

---

---

**Information technology —  
Telecommunications and information  
exchange between systems — Protocol  
for Computer Supported  
Telecommunications Applications (CSTA)  
Phase III**

*Technologies de l'information — Télécommunications et échange  
d'information entre systèmes — Protocole pour applications en  
télécommunications supportées par ordinateur (CSTA) en phase III*

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**PDF disclaimer**

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

© ISO/IEC 2000

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 ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office  
Case postale 56 • CH-1211 Geneva 20  
Tel. + 41 22 749 01 11  
Fax + 41 22 749 09 47  
E-mail [copyright@iso.ch](mailto:copyright@iso.ch)  
Web [www.iso.ch](http://www.iso.ch)

Printed in Switzerland

**Contents**

Foreword	x
Introduction	xi
<b>1</b> Scope	<b>1</b>
<b>2</b> Normative References	<b>1</b>
<b>3</b> Conformance	<b>2</b>
<b>3.1</b> Static Requirements	<b>2</b>
<b>3.2</b> Dynamic Requirements	<b>2</b>
<b>3.3</b> PICS Requirement	<b>2</b>
<b>4</b> Definitions and Abbreviations	<b>2</b>
<b>5</b> CSTA Service Definition Model	<b>2</b>
<b>5.1</b> CSTA Application Layer Structure	<b>2</b>
<b>5.2</b> Remote Operations	<b>2</b>
<b>5.3</b> The CSTA Service Response	<b>3</b>
<b>5.4</b> Cross Referencing of Event Reports	<b>3</b>
<b>5.5</b> Handling of Private Data	<b>3</b>
<b>6</b> Interconnection service boundary	<b>3</b>
<b>7</b> Security	<b>3</b>
<b>8</b> Association Management	<b>4</b>
<b>8.1</b> Implicit association	<b>4</b>
<b>8.2</b> Dynamic association management using ACSE	<b>4</b>
<b>8.2.1</b> Encoding of application context name	<b>4</b>
<b>8.2.2</b> Encoding of application context information	<b>4</b>
<b>9</b> CSTA parameter types	<b>8</b>
<b>9.1</b> Switching function objects	<b>8</b>
<b>9.2</b> Device Identifiers	<b>9</b>
<b>9.3</b> Call and connection identifiers	<b>11</b>
<b>9.4</b> Connection states	<b>12</b>
<b>9.5</b> Status reporting	<b>13</b>
<b>9.6</b> Device and feature types and other parameters	<b>15</b>
<b>9.7</b> Security	<b>20</b>
<b>9.8</b> Common extensions	<b>21</b>
<b>9.9</b> Call control	<b>22</b>
<b>9.10</b> Capability Exchange	<b>24</b>
<b>9.11</b> Call Detail Record	<b>73</b>
<b>9.12</b> Charge information	<b>75</b>
<b>9.13</b> Data call types	<b>76</b>
<b>9.14</b> Escape types	<b>77</b>
<b>9.15</b> Media services	<b>78</b>
<b>9.16</b> Physical device features	<b>80</b>
<b>9.17</b> Data Collection	<b>82</b>
<b>9.18</b> Event Cause	<b>83</b>
<b>9.19</b> Error Value	<b>85</b>

10	Event Report Service	90
11	Capability exchange services	95
11.1	Services	95
11.1.1	Get logical device information	95
11.1.2	Get physical device information	97
11.1.3	Get switching function capabilities	98
11.1.4	Get switching function devices	101
11.1.5	Switching function devices	102
12	System services	103
12.1	Registration services	103
12.1.1	Change system status filter	103
12.1.2	System register	104
12.1.3	System register abort	105
12.1.4	System register cancel	106
12.2	Services	107
12.2.1	Request system status	107
12.2.2	System status	108
12.2.3	Switching function capabilities changed	109
12.2.4	Switching function devices changed	110
13	Monitoring services	111
13.1	Services	111
13.1.1	Change monitor filter	111
13.1.2	Monitor start	112
13.1.3	Monitor stop	113
14	Snapshot services	114
14.1	Services	114
14.1.1	Snapshot call	114
14.1.2	Snapshot device	116
14.1.3	Snapshot calldata	117
14.1.4	Snapshot devicedata	118
15	Call control services and events	119
15.1	Services	119
15.1.1	Accept call	119
15.1.2	Alternate call	120
15.1.3	Answer call	121
15.1.4	Call back call-related	122
15.1.5	Call back message call-related	123
15.1.6	Camp on call	124
15.1.7	Clear call	125
15.1.8	Clear Connection	126
15.1.9	Conference call	127
15.1.10	Consultation call	128
15.1.11	Deflect call	129

15.1.12	Dial digits	130
15.1.13	Directed pickup call	131
15.1.14	Group pickup call	132
15.1.15	Hold call	133
15.1.16	Intrude call	134
15.1.17	Join call	135
15.1.18	Make call	136
15.1.19	Make predictive call	137
15.1.20	Park call	139
15.1.21	Reconnect call	140
15.1.22	Retrieve call	141
15.1.23	Single step conference call	142
15.1.24	Single step transfer call	143
15.1.25	Transfer call	144
15.2	Events	145
15.2.1	Bridged	145
15.2.2	Call cleared	146
15.2.3	Conferenced	147
15.2.4	Connection cleared	148
15.2.5	Delivered	149
15.2.6	Digits dialed	150
15.2.7	Diverted	151
15.2.8	Established	152
15.2.9	Failed	153
15.2.10	Held	154
15.2.11	Network capabilities changed	155
15.2.12	Network reached	156
15.2.13	Offered	157
15.2.14	Originated	158
15.2.15	Queued	159
15.2.16	Retrieved	160
15.2.17	Service initiated	161
15.2.18	Transferred	162
16	Call associated features	163
16.1	Services	163
16.1.1	Associate data	163
16.1.2	Cancel telephony tones	164
16.1.3	Generate digits	165
16.1.4	Generate telephony tones	166
16.1.5	Send user information	167
16.2	Events	168
16.2.1	Call information	168
16.2.2	Charging	169
16.2.3	Digits generated	170
16.2.4	Telephony tones generated	171
16.2.5	Service completion failure	172
17	Media attachment services and events	173
17.1	Services	173

