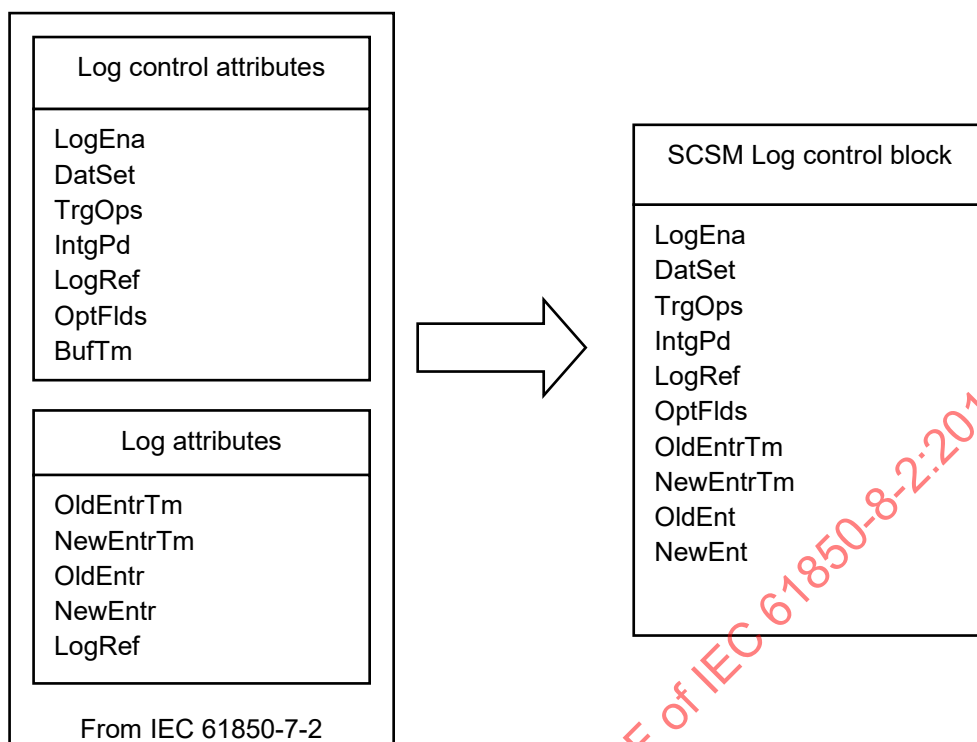
17.3 Modèle Log (journal)

17.3.1 Vue d'ensemble

L'IEC 61850-7-2 spécifie deux composants majeurs pour le modèle Log: une classe de commande et la classe de journal elle-même. Toutefois, pour le présent SCSM, ces deux classes sont fusionnées en un bloc de commande de journal (LCB – Log Control Block) sous la contrainte fonctionnelle "LG".

Le bloc de commande de journal combine les attributs de la classe de commande de journal de l'IEC 61850-7-2 et les autres attributs de la classe de journal (voir la Figure 69).

NOTE Les entrées de journal définies dans l'IEC 61850-7-2 en tant qu'attributs Entry[1..n] de la classe de commande de journal peuvent être accessibles par les services QueryLogByTime et QueryLogAfter en tant qu'éléments <JournalEntry/> (voir 17.3.3.4).



IEC 815/11

Figure 69 – Relation des attributs LCB avec les définitions de journal (log) de l'IEC 61850-7-2

La structure du LCB doit être telle que définie au Tableau 60. Les types d'attributs LCB doivent être mappés comme indiqué en 8.3.

Tableau 60 – Structure LCB

Attributs LCB	Type ACSI	lecture/ écriture	o/f	Commentaires
LogEna	BOOLEAN	lecture/écriture	o	
LogRef	ObjectReference	lecture/écriture	o/f	Peut ne pas être écrit lorsque LogEna=TRUE
DatSet	ObjectReference	lecture/écriture	o/f	Peut ne pas être écrit lorsque LogEna=TRUE
OldEntrTm	EntryTime	lecture	o	Voir ^a
NewEntrTm	EntryTime	lecture	o	Voir ^a
OldEntr	EntryID	lecture	o	Voir ^a
NewEntr	EntryID	lecture	o	Voir ^a
OptFlds	LCBLogEntryOptions			Non mappé. Le présent Article oblige les reasonCodes à toujours apparaître dans le journal.
TrgOps	TriggerConditions	lecture/écriture	o	Peut ne pas être écrit lorsque LogEna=TRUE
IntgPd	INT32U	lecture/écriture	f	Peut ne pas être écrit lorsque LogEna=TRUE
bufTm				Non mappé. Chaque entrée est traitée comme si bufTm était 0, c'est-à-dire sans mise en mémoire tampon supplémentaire des événements.
^a Ces attributs sont définis dans le cadre de Log Object de l'IEC 61850-7-2, mais sont mappés vers des attributs supplémentaires dans le LCB dans le contexte de la présente norme. Les valeurs de ces attributs doivent refléter les valeurs appropriées pour le journal spécifié par la valeur de LogRef.				

17.3.2 Description des attributs LCB

17.3.2.1 LogEna

La valeur par défaut de LogEna est définie localement. Cependant, la valeur par défaut doit être FALSE pour des LCB non configurés ou incorrectement configurés.

17.3.2.2 LogRef

Doit être du type de données spécifié en 8.3.3.2.2. La forme de la valeur doit être celle d'un ObjectReference. La valeur DEFAULT doit être un ObjectReference spécifiant le dispositif logique et le nom réel d'un journal (c'est-à-dire LogName).

NOTE Un journal est le lieu de stockage des entrées de journal. Il s'agit d'un concept hérité de MMS, mais toutes les informations exigées pour le mettre en œuvre dans le contexte de ce SCSM sont fournies ici.

Le nom du journal par défaut doit être "GeneralLog". Par conséquent, la valeur par défaut de LogRef doit être sous la forme: <Nom de dispositif logique>/<LNName>\$GeneralLog.

Exemple Pour un LogRef dans le nœud logique LLN0 du dispositif logique "LD1", la valeur de LogRef serait "LD1/LLN0\$GeneralLog".

L'écriture d'une valeur LogRef qui spécifie un journal inexistant doit échouer.

Une valeur NULL est réservée pour indiquer qu'aucun journal n'a été sélectionné.

Plusieurs LCB peuvent référencer le même journal en spécifiant la même valeur de LogRef.

Il est recommandé que les mises en œuvre revendiquant la conformité au modèle de journal mettent en œuvre au moins un LCB faisant partie de LLN0.

17.3.2.3 DataSet

Doit être du type de données spécifié en 8.3.3.2.2. L'écriture d'une valeur DataSet qui spécifie un ensemble de données inexistant doit échouer.

Une valeur NULL est réservée pour indiquer qu'aucun ensemble de données n'a été sélectionné.

17.3.2.4 OldEntrTm et NewEntrTm

Les valeurs doivent spécifier l'Entry-Time.

En l'absence d'entrée dans le journal, spécifié par LogRef, la valeur de ces composants doit être égale et les valeurs binary-time doivent être égales à 0.

17.3.2.5 OldEntr et NewEntr

Les valeurs contenues dans OldEntr et NewEntr doivent être du type EntryID (voir 8.3.3.5). Ces valeurs doivent représenter les valeurs entryIdentifier de journal pour les entrées les plus anciennes et les plus récentes.

Si le journal spécifié par LogRef ne contient pas d'entrée, la valeur de ces composants doit être égale et NULL.

17.3.3 Mapping des services de journal et de commande de journal

17.3.3.1 GetLCBValues

Le mapping de GetLCBValues s'apparente à celui de GetBRCBValues (voir 17.2.2), mais la référence à lire est le LCB (avec FC=LG) et la Response+ doit renvoyer les valeurs des attributs LCB.

17.3.3.2 SetLCBValues

Le mapping de SetLCBValues ASCII s'apparente à celui de SetBRCBValues (voir 17.2.3).

L'écriture d'une valeur, lorsque la valeur du composant LogEna est TRUE, doit échouer pour les attributs LogRef, DataSet, TrgOps et IntgPd.

17.3.3.3 GetLogStatusValues

Les attributs définis dans le cadre de Log Object de l'IEC 61850-7-2 étant mappés à des attributs supplémentaires dans le LCB, le GetLogStatusValues est identique à GetLCBValues.

17.3.3.4 QueryLogByTime

17.3.3.4.1 Généralités

La demande QueryLogByTime ACSI doit être mappée à un élément XML <confirmed-RequestPDU/> contenant un élément <readJournal/> (voir l'exemple de la Figure 70).

La Response+ de QueryLogByTime ACSI mappe vers un <confirmed-ResponsePDU/> contenant un élément <readJournal/> (voir l'exemple à la Figure 71).

L'erreur de service de Response- doit être mappée comme indiqué au 17.3.3.4.2.

Le mapping des paramètres de service ACSI doit être tel que défini au Tableau 61.

Tableau 61 – Mapping des paramètres de service QueryLogByTime

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <readJournal/>		Voir la structure de la demande XML à la Figure 70
LogReference	<domainId/>	<LDName>	Voir 17.3.2.2
	<itemId/>	<LNName>\${<LogName>	
RangeStartTime [0..1]	<startingTime/>	xsd:hexbinary de 6 octets	Voir le type TimeOfDay en 8.3.3.8
RangeStopTime [0..1]	<endingTime/>		
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <readJournal/>		Voir la structure de la demande XML à la Figure 71
ListOfLogEntries	<listOfJournalEntry/>		
TimeOfEntry EntryID EntryData [1..n] DataRef Valeur ReasonCode	<JournalEntry/>		Voir les détails ci-dessous. Plusieurs LogEntry peuvent être incluses dans un <JournalEntry/>
Response–	<confirmed-ErrorPDU/>	Voir 17.3.3.4.2	

```

<confirmed-RequestPDU>
  <invokeID>3722376</invokeID>
  <ConfirmedServiceRequest>
    <readJournal>
      <journalName>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId>LLN0$GeneralLog</itemId>
        </domain-specific>
      </journalName>
      <rangeStartSpecification>
        <startingTime>01D2EB402DB1</startingTime>
      </rangeStartSpecification>
      <rangeStopSpecification>
        <endingTime>01F3E0C02DB1</endingTime>
      </rangeStopSpecification>
    </readJournal>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 70 – Structure XML de QueryLogByTime-Request

```

<confirmed-ResponsePDU>
  <invokeID>3722376</invokeID>
  <ConfirmedServiceResponse>
    <readJournal moreFollows="false">
      <listOfJournalEntry>
        <JournalEntry>
          <entryIdentifier>0000000000000009</entryIdentifier>
          <originatingApplication/>
          <entryContent>
            <occurrenceTime>01D377DF2DB1</occurrenceTime>
            <entryForm>
              <data>
                <event>
                  <eventConditionName>
                    <domain-specific>
                      <domainId>CONTROL</domainId>
                      <itemId>LLN0$LG$myLCB</itemId>
                    </domain-specific>
                  </eventConditionName>
                  <currentState>active</currentState>
                </event>
                <listOfVariables>
                  <SEQUENCE>
                    <variableTag>
                      CONTROL/GGIO1$ST$Ind1$stVal
                    </variableTag>
                    <valueSpecification>
                      <boolean>true</boolean>
                    </valueSpecification>
                  </SEQUENCE>
                  <SEQUENCE>
                    <variableTag>ReasonCode</variableTag>
                    <valueSpecification>
                      <bit-string>000010</bit-string>
                    </valueSpecification>
                  </SEQUENCE>
                </listOfVariables>
              </data>
            </entryForm>
          </entryContent>
        </JournalEntry>
        <JournalEntry>
          <entryIdentifier>000000000000000A</entryIdentifier>
          <originatingApplication>
            <ap-invocation-id>101</ap-invocation-id>
          </originatingApplication>
          <entryContent>
            <occurrenceTime>01D982BF2DB1</occurrenceTime>
            <entryForm>
              <data>
                <listOfVariables>
                  <SEQUENCE>
                    <variableTag>
                      CONTROL/GGIO1$ST$Ind1$stVal
                    </variableTag>
                    <valueSpecification>
                      <boolean>>false</boolean>
                    </valueSpecification>
                  </SEQUENCE>
                  <SEQUENCE>
                    <variableTag>ReasonCode</variableTag>
                    <valueSpecification>

```

```

        <bit-string>000010</bit-string>
      </valueSpecification>
    </SEQUENCE>
  </listOfVariables>
</data>
</entryForm>
</entryContent>
</JournalEntry>
</listOfJournalEntry>
</readJournal>
</ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 71 – Structure XML de QueryLogByTime-Response

TimeOfEntry

Mappe vers l'élément <occurrenceTime/> dans <entryContent/>. La valeur doit être xsd:hexbinary de 6 octets (voir TimeOfDay en 8.3.3.8).

EntryID

Mappe vers l'élément <entryIdentifier/> de <JournalEntry/>. La valeur doit être un xsd:hexbinary de 8 octets. Le contenu de la valeur est défini localement. Un identifiant d'entrée dont les octets ont tous une valeur de 0 ne doit pas être autorisé. Cette valeur est réservée pour indiquer que le journal est vide.

EntryData

Mappe vers un élément <entryForm/> dans <entryContent/>. Dans <entryForm/>, le choix <data/> doit être utilisé pour mapper l'EntryData. L'utilisation du choix <annotation/> est définie localement.

Dans l'élément <data/>, le sous-élément <event/> doit être utilisé pour indiquer des informations entrées dans le journal en liaison avec la commande ou le traitement d'un bloc de commande de journal. L'utilisation de cette contrainte n'est pas restreinte exclusivement à l'indication d'une commande LCB, mais d'autres utilisations sont considérées comme étant définies localement.

L'élément <event/> doit être utilisé pour journaliser des transitions de la valeur de LogEna LCD comme prescrit:

- L'élément <eventConditionName/> doit contenir l'ObjectName de NamedVariable représentant le bloc de commande de journal à l'origine de la création de l'entrée de journal.
- L'état de condition d'événement (élément <currentState/>) doit être l'état du composant LogEna du bloc de commande de journal à l'origine de la création de l'entrée de journal.

Le mapping de valeurs entre les valeurs de LogEna et les valeurs d'état de condition d'événement doit être:

- Une valeur de LogEna FALSE doit être mappée à l'état de condition d'événement <disabled/>. La transition de la valeur de LogEna de TRUE à FALSE doit provoquer la création d'une entrée de journal.
- Une valeur de LogEna TRUE doit être mappée à l'état de condition d'événement <idle/> ou <active/>. L'état <idle/> est réservé pour indiquer une transition de la valeur de LogEna de FALSE à TRUE. Seule une entrée de journal avec une valeur d'état de condition d'événement <active/> doit avoir une liste de variables de journal.

Lors d'une mise sous tension ou d'une réinitialisation, il convient de journaliser le statut du LogEna du LCB avec une valeur <idle/>.

Ensuite, l'élément <listOfVariables/> doit être utilisé pour la journalisation de valeurs issues des DataSet spécifiés par un LCB.

Chaque LogEntry est mappée dans le même <listOfVariables/> que deux <SEQUENCE/>, respectivement pour le codage de dataRef/Value et de reasonCode.

NOTE Plusieurs LogEntry peuvent être incluses dans le même <listOfVariables/>. Chaque LogEntry est mappée comme deux éléments <SEQUENCE/>, comme indiqué dans le présent article.

Chaque <SEQUENCE/> est composée d'un <variableTag/> et d'une <valueSpecification/>. Le <variableTag/> doit avoir une taille minimale-maximale de 129 octets.

Le <variableTag/> du premier <SEQUENCE/> (c'est-à-dire le codage de DataRef/Value) doit avoir une valeur se présentant sous la forme ObjectReference (voir 8.3.3.2.2). L'élément <valueSpecification/> doit contenir une valeur de type de base ou un élément <structure/> avec un élément <structure/> éventuellement imbriqué de manière récursive (voir 8.4).

L'élément <variableTag/> du second <SEQUENCE/> (c'est-à-dire le codage de ReasonCode) doit avoir la valeur de "ReasonCode". L'élément <valueSpecification/> doit être un BITSTRING tel que défini en 8.3.3.11 ReasonForInclusionInLog.

Exemple La valeur d'un variableTag pour un reasonCode relatif à un dataRef, dont l'ObjectReference est LD1/MMXU\$MX\$V, serait: "ReasonCode".

17.3.3.4.2 Response-

La Response- est mappée sur un <confirmed-ErrorPDU/> contenant un <serviceError/>.

- Si la demande spécifie un nom de dispositif logique qui n'existe pas, un <confirmed-ErrorPDU/> doit être renvoyé et le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-non-existent/>.
- Si la demande spécifie un LogName pour lequel le client ne dispose d'aucun privilège d'accès, un <confirmed-ErrorPDU/> doit être renvoyé, et le <serviceError/> doit être <errorClass/><access/> avec un errorCode=<object-access-denied/> ou <errorClass/><access/> avec un errorCode=<object-non-existent/>.

Voir le Tableau 62 pour les mappings de ServiceError pour les services Log.

Tableau 62 – Mappings de ServiceError pour les services Log

Erreur de service ACSI	ServiceError		Commentaires
	Classe d'erreur	Code d'erreur	
instance-not-available	access	object-non-existent	
access-violation	access	object-access-denied	
failed-due-to-server-constraint	Toutes les classes	Tous les codes d'erreur non mappés	

17.3.3.5 QueryLogAfter

La demande QueryLogAfter ACSI doit être mappée à un élément XML <confirmed-RequestPDU/> contenant un élément <readJournal/> (voir l'exemple de la Figure 72).

La Response+ du QueryLogAfter ACSI s'apparente à la Response+ de QueryLogByTime (voir 17.3.3.4.1)

La Response- du QueryLogAfter ACSI s'apparente à la Response- de QueryLogByTime (voir 17.3.3.4.2)

Le mapping des paramètres de demande doit être tel que défini au Tableau 63.

Tableau 63 – Mapping des paramètres de QueryLogAfter-Request

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <readJournal/>		Voir la structure de la demande XML à la Figure 72
LogReference	<domainId/>	<LDName>	Voir 17.3.2.2
	<itemId/>	<LNName>\${<LogName>	
RangeStartTime	<timeSpecification/>	xsd:hexbinary de 6 octets	Voir le type TimeOfDay en 8.3.3.8
Entry	<entrySpecification/>	xsd:hexbinary de 8 octets	Voir l'explication pour EntryID en 17.3.3.4.1

```

<confirmed-RequestPDU>
  <invokeID>3722380</invokeID>
  <ConfirmedServiceRequest>
    <readJournal>
      <journalName>
        <domain-specific>
          <domainId>CONTROL</domainId>
          <itemId>LLN0$GeneralLog</itemId>
        </domain-specific>
      </journalName>
      <entryToStartAfter>
        <timeSpecification>01D2EB402DB1</timeSpecification>
        <entrySpecification>0000000000000009</entrySpecification>
      </entryToStartAfter>
    </readJournal>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 72 – Structure XML de QueryLogAfter-Request

17.3.4 Conformité

Une mise en œuvre qui revendique la conformité au modèle de journal ACSI doit prendre en charge les services décrits au Tableau 64.

Tableau 64 – Exigences de conformité de journal

Service ACSI	Client	Serveur
GetLCBValue	o	o
SetLCBValue	f	o
QueryLogByTime	c1	o
QueryLogAfter	c1	o
GetLogStatusValues	f	o
c1 QueryLogByTime ou QueryLogAfter doit être pris en charge		

18 Mapping du modèle d'événement de poste générique (GSE) – Événement de poste orienté objet générique (GOOSE)

18.1 Définition de commande GOOSE

La commande GOOSE, telle que définie dans l'IEC 61850-7-2, doit être mappée à un bloc de commande GOOSE (GoCB) tel que défini au Tableau 65. Les types d'attributs GoCB doivent être mappés comme indiqué en 8.3.

Tableau 65 – Définition de TypeDescription pour la structure de GoCB

Attributs GoCB	Type ACSI	lecture/écriture	o/f	Commentaires
GoEna	BOOLEAN	rw	o	
GoID	VISIBLE STRING	lecture	o	
DatSet	ObjectReference	lecture	o	Doit être limité à l'ensemble de données persistant (à l'intérieur ou à l'extérieur d'un dispositif logique)
ConfRev	INT32U	lecture	o	
NdsCom	BOOLEAN	lecture	o	
DstAddress	PhyComAddr	lecture	o	La structure PhyComAddr dépend du mapping utilisé pour la communication GOOSE (voir l'Article 18.2 et l'Article 18.3)
MinTime	INT32U	lecture	f	Comme spécifié dans le fichier SCD pour le GoCB
MaxTime	INT32U	lecture	f	Comme spécifié dans le fichier SCD pour le GoCB
FixedOffs	BOOLEAN	lecture	f	Comme spécifié dans le fichier SCD pour le GoCB

GoEna

Doit être tel que défini dans l'IEC 61850-7-2.

GoID

Doit être tel que défini dans l'IEC 61850-7-2. La valeur par défaut de cet attribut doit être la référence du bloc de commande GOOSE.

DatSet

La valeur doit être restreinte à tous les ensembles de données existants. L'écriture d'une valeur indiquant un ensemble de données inexistant doit échouer.

MinTime

Doit être comme défini dans l'IEC 61850-6, le délai d'envoi lors d'un changement de données entre le premier envoi immédiat du changement et la première répétition en ms.

MaxTime

Doit être comme défini dans l'IEC 61850-6, le temps de supervision source en ms (temps de cycle de supervision).

Pendant cette période, un message d'échec provenant de la source doit être détecté par le client.

18.2 Spécialisation pour le GoCB Couche 2

Tous les GoCB Couche 2 doivent être de contrainte fonctionnelle "GO".

L'attribut DstAddress doit être un type structuré dont les composants sont définis comme décrit au Tableau 4 de PhyComAddr pour la définition de communication Couche 2.

18.3 Spécialisation pour le GOOSE acheminable

Tous les GoCB acheminables doivent être de contrainte fonctionnelle "RG".

L'attribut DstAddress doit être un type structuré dont les composants sont définis comme décrit au Tableau 5 de PhyComAddr pour la définition de communication UDP/IP.

18.4 Services GOOSE

18.4.1 Généralités

Les services ACSI GetGoReference, GetGOOSEElementNumber et SendGOOSEMessage ne sont pas mappés par le présent SCSM, car un mapping XMPP ne satisferait pas aux performances exigées. Pour ces services, les mises en œuvre doivent utiliser les SCSM définis dans l'IEC 61850-8-1.

18.4.2 GetGoCBValues

Ce service s'apparente au GetBRCBValues (voir 17.2.2), mais la référence à lire est le GoCB (avec FC=GO ou RG) et la Response+ doit renvoyer les valeurs des attributs GoCB.

18.4.3 SetGoCBValues

Le mapping de SetGoCBValues ACSI s'apparente à celui de SetBRCBValues (voir 17.2.3), limité à l'attribut GoEna.

19 Transmission de modèle de classe de valeurs échantillonnées

19.1 Bloc de commande de valeur échantillonnée

19.1.1 Généralités

Les blocs de commande multidiffusion et envoi individuel doivent être tels que spécifiés dans l'IEC 61850-9-2, à l'exception de la contrainte fonctionnelle et de DstAddress.

19.1.2 Spécialisation pour la valeur échantillonnée Couche 2

19.1.2.1 Contrainte fonctionnelle

Tous les blocs de commande de valeur échantillonnée de la valeur échantillonnée Couche 2 doivent être de contrainte fonctionnelle "MS".

19.1.2.2 DstAddress

Il doit s'agir d'un type structuré dont les composants sont définis comme décrit dans la définition de PhyComAddr pour la communication Couche 2 (voir 8.3.3.3.1).

19.1.3 Spécialisation pour la valeur échantillonnée acheminable

19.1.3.1 Contrainte fonctionnelle

Tous les blocs de commande de valeur échantillonnée de la valeur échantillonnée acheminable doivent être de contrainte fonctionnelle "RS".

19.1.3.2 DstAddress

Il doit s'agir d'un type structuré dont les composants sont définis comme décrit dans la définition de PhyComAddr pour la communication UDP/IP (voir 8.3.3.2).

19.1.4 Spécialisation pour la valeur échantillonnée envoi individuel

19.1.4.1 Contrainte fonctionnelle

Tous les blocs de commande de valeur échantillonnée de la valeur échantillonnée acheminable doivent être de contrainte fonctionnelle "US".

19.1.4.2 DstAddress

Il doit s'agir d'un type structuré dont les composants sont définis comme décrit dans la définition de PhyComAddr pour la communication Couche 2 (voir 8.3.3.1).

19.2 Services de valeur échantillonnée

19.2.1 Généralités

Seuls les services ACSI pour l'obtention et la définition des attributs des blocs de commande connexes sont mappés dans le présent SCSM (voir le Tableau 1). Les services SendMSVMessage, GetMsvReference et GetMSVElementNumber ne sont pas mappés dans le présent document, car un mapping XMPP ne satisferait pas aux performances exigées.

19.2.2 GetMSVCBValues

Ce service s'apparente au GetBRCBValues (voir 17.2.2), mais la référence à lire est le MSVCB (avec FC=MS ou RS) et la Response doit renvoyer les valeurs des attributs MSVCB.

19.2.3 SetMSVCBValues

Le mapping de SetMSVCBValues ACSI s'apparente à celui de SetBRCBValues (voir 17.2.3).

20 Modèle de classe de commande

20.1 Généralités

Le principe consiste à mapper les paramètres des services de commande ACSI (tels que définis dans l'IEC 61850-7-2) à certains attributs de données dédiés (composant de commande) dans l'objet de données sous contrôle. Plus précisément, chaque service de commande consiste à écrire (ou lire) les sous-attributs de l'un de ces attributs de données dédiés, chaque sous-attribut correspondant à l'un des paramètres de service définis dans l'IEC 61850-7-2.

NOTE Seuls les objets de données contrôlables dont l'attribut de données ctrlModel n'est pas défini sur "status-only" sont concernés par le présent article.

La contrainte fonctionnelle pour ces attributs de données doit être "CO". Les références à utiliser pour accéder aux paramètres de commande d'un DO donné doivent donc commencer par <LDname>/<LNname>\$CO\$<DOname>. Pour des raisons pratiques, ce type de références est appelé CO_CtrlObjectRef dans les articles ci-après.

En fonction des services, l'un des composants de commande suivants (dans un CO_CtrlObjectRef) est écrit ou lu: OPER, CANCEL, SBO, SBOw. Chacun d'eux peut inclure les sous-attributs décrits au Tableau 66 pour le mapping des paramètres nécessaires.

Le Tableau 67 présente les composants de commande utilisés par les services de commande selon le type de modèles de commande.

Pour être suffisamment générique, le type du paramètre `ctlVal` n'est pas défini dans l'IEC 61850-7-2. Le type utilisé pour le sous-attribut `ctlVal` dans le composant de commande doit correspondre à celui de l'attribut `ctlVal` défini dans le CDC de l'objet sous contrôle.

Tableau 66 – Paramètres de service contrôlables

Paramètre IEC 61850-7-2	Sémantique	Type ACSI	Commentaire
<code>ctlVal</code>	Valeur de commande	Dépend de la CDC de l'objet de commande – voir l'IEC 61850-7-3	
<code>operTm</code>	Timestamp	Timestamp	Voir 8.3.3.7
<code>origin</code>	Origin	Structure ConstructedAttribute Originator – 8.3.3.16	
<code>ctlNum</code>	Numéro de commande	INT8U	Voir 8.3.1
<code>T</code>	Horodatage de commande	Timestamp	Voir 8.3.3.7
<code>Test</code>	Statut d'essai	BOOLEAN	Voir 8.3.1
<code>Check</code>	Condition de vérification	CheckConditions	Voir 8.3.3.6
<code>AddCause</code>	Diagnostic supplémentaire des causes	ENUMERATED	Voir 8.3.2.2

Tableau 67 – Mapping du modèle de commande de l'IEC 61850-7-2 aux composants de commande

Choix du service	DataAttributes	O/F/C
Direct avec sécurité normale		
	Oper	O
	Cancel	F
SBO avec sécurité normale		
	SBO	O
	Oper	O
	Cancel	O
SBO avec sécurité avancée		
	SBOw	O
	Oper	O
	Cancel	O
Direct avec sécurité avancée		
	Oper	O
	Cancel	F

20.2 Vue d'ensemble du mapping des services de commande

Le Tableau 68 donne une vue d'ensemble du mapping des services de commande.

Tableau 68 – Mapping des services de commande

Service ACSI		Type d'éléments XML	Spécification de variable	Résultat d'accès
Select	Request	Read-Request	SBO	
	Response+	Read-Response	CO_CtrlObjectRef	Succès
	Response–	Read-Response	Chaîne vide	Succès
SelectWithValue	Request	Write-Request	SBOw	
	Response+	Write-Response		Succès
	Response–	InformationReport (ListOfVariable)	LastApplError	
		Write-Response		Failure
Cancel	Request	Write-Request	Cancel	
	Response+	Write-Response		Succès
	Response–	InformationReport ^b (ListOfVariable)	LastApplError	
		Write-Response		Échec
Operate ou TimeActivatedOperate (operTm = 0)	Request	Write-Request	Oper	
	Response+	Write-Response		Succès
	Response–	InformationReport ^b (ListOfVariable)	LastApplError	
		Write-Response		Échec
TimeActivatedOperate (operTm != 0)	Request	Write-Request	Oper	
	Response+	Write-Response		Succès
	Response–	InformationReport ^b (ListOfVariable)	LastApplError	
		Write-Response		Échec
TimeActivatedOperateTermination (operTm != 0)	Request+	InformationReport (ListOfVariable)	Oper	
	Request–	InformationReport (ListOfVariable)	LastApplError	
CommandTermination	Request+ ^a	InformationReport (ListOfVariable)	Oper	
	Request– ^a	InformationReport (ListOfVariable)	LastApplError Oper	
^a Le cas échéant, la valeur d'operTm doit être zéro (0) dans le message CommandTermination.				
^b Facultatif pour la commande avec sécurité normale				

20.3 Select

La demande Select ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> comprenant un élément <read/> dans lequel l'élément <variableAccessSpecification/> contient un <variableSpecification/> faisant référence au composant SBO de <CO_CtrlObjectRef/>.

La Response+ de Select ACSI doit mapper vers un <confirmed-ResponsePDU/> contenant un élément <read/> avec un élément <success/> renvoyant la valeur <CO_CtrlObjectRef > en tant qu'élément <visible-string/>.

La Response- de Select ACSI doit mapper vers un <confirmed-ResponsePDU/> contenant un élément <read/> avec un élément <success/> renvoyant un <visible-string/> vide.

Le mapping des paramètres est tel que spécifié au Tableau 69.

Tableau 69 – Mapping des paramètres Select

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <read/>		Voir la structure de la demande XML à la Figure 73
ControlObjectReference	<variableSpecification/>	Voir 8.2.5	Restreint à <CO_CtrlObjectRef>=\$SBO
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		Voir la structure de la demande XML à la Figure 74
ControlObjectReference	<listOfAccessResult/> avec un élément <success/>	<CO_CtrlObjectRef>	Mappé en tant que <visible-string/>
Response-	<confirmed-ResponsePDU/> <ConfirmedServiceResponse/> <read/>		
ControlObjectReference	<listOfAccessResult/> avec un élément <success/>	visible-string vide	

```

<confirmed-RequestPDU>
  <invokeID>1874014</invokeID>
  <ConfirmedServiceRequest>
    <read specificationWithResult="false">
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <!-- Select = Read $SBO -->
                  <itemId>GGIO1$CO$DPCSO1$SBO</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
    </read>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 73 – Structure XML de Select-Request


```

<confirmed-ResponsePDU>
  <invokeID>1874014</invokeID>
  <ConfirmedServiceResponse>
    <read>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <visible-string>CONTROL/GGIO1$CO$DPCSO1</visible-string>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </read>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 74 – Structure XML de Select-Response

20.4 SelectWithValue

20.4.1 Mapping des paramètres de service SelectWithValue

Le service SelectWithValue doit être assuré en écrivant l'attribut SBOw de l'objet sous contrôle. Par conséquent, les paramètres de commande sont mappés aux sous-attributs de SBOw (voir le Tableau 70).

Tableau 70 – Mapping des paramètres de service SelectWithValue

Paramètre IEC 61850-7-2	Nom de composant IEC 61850-8-2	FC	TypeDefinition	o/f
ctlVal	<CO_CtrlObjectRef>\$SBOw\$ctlVal		Voir le Tableau 66	o
operTm	<CO_CtrlObjectRef>\$SBOw\$operTm		Voir le Tableau 66	c
origin	<CO_CtrlObjectRef>\$SBOw\$origin		Voir le Tableau 66	o
ctlNum	<CO_CtrlObjectRef>\$SBOw\$ctlNum		Voir le Tableau 66	o
T	<CO_CtrlObjectRef>\$SBOw\$T		Voir le Tableau 66	o
Test	<CO_CtrlObjectRef>\$SBOw\$Test		Voir le Tableau 66	o
Check	<CO_CtrlObjectRef>\$SBOw\$Check		Voir le Tableau 66	o
AddCause	/LastApplError		Voir 20.9	o
NOTE 1 Le type ctlVal dépend de la CDC de l'objet (ctlVal ou ctlVal\$, par exemple).				
NOTE 2 Le paramètre de service operTm doit uniquement être présent si l'objet de commande prend en charge TimeActivatedOperate.				

20.4.2 Mapping général du service SelectWithValue

La demande SelectWithValue ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> comprenant un élément <write/> dans lequel l'élément <variableAccessSpecification/> contient un <variableSpecification/> faisant référence au composant SBOw de <CO_CtrlObjectRef>. L'élément <listOfData/> doit contenir une valeur pour tous les sous-attributs du SBOw figurant au Tableau 70.

La Response+ de SelectWithValue ACSI doit mapper vers un <confirmed-ResponsePDU/> contenant un élément <write/> avec un élément <success/>.

La Response- de SelectWithValue ACSI doit mapper vers deux messages (voir 20.4.3).

Le mapping des paramètres est tel que spécifié au Tableau 71.

Tableau 71 – Mapping des paramètres SelectWithValue

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		Voir la structure de la demande XML à la Figure 75
ControlObjectReference	<variableSpecification/>	Voir 8.2.5	Restreint à <CO_ CtrlObjectRef>\$SBOW
ctlVal	<listOfData/>	Voir l'exemple de la Figure 75	Valeur de chaque sous-attribut de SBOW (qui mappent les paramètres définis dans l'IEC 61850-7-2)
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse /> <write/>		Voir la structure de la réponse XML à la Figure 76
ControlObjectReference	Un élément <success/>		Les valeurs de paramètre sont éventuellement disponibles en tant que suivi du service contrôlable
ctlVal			
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
Response-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		Pour informationReport, voir 20.4.3 et 20.9
ControlObjectReference	<variableAccessSpecification />		/LastApplError (voir 20.9)
ctlVal	et <confirmed-ResponsePDU/> <ConfirmedServiceResponse /> <write/> <CHOICE/> <failure/>		Pour Write-Response, voir 20.4.3
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
AddCause			

^a Le paramètre de service operTm doit uniquement être présent si l'objet de commande prend en charge TimeActivatedOperate, et est donc mappé à <CO_ControlObjectReference>\$SBOW\$OperTm.

```

<confirmed-RequestPDU>
  <invokeID>89351</invokeID>
  <ConfirmedServiceRequest>
    <write>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1$CO$DPCSO1$SBOW</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfData>
        <Data>
          <structure>
            <Data>
              <boolean>true</boolean>
            </Data>
            <Data>
              <structure>
                <Data>
                  <integer>2</integer>
                </Data>
                <Data>
                  <octet-string>53544154494F4E</octet-string>
                </Data>
              </structure>
            </Data>
            <Data>
              <unsigned>25</unsigned>
            </Data>
            <Data>
              <utc-time>017215E00000001F</utc-time>
            </Data>
            <Data>
              <boolean>>false</boolean>
            </Data>
            <Data>
              <bit-string>00</bit-string>
            </Data>
          </structure>
        </Data>
      </listOfData>
    </write>
  </ConfirmedServiceRequest>
</confirmed-RequestPDU>

```

Figure 75 – Structure XML de SelectWithValue-Request

```

<confirmed-ResponsePDU>
  <invokeID>89351</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <success/>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 76 – Structure XML de SelectWithValue-Response+

20.4.3 Response- de SelectWithValue

La Response- de SelectWithValue IEC 61850-7-2 mappe vers deux messages envoyés au client par le serveur. L'ordre des messages doit être le suivant:

- <unconfirmed-PDU/> contenant un <informationReport/> avec AdditionalCauseDiagnostic (voir 20.9).
- <confirmed-ResponsePDU/> correspondant à la demande SelectWithValue, indiquant <failure/> comme enfant indirect de l'élément <write/> et les valeurs de DataAccessError telles que définies au Tableau 72 (voir l'exemple de la Figure 77).

Tableau 72 – Spécification de SelectWithValue, d'Oper et de Cancel DataAccessError

Erreur ACSI	DataAccessError	Condition d'erreur
TEMPORARILY-UNAVAILABLE	TEMPORARILY UNAVAILABLE	La commande est déjà sélectionnée ou en cours de fonctionnement
HARDWARE-FAULT	HARDWARE-FAULT	La commande peut ne pas fonctionner en raison d'un défaut matériel
OBJECT-ACCESS-DENIED	OBJECT-ACCESS-DENIED	Échec de la commande d'accès
OBJECT-UNDEFINED	OBJECT-UNDEFINED	La commande n'existe pas dans cette vue de sécurité

```

<confirmed-ResponsePDU>
  <invokeID>89351</invokeID>
  <ConfirmedServiceResponse>
    <write>
      <CHOICE>
        <failure>
          <temporarily-unavailable/>
        </failure>
      </CHOICE>
    </write>
  </ConfirmedServiceResponse>
</confirmed-ResponsePDU>

```

Figure 77 – Structure XML de SelectWithValue-Response-

20.5 Cancel

20.5.1 Mapping des paramètres de service Cancel

Le service Cancel doit être assuré en écrivant l'attribut Cancel de l'objet sous contrôle. Par conséquent, les paramètres de commande sont mappés aux sous-attributs de Cancel (voir le Tableau 73).

Tableau 73 – Mapping des paramètres de service Cancel

Paramètre IEC 61850-7-2	Nom de composant IEC 61850-8-1	FC	TypeDefinition	o/ f
	<CO_CtrlObjectRef>\$Cancel	CO		
ctlVal	<CO_CtrlObjectRef>\$Cancel\$ctlVal		Voir le Tableau 66	o
operTm	<CO_CtrlObjectRef>\$Cancel\$operTm		Voir le Tableau 66	c
origin	<CO_CtrlObjectRef>\$Cancel\$origin		Voir le Tableau 66	o
ctlNum	<CO_CtrlObjectRef>\$Cancel\$ctlNum		Voir le Tableau 66	o
T	<CO_CtrlObjectRef>\$Cancel\$T		Voir le Tableau 66	o
Test	<CO_CtrlObjectRef>\$Cancel\$Test		Voir le Tableau 66	o
AddCause	/LastApplError		Voir 20.9	o
NOTE 1 Le type ctlVal dépend de la CDC de l'objet (ctlVal ou ctlVal\$f, par exemple).				
NOTE 2 Le paramètre de service operTm doit uniquement être présent si l'objet de commande prend en charge TimeActivatedOperate.				

20.5.2 Mapping général du service Cancel

La demande Cancel ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> comprenant un élément <write/> dans lequel l'élément <variableAccessSpecification/> contient un <variableSpecification/> faisant référence au composant Cancel de <CO_CtrlObjectRef>. L'élément <listOfData/> doit contenir une valeur pour tous les sous-attributs de Cancel figurant au Tableau 73.

La Response+ de Cancel ACSI doit mapper vers un <confirmed-ResponsePDU/> contenant un élément <write/> avec un élément <success/>.

La Response- de Cancel ACSI doit mapper vers un ou deux message(s) (voir 20.5.3).

Le mapping des paramètres est tel que spécifié au Tableau 74.

Tableau 74 – Mapping du service Cancel

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		S'apparente à la structure de la demande XML pour SelectWithValue de la Figure 75, mais sans valeur pour le dernier paramètre (Check) qui n'existe pas pour Cancel
ControlObjectReference	<variableSpecification/>	Voir 8.2.5	Restreint à <CO_CtrlObjectRef>\$Cancel
ctlVal	<listOfData/>	S'apparente à l'exemple de la Figure 75 (mais sans la dernière valeur bit-string)	Valeur de chaque sous-attribut de Cancel (qui mappent les paramètres définis dans l'IEC 61850-7-2)
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse /> <write/>		Voir la structure de la réponse XML à la Figure 76
ControlObjectReference	Un élément <success/>		Les valeurs de paramètre sont éventuellement disponibles en tant que suivi du service contrôlable
ctlVal			
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Response-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		Pour informationReport ^b , voir 20.5.3 et 20.9.
ControlObjectReference	<variableAccessSpecification />		/LastApplError (voir 20.9)
ctlVal	et <confirmed-ResponsePDU/> <ConfirmedServiceResponse /> <write/> <CHOICE/> <failure/>		Pour Write-Response, voir 20.5.3
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
AddCause			
^a Le paramètre de service operTm doit uniquement être présent si l'objet de commande prend en charge TimeActivatedOperate, et est donc mappé à <CO_ControlObjectReference>\$Cancel\$OperTm.			
^b Facultatif pour les modèles de commande avec sécurité normale. Sinon, obligatoire			

20.5.3 Cancel response–

La Response- de Cancel IEC 61850-7-2 mappe vers un ou deux message(s) envoyé(s) au client par le serveur. L'ordre des messages doit être le suivant:

- <unconfirmed-PDU/> contenant un <informationReport/> avec AdditionalCauseDiagnostic (facultatif pour la sécurité normale et obligatoire pour les modèles de commande à sécurité avancée) (voir 20.9).
- <confirmed-ResponsePDU/> correspondant à la demande Cancel, indiquant <failure/> comme enfant indirect de l'élément <write/> et les valeurs de DataAccessError telles que définies au Tableau 72 (voir l'exemple de la Figure 77).

20.6 Operate

20.6.1 Mapping du paramètre de service Operate

Le service Operate doit être assuré en écrivant l'attribut Oper de l'objet sous contrôle. Par conséquent, les paramètres de commande sont mappés aux sous-attributs de Oper (voir le Tableau 75).

Tableau 75 – Mapping du paramètre de service Operate

Paramètre IEC 61850-7-2	Nom de composant IEC 61850-8-1	FC	TypeDefinition	o/f
	<CO_CtrlObjectRef>\$Oper	CO		
ctlVal	<CO_CtrlObjectRef>\$Oper\$ctlVal		Voir le Tableau 66	o
operTm	<CO_CtrlObjectRef>\$Oper\$operTm		Voir le Tableau 66	c
origin	<CO_CtrlObjectRef>\$Oper\$origin		Voir le Tableau 66	o
ctlNum	<CO_CtrlObjectRef>\$Oper\$ctlNum		Voir le Tableau 66	o
T	<CO_CtrlObjectRef>\$Oper\$T		Voir le Tableau 66	o
Test	<CO_CtrlObjectRef>\$Oper\$Test		Voir le Tableau 66	o
Check	<CO_CtrlObjectRef>\$Oper\$Check		Voir le Tableau 66	o
AddCause	/LastAppError		Voir 20.9	o
NOTE 1 Le type ctlVal dépend de la CDC de l'objet (ctlVal ou ctlVal\$f, par exemple).				
NOTE 2 Le paramètre de service operTm doit uniquement être présent si l'objet de commande prend en charge TimeActivatedOperate.				

20.6.2 Mapping général du service Operate

La demande Operate ACSI doit mapper vers un élément XML <confirmed-RequestPDU/> comprenant un élément <write/> dans lequel l'élément <variableAccessSpecification/> contient un <variableSpecification/> faisant référence au composant Oper de <CO_CtrlObjectRef>. L'élément <listOfData/> doit contenir une valeur pour tous les sous-attributs de l'Oper figurant au Tableau 75.

La Response+ d'Operate ACSI doit mapper vers un <confirmed-ResponsePDU/> contenant un élément <write/> avec un élément <success/>.

La Response- d'Operate ACSI doit mapper vers un ou deux message(s) (voir 20.6.3 et 20.9).

Le mapping des paramètres est tel que spécifié au Tableau 76.

Tableau 76 – Mapping du service Operate

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request	<confirmed-RequestPDU/> <ConfirmedServiceRequest/> <write/>		S'apparente à la structure de la demande XML pour SelectWithValue de la Figure 75
ControlObjectReference	<variableSpecification/>	Voir 8.2.5	Restreint à <CO_ CtrlObjectRef >\$Oper
ctlVal	<listOfData/>	S'apparente à l'exemple de la Figure 75	Valeur de chaque sous-attribut d'Oper (qui mappent les paramètres définis dans l'IEC 61850-7-2)
operTm[0..1] ^a			
Origin			
ctlNum			
T			
Test			
Check			
Response+	<confirmed-ResponsePDU/> <ConfirmedServiceResponse /> <write/>		Voir la structure de la réponse XML à la Figure 76
ControlObjectReference	Un élément <success/>		Les valeurs de paramètre sont éventuellement disponibles en tant que suivi du service contrôlable
ctlVal			
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
Response-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		Pour informationReport ^b , voir 20.6.3 et 20.9.
ControlObjectReference	<variableAccessSpecification />		/LastApplError (voir 20.9)
ctlVal	et <confirmed-ResponsePDU/> <ConfirmedServiceResponse /> <write/> <CHOICE/> <failure/>		Pour Write-Response, voir 20.6.3
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
AddCause			

^a Le paramètre de service operTm doit uniquement être présent si l'objet de commande prend en charge TimeActivatedOperate, et est donc mappé à <CO_CtrlObjectRef>\$Oper\$OperTm>. Sa valeur doit être zéro (0) pour le service Operate.

^b Les mises en œuvre client doivent savoir qu'elles peuvent ne pas recevoir le rapport d'informations dans le contexte des passerelles avec d'autres SCSM. Toutefois, les serveurs conformes au présent SCSM doivent le mettre en œuvre.

20.6.3 Operate response–

La Response- d'Operate IEC 61850-7-2 mappe vers un ou deux message(s) envoyé(s) au client par le serveur. L'ordre des messages doit être le suivant:

- <unconfirmed-PDU/> contenant un <informationReport/> avec AdditionalCauseDiagnostic (facultatif pour la sécurité normale et obligatoire pour les modèles de commande à sécurité avancée) (voir 20.9).
- <confirmed-ResponsePDU/> correspondant à la demande Operate, indiquant <failure/> comme enfant indirect de l'élément <write/> et les valeurs de DataAccessError telles que définies au Tableau 72 (voir l'exemple de la Figure 77).

CommandTermination

20.6.4 Mapping des paramètres de service CommandTermination

Aucun des paramètres de service de l'IEC 61850-7-2 n'est nécessaire en plus de celui spécifié pour le service Operate / TimeActivatedOperate.

20.6.5 Mapping général du service CommandTermination

La Request+ de CommandTermination doit être mappée vers un élément XML <unconfirmed-PDU/> contenant un élément <informationReport/> (voir l'exemple de la Figure 78).

La partie <variableAccessSpecification/> de cet <informationReport/> doit mapper vers la référence au composant Oper de l'objet sous contrôle (c'est-à-dire <CO_CtrlObjectRef>\$Oper), la partie <listOfAccessResult/> devant mapper vers les valeurs du même composant Oper.

La valeur de l'attribut operTm de la Request+ de CommandTermination d'une commande avec TimeActivation doit être zéro (0).

La Request- de CommandTermination doit être mappée vers un élément XML <unconfirmed-PDU/> contenant un élément <informationReport/> contenant deux variables (voir l'exemple de la Figure 79). La première variable est la variable spécifique vmd "LastApplError" (voir 20.9). La seconde variable est le composant Oper de l'objet sous contrôle, comme pour Request+.

Le service CommandTermination doit être mappé comme indiqué au Tableau 77.

Tableau 77 – Mapping du service CommandTermination

Paramètres tels que spécifiés dans l'IEC 61850-7-2	Élément XML	Valeur	Commentaire
Request+	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		Voir la structure de la demande XML à la Figure 78
ControlObjectReference	<variableSpecification/>	Voir 8.2.5	Restreint à <CO_ CtrlObjectRef >\$Oper
ctlVal	<listOfAccessResult/>	Voir l'exemple de la Figure 78	Valeur de chaque sous-attribut d'Oper (qui mappent les paramètres définis dans l'IEC 61850-7-2)
operTm[0..1] ^a			
Origin			
ctlNum			
T			
Test			
Check			
Request-	<unconfirmed-PDU/> <UnconfirmedService/> <informationReport/>		Voir la structure de la demande XML à la Figure 79
ControlObjectReference	<variableSpecification/>	Voir 8.2.3 et 8.2.5	/LastApplError + <CO_ CtrlObjectRef>\$Oper
ctlVal	<listOfAccessResult/>	Voir l'exemple de la Figure 79	En premier lieu, l'élément <success/> avec les valeurs pour LastApplError, puis l'élément <success/> avec des valeurs pour chaque sous-attribut d'Oper (qui mappent les paramètres définis dans l'IEC 61850-7-2)
operTm[0..1] ^a			
origin			
ctlNum			
T			
Test			
Check			
AddCause			

^a Le paramètre de service operTm doit uniquement être présent si l'objet de commande prend en charge TimeActivatedOperate, et est donc mappé à <CO_ControlObjectReference>\$Oper\$OperTm. Sa valeur doit être zéro (0) pour le service Operate.

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <domain-specific>
                  <domainId>CONTROL</domainId>
                  <itemId>GGIO1$CO$DPCSO1$Oper</itemId>
                </domain-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <boolean>true</boolean>
              </Data>
              <Data>
                <structure>
                  <Data>
                    <integer>2</integer>
                  </Data>
                  <Data>
                    <octet-string>53544154494F4E</octet-string>
                  </Data>
                </structure>
              </Data>
              <Data>
                <unsigned>25</unsigned>
              </Data>
              <Data>
                <utc-time>017215E00000001F</utc-time>
              </Data>
              <Data>
                <boolean>>false</boolean>
              </Data>
              <Data>
                <bit-string>00</bit-string>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </informationReport>
  </UnconfirmedService>
</unconfirmed-PDU>

```

Figure 78 – Structure XML de la Request+ de CommandTermination

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <vmd-specific>LastApplError</vmd-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        <SEQUENCE>
          <variableSpecification>
            <name>
              <domain-specific>
                <domainId>CONTROL</domainId>
                <itemId>GGIO1$CO$DPCSO1$Oper</itemId>
              </domain-specific>
            </name>
          </variableSpecification>
        </SEQUENCE>
      </listOfVariable>
    </variableAccessSpecification>
  <listOfAccessResult>
    <AccessResult>
      <success>
        <structure>
          <Data>
            <visible-string>
              CONTROL/GGIO1$CO$DPCSO1$Oper
            </visible-string>
          </Data>
          <Data>
            <integer>3</integer>
          </Data>
          <Data>
            <structure>
              <Data>
                <integer>2</integer>
              </Data>
              <Data>
                <octet-string>53544154494F4E</octet-string>
              </Data>
            </structure>
          </Data>
          <Data>
            <unsigned>25</unsigned>
          </Data>
          <Data>
            <integer>3</integer>
          </Data>
        </structure>
      </success>
    </AccessResult>
  <AccessResult>
    <success>
      <structure>
        <Data>
          <boolean>true</boolean>
        </Data>
        <Data>

```

```

<structure>
  <Data>
    <integer>2</integer>
  </Data>
  <Data>
    <octet-string>53544154494F4E</octet-string>
  </Data>
</structure>
</Data>
<Data>
  <unsigned>25</unsigned>
</Data>
<Data>
  <utc-time>017215E00000001F</utc-time>
</Data>
<Data>
  <boolean>>false</boolean>
</Data>
<Data>
  <bit-string>00</bit-string>
</Data>
</structure>
</success>
</AccessResult>
</listOfAccessResult>
</informationReport>
</UnconfirmedService>
</unconfirmed-PDU>

```

Figure 79 – Structure XML de la Request- de CommandTermination

20.7 TimeActivatedOperate

20.7.1 Mapping des paramètres de service TimeActivatedOperate

Aucun des paramètres de service de l'IEC 61850-7-2 n'est nécessaire en plus de celui spécifié pour le service Operate.

20.7.2 Mapping du service TimeActivatedOperate

Le mapping du service TimeActivatedOperate doit s'apparenter à celui d'Operate, mais le paramètre OperTm doit être présent et doit contenir un horodatage valide. La valeur OperTm = 0 (c'est-à-dire, avec tous les octets définis sur 0) équivaut à une commande avec sécurité avancée (c'est-à-dire sans activation temporelle).

20.8 Service TimeActivatedOperateTermination

Le mapping du service TimeActivatedOperateTermination doit être similaire à celui de CommandTermination, avec les différences suivantes:

- Dans la Request+, la valeur de l'attribut operTm doit être la même que celle écrite dans la demande TimeActivatedOperate. Si la valeur d'OperTm est zéro (0) (tous les octets définis sur 0), cela doit indiquer le CommandTermination.
- La Request- doit contenir uniquement la première variable: la variable spécifique vmd "LastApplError" (voir 20.9).

20.9 AdditionalCauseDiagnosis dans les réponses de service de commande négatives

Certains services de commande abstraits donnent des informations supplémentaires spécifiques à l'application dans leurs réponses négatives. Ces informations sont définies par le paramètre de service AdditionalCauseDiagnosis (AddCause) défini dans l'IEC 61850-7-2. La transmission de ces informations doit mapper vers un élément XML <unconfirmed-PDU/> incluant un élément <informationReport/> contenant la variable spécifique vmd "LastApplError" de la structure définie au Tableau 78 (voir l'exemple du message XML obtenu à la Figure 80). Cette variable doit être créée instantanément, rapportée, puis supprimée.

Tableau 78 – Définition de la structure de variable LastApplError

Nom de composant	TypeDescription ACSI	lecture/écriture	o/f	Commentaires
CntrlObj	VISIBLE STRING	lecture	o	
Error	ENUMERATED	lecture	o	
Origin	Originator	lecture	o	Voir l'IEC 61850-7-2
ctlNum	INT8U	lecture	o	Voir l'IEC 61850-7-2
AddCause	ENUMERATED	lecture	o	

IECNORM.COM : Click to view the full PDF of IEC 61850-8-2:2018

```

<unconfirmed-PDU>
  <UnconfirmedService>
    <informationReport>
      <variableAccessSpecification>
        <listOfVariable>
          <SEQUENCE>
            <variableSpecification>
              <name>
                <vmd-specific>LastApplError</vmd-specific>
              </name>
            </variableSpecification>
          </SEQUENCE>
        </listOfVariable>
      </variableAccessSpecification>
      <listOfAccessResult>
        <AccessResult>
          <success>
            <structure>
              <Data>
                <visible-string>
                  CONTROL/GGIO1$CO$DPCSO1$Oper
                </visible-string>
              </Data>
              <Data>
                <integer>3</integer>
              </Data>
              <Data>
                <structure>
                  <Data>
                    <integer>2</integer>
                  </Data>
                  <Data>
                    <octet-string>53544154494F4E</octet-string>
                  </Data>
                </structure>
              </Data>
              <Data>
                <unsigned>25</unsigned>
              </Data>
              <Data>
                <integer>3</integer>
              </Data>
            </structure>
          </success>
        </AccessResult>
      </listOfAccessResult>
    </informationReport>
  </UnconfirmedService>
</unconfirmed-PDU>

```

Figure 80 – Structure XML d'InformationReport avec AdditionalCauseDiagnosis

La valeur de la variable doit représenter la dernière erreur d'application de service de commande détectée qui a nécessité des diagnostics supplémentaires. La valeur de LastApplError doit être rapportée sous la forme d'une seule variable (la variable structurée doit être rapportée, par exemple). La valeur de cette variable doit être volatile et doit accepter l'utilisation des valeurs par défaut spécifiées en cas d'établissement d'association bipartite.

CntrlObj

CntrlObj doit avoir un type de données VISIBLE STRING de la structure

<LDeviceName>/<LNVariableName>\${<FC>}\${<LNDataName1>}\${<AttributeName1>
(voir 8.3.3.2)

et doit avoir une taille maximale de 129 octets. La valeur doit représenter l'objet de commande qui a été appliqué et a provoqué la génération de la cause additionnelle.

La valeur par défaut doit être NULL.

Error

Error doit avoir un type de données ENUMERATED comme défini en 8.3.2.2. Les valeurs énumérées pour une erreur doivent être définies comme suit:

- (0) No Error
- (1) Unknown
- (2) Timeout Test Not OK
- (3) Operator Test Not OK

La valeur par défaut doit être No Error (0) (Pas d'erreur). La valeur "No Error" (0) (Pas d'erreur) doit être retournée lorsque l'accusé de réception négatif n'est pas émis par des résultats d'essai négatifs dans l'automate de commande pour un modèle de commande donné (souvent pour des valeurs addCause qui codent l'erreur d'application: Time-limit-over, Position-reached, 1-of-n-control, ...).

"Unknown" (Inconnu) doit être retourné lorsque l'accusé de réception négatif n'est pas émis par des résultats d'essai négatifs dans l'automate de commande pour un modèle de commande donné et qu'une erreur interne s'est produite.

"TimeOut Test Not OK" (Problème d'essai de délai d'attente) doit être retourné si une commande TimeActivated est émise avec TEST=TRUE, et que l'opération échoue.

"Operator Test Not OK" (Problème d'essai d'opérateur) doit être retourné si une commande Operate est émise avec TEST=TRUE, et que l'opération échoue.

origin

Utilisé pour désigner le client qui a initié l'action de commande.

ctlNum

Le numéro de séquence de commande comme indiqué par le client initiant le service.

NOTE Les réponses de commande négatives sont la plupart du temps composées d'un <confirmed-ResponsePDU/> contenant un élément <write/> indiquant l'échec de la demande de service Control (mappée à un Write-Request) et d'une demande InformationReport supplémentaire définie ci-dessus. La transmission d'origin et de ctlNum dans la demande InformationReport permet au client d'assigner cette demande InformationReport à la demande de service de commande correspondante.

La valeur par défaut doit être 0.

AddCause

AddCause doit être un type de données ENUMERATED comme défini en 8.3.2.2. Le Tableau 79 définit les valeurs réelles attribuées à AddCause.

Tableau 79 – Mapping des valeurs AddCause ACSI

Valeur ACSI	Valeur
Unknown	0
Not-supported	1
Blocked-by-switching-hierarchy	2
Select-failed	3
Invalid-position	4
Position-reached	5
Parameter-change-in-execution	6
Step-limit	7
Blocked-by-Mode	8
Blocked-by-process	9
Blocked-by-interlocking	10
Blocked-by-synchrocheck	11
Command-already-in-execution	12
Blocked-by-health	13
1-of-n-control	14
Abortion-by-cancel	15
Time-limit-over	16
Abortion-by-trip	17
Object-not-selected	18
Object-already-selected	19
No-access-authority	20
Ended-with-overshoot	21
Abortion-due-to-deviation	22
Abortion-by-communication-loss	23
Blocked-by-command	24
Aucun	25
Inconsistent-parameters	26
Locked-by-other-client	27

20.10 Mapping des services de commande

20.10.1 Généralités

Le modèle ServiceTracking des services de commande mappe directement vers une spécialisation de données.

Il est important de noter que l'attribut de données de type "object reference" (référence d'objet) est mappé vers VISIBLE STRING 129 et qu'il doit contenir la référence d'objet ACSI comme indiqué dans l'IEC 61850-7-2 et ne pas contenir l'adresse d'objet (voir 8.3.3.2.3).

NOTE Le mapping de la CDC CST n'est pas défini dans le présent SCSSM. Il peut être directement déduit de l'article général décrivant le mapping des attributs de données (voir 8.3).

20.10.2 Mapping du Control Service Tracking (CTS)

Le mapping de la CDC CTS doit correspondre à la définition spécifiée au Tableau 80.

Tableau 80 – Mapping de la CDC CTS

Nom d'attribut ACSI	Nom de composant	Type ACSI	Commentaire
Doit hériter de tous les composants du CST CDC			
Spécifique au CTS			
ctlVal	ctlVal	Voir l'Article 8.3.	En fonction de la CDC de l'objet de commande – voir l'IEC 61850-7-3
operTm	operTm	Timestamp	Voir 8.3.3.7
origin	origin	Structure – voir 8.3.3.16	Structure ConstructedAttribute Originator – voir l'IEC 61850-7-2
ctlNum	ctlNum	INT8U	Voir 8.3.1
T	T	Timestamp	Voir 8.3.3.7
Test	Test	BOOLEAN	Voir 8.3.1
Check	Check	CheckConditions	Voir 8.3.3.15
respAddCause	respAddCause	ENUMERATED	Voir 8.3.2.2 et Tableau 79

21 Modèle de temps et de synchronisation temporelle

La synchronisation temporelle doit être effectuée en utilisant SNTP (voir 5.3).

Les mécanismes de synchronisation matériels (GPS ou autre, par exemple) dépassent le domaine d'application de la présente norme.

22 Conventions de dénomination

Voir l'Article 8.2.

23 Transfert de fichier

23.1 Modèle de transfert de fichier

Le principe est que la classe de fichier IEC 61850-7-2 doit être mappée à l'objet fichier MMS. Toutefois, dans le contexte de ce SCSM, la connaissance de l'objet fichier MMS n'est pas une condition préalable. Si une mise en œuvre déclare la prise en charge du transfert de fichier, elle doit être mise en œuvre conformément au présent Article. De plus, la mise en œuvre du protocole de transfert de fichier IETF (FTP) ou du protocole de transfert de fichier sécurisé (SFTP) est définie localement. La conformité au RFC FTP est hors du domaine d'application de la présente norme.

Le mapping est spécifié au Tableau 81.

Tableau 81 – Mapping de la classe fichier ACSI à l'objet fichier MMS

Nom d'attribut de classe de fichier IEC 61850-7-2		Attribut d'objet fichier MMS		PresCond
Attribut	Type	Attribut	Type	
FileName	VisString255	FileName	FileName	O
FileSize	INT32U	Size	Unsigned	O
LastModified	TimeStamp	LastModified	GeneralizedTime	O

FileName

L'attribut FileName de l'IEC 61850-7-2 doit être mappé à l'attribut FileName MMS. Le FileName MMS doit être composé d'une séquence de noms de répertoire et d'un nom de fichier (name-of-a-file). Les FileNames de l'IEC 61850-7-2 doivent être limités à 255 octets au maximum.

La spécification de chemin d'accès composée d'une séquence de noms de répertoire de fichiers est obligatoire dans le nom de fichier. Les noms de répertoires de fichiers doivent être limités à une taille inférieure à 32 octets. Les noms de répertoire doivent être séparés par le caractère déterminé par PIXIT (généralement "\" ou "/").

Un serveur qui contient des fichiers associés à des dispositifs logiques doit avoir, en tant qu'un de ses répertoires racines, un répertoire dont le nom est "LD". Au-dessous du répertoire racine LD doit être situé un ensemble de noms de répertoire qui représentent les dispositifs logiques dans le serveur. Ces noms de répertoire de fichiers doivent avoir la même valeur que les noms de dispositif logique (les noms des domaines, par exemple) et doivent être présents s'il existe des fichiers. Des fichiers qui sont spécifiques à un dispositif logique doivent être situés dans le répertoire qui représente le dispositif logique.

NOTE Les fichiers ne peuvent pas tous être associés à un LD particulier. Par exemple, le fichier d'image du serveur lui-même ne peut pas être directement associé à un LD. Par conséquent, pour les fichiers non associés à un LD, le répertoire de stockage de tels fichiers est défini de manière locale.

La taille minimale-maximale de name-of-a-file doit être de 12 octets. Il est recommandé que la spécification de fichier soit limitée à une taille maximale de 64 octets. La taille maximale de la spécification de fichier doit être indiquée dans la déclaration PIXIT de la mise en œuvre.

Il convient d'utiliser des suffixes FileName afin de différencier le format du contenu des fichiers. Il convient que le suffixe soit au maximum de 3 octets. Les suffixes du Tableau 82 sont réservés.

Il est recommandé qu'un FileName retourné, c'est-à-dire un répertoire sur le serveur, soit terminé par un délimiteur déterminé par PIXIT ("\\" ou "/").

Mappings de FILE ServiceError vers les erreurs de service ACSI

Les services de fichier utilisent des codes d'erreur spécialisés renvoyés dans les éléments <serviceError/> avec <errorClass/><file/>. Le Tableau 84 présente les mappings appropriés entre les valeurs d'erreur de service ACSI et les ServiceError pour les opérations File.

Tableau 82 – Suffixes de fichier réservés

Suffixe	Contenu
Bin	Doit indiquer un format binaire.
Dtd	Fichier eXtensible Markup Language Document Type Description de format dtd:XML
Gif	Doit indiquer un contenu au format Graphics Interchange Format
Htm	Doit indiquer un contenu HTML
Pqd	Format Power Quality Data Interchange Format – PQDIF
txt	Doit indiquer un contenu au format ASCII
Xml	Fichier eXtensible Markup Language au format table-xml:XML
Xsd	Fichier de définition de schéma eXtensible Markup Language au format xsd:XML
Zip	Doit indiquer un format de fichier compressé

Le respect des majuscules/minuscules dans FileName doit être spécifié dans la déclaration PIXIT de la mise en œuvre

FICHIERS COMTRADE

Les fichiers IEC 60255-24:2013 / IEEE C37.111-2013 ou IEEE C37.111.1999 (COMTRADE) doivent être contenus dans un répertoire de fichiers dont le nom est "COMTRADE". Les spécifications de fichier doivent être cohérentes avec les conventions de dénomination et les suffixes spécifiés dans l'IEC 60255-24:2013 resp. IEEE C37.111:1999.

La spécification IEC COMTRADE IEC 60255-24:2013 spécifie l'utilisation de cinq suffixes différents (cff, inf, hdr, cfg et dat, par exemple). Dans l'utilisation pour des applications d'informations/informatiques normales, ces suffixes peuvent représenter des fichiers autres que COMTRADE.

La spécification IEEE COMTRADE IEEE C37.111:1999 spécifie l'utilisation de trois suffixes différents (hdr, cfg et dat, par exemple). Dans l'utilisation pour des applications d'informations/informatiques normales, ces suffixes peuvent représenter des fichiers autres que COMTRADE.

Si le répertoire contient un fichier avec un suffixe "zip", ce fichier doit comprendre le contenu compressé des fichiers COMTRADE hdr, cfg et dat des fichiers de même nom. Le fichier peut éventuellement contenir les fichiers hdr et/ou inf associés.

Les répertoires COMTRADE doivent être situés dans le chemin d'accès de répertoire approprié (dans le répertoire LD ou au niveau racine, par exemple).

FileSize

Une valeur 0 peut être utilisée pour indiquer que la taille du fichier est inconnue ou que le fichier contient 0 octet.

La valeur de taille doit être interprétée en tant que taille de fichier estimée et ne doit pas être utilisée pour déterminer la taille absolue.

NOTE Les transferts d'un système de fichiers à un autre peuvent augmenter ou diminuer la taille réelle du fichier.

LastModified

Les mappes vers un xsd:dateTime.Local ou un temps UTC sont admis, même si le temps UTC est le format recommandé.

23.2 Services de fichier

23.2.1 GetFile

23.2.1.1 Généralités

Le service GetFile ACSI doit être mappé à une séquence de services unitaires: un FileOpen, un ou plusieurs FileRead et un FileClose. Chacun de ces services unitaires doit être composé d'un échange de deux messages XML, l'un pour la demande contenant un élément XML <confirmed-RequestPDU/> avec respectivement des éléments de type FileOpen-Request, FileRead-Request ou FileClose-Request à l'intérieur, et un second pour la confirmation contenant un élément XML <confirmed-ResponsePDU/> avec des éléments de type FileOpen-Response, FileRead-Response ou FileClose-Response.

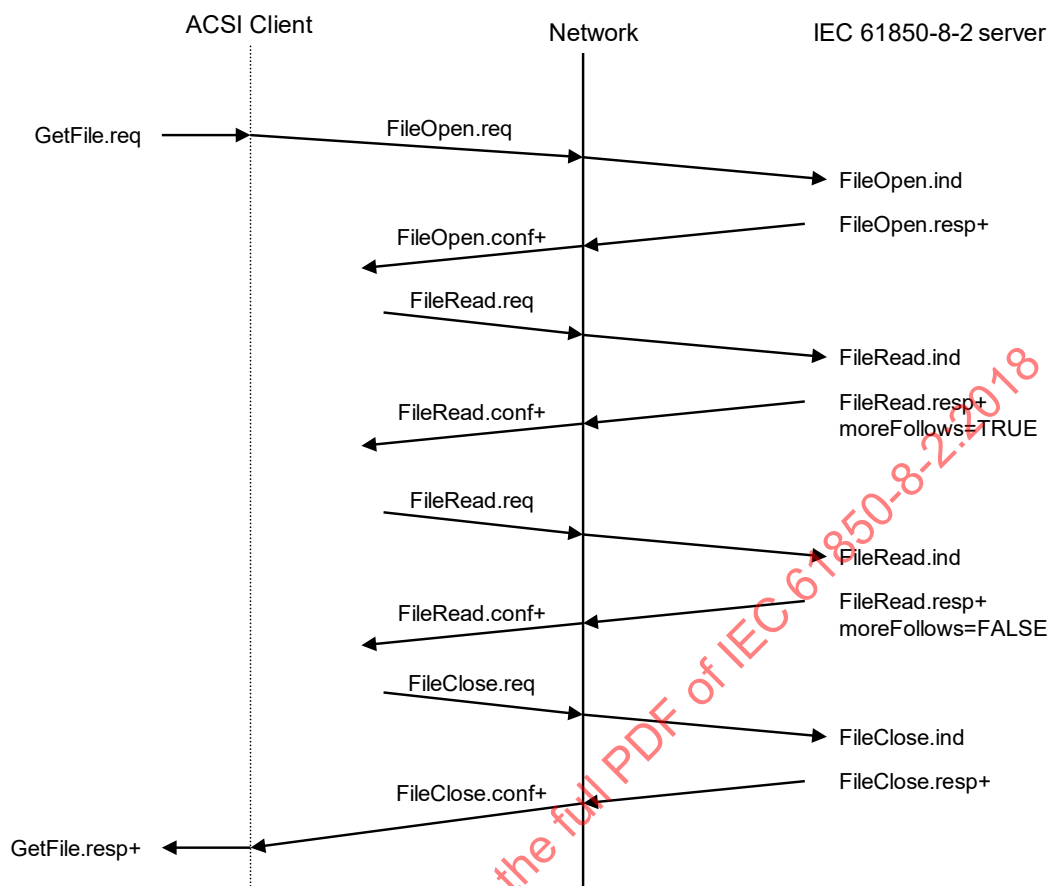


Figure 81 – Mapping de GetFile ACSI à FileOpen, FileRead, FileClose

La Figure 81 décrit la séquence de services unitaires qui résultent d'une demande GetFile ACSI. Une demande GetFile ACSI génère une demande FileOpen. Le paramètre initialPosition doit être 0. Une confirmation positive de FileOpen doit générer la première d'une séquence possible de demandes FileRead. Une confirmation positive de FileRead avec moreFollows=FALSE doit générer l'émission d'une demande FileClose. Une confirmation positive de FileRead, avec moreFollows=TRUE doit générer une autre demande dans la séquence de demandes FileRead. Lors de la réception d'une confirmation positive de FileClose, une indication de réponse positive de GetFile doit être produite.

Si des services unitaires retournent une confirmation négative, une indication de réponse négative de GetFile doit être produite comme indiqué au Tableau 84 et en 23.2.1.2.

Le Tableau 83 spécifie le mapping du service GetFile ACSI à une séquence de services de fichiers unitaires. La Figure 82 à la Figure 89 donnent des exemples de messages XML résultant de cette séquence.