17.1.1	Attach media service	173
17.1.2	Detach media service	174
17.2	Events	175
17.2.1	Media attached	175
17.2.2	Media detached	176
18	Routeing services	177
18.1	Registration services	177
18.1.1	Route register	177
18.1.2	Route register abort	178
18.1.3	Route register cancel	179
18.2	Services	180
18.2.1	Re-Route	180
18.2.2	Route end	181
18.2.3	Route reject	182
18.2.4	Route request	183
18.2.5	Route select	184
18.2.6	Route used	185
19	Physical device features	186
19.1	Services	186
19.1.1	Button press	186
19.1.2	Get auditory apparatus information	187
19.1.3	Get button information	188
19.1.4	Get display	189
19.1.5	Get hookswitch status	190
19.1.6	Get lamp information	191
19.1.7	Get lamp mode	192
19.1.8	Get message waiting indicator	193
19.1.9	Get microphone gain	194
19.1.10	Get microphone mute	195
19.1.11	Get ringer status	196
19.1.12	Get speaker mute	197
19.1.13	Get speaker volume	198
19.1.14	Set button information	199
19.1.15	Set display	200
19.1.16	Set hookswitch status	201
19.1.17	Set lamp mode	202
19.1.18	Set message waiting indicator	203
19.1.19	Set microphone gain	204
19.1.20	Set microphone mute	205
19.1.21	Set ringer status	206
19.1.22	Set speaker mute	207
19.1.23	Set speaker volume	208
19.2	Events	209
19.2.1	Button information	209
19.2.2	Button press	210
19.2.3	Display updated	211

19.2.4	Hookswitch	212
19.2.5	Lamp mode	213
19.2.6	Message waiting	214
19.2.7	Microphone gain	215
19.2.8	Microphone mute	216
19.2.9	Ringer status	217
19.2.10	Speaker mute	218
19.2.11	Speaker volume	219
20	Logical device features	220
20.1	Services	220
20.1.1	Call back non-call-related	220
20.1.2	Call back message non-call-related	221
20.1.3	Cancel call back	222
20.1.4	Cancel call back message	223
20.1.5	Get agent state	224
20.1.6	Get auto answer	225
20.1.7	Get auto work mode	226
20.1.8	Get caller id status	227
20.1.9	Get do not disturb	228
20.1.10	Get forwarding	229
20.1.11	Get last number dialed	230
20.1.12	Get routeing mode	231
20.1.13	Set agent state	232
20.1.14	Set auto answer	233
20.1.15	Set auto work mode	234
20.1.16	Set caller id status	235
20.1.17	Set do not disturb	236
20.1.18	Set forwarding	237
20.1.19	Set routeing mode	238
20.2	Events	239
20.2.1	Agent busy	239
20.2.2	Agent logged off	240
20.2.3	Agent logged on	241
20.2.4	Agent not ready	242
20.2.5	Agent ready	243
20.2.6	Agent working after call	244
20.2.7	Auto answer	245
20.2.8	Auto work mode	246
20.2.9	Call back	247
20.2.10	Call back message	248
20.2.11	Caller id status	249
20.2.12	Do not disturb	250
20.2.13	Forwarding	251
20.2.14	Routeing mode	252
21	Device maintenance events	253
21.1	Events	253
21.1.1	Back in service	253
21.1.2	Device capabilities changed	254
21.1.3	Out of service	255

22	I/O services	256
22.1	Registration services	256
22.1.1	I/O register	256
22.1.2	I/O register abort	257
22.1.3	I/O register cancel	258
22.2	Services	259
22.2.1	Data path resumed	259
22.2.2	Data path suspended	260
22.2.3	Fast data	261
22.2.4	Resume data path	262
22.2.5	Send broadcast data	263
22.2.6	Send data	264
22.2.7	Send multicast data	265
22.2.8	Start data path	266
22.2.9	Stop data path	267
22.2.10	Suspend data path	268
23	Data Collection Services	269
23.1	Services	269
23.1.1	Data Collected	269
23.1.2	Data Collection Resumed	271
23.1.3	Data Collection Suspended	272
23.1.4	Resume Data Collection	273
23.1.5	Start Data Collection	274
23.1.6	Stop Data Collection	275
23.1.7	Suspend Data Collection	276
24	Voice unit services and events	277
24.1	Services	277
24.1.1	Concatenate message	277
24.1.2	Delete message	278
24.1.3	Play message	279
24.1.4	Query voice attribute	280
24.1.5	Record message	281
24.1.6	Reposition	282
24.1.7	Resume	283
24.1.8	Review	284
24.1.9	Set voice attribute	285
24.1.10	Stop	286
24.1.11	Suspend	287
24.1.12	Synthesize message	288
24.2	Events	289
24.2.1	Play	289
24.2.2	Record	290
24.2.3	Review	291
24.2.4	Stop	292
24.2.5	Suspend play	293
24.2.6	Suspend record	294

24.2.7	Voice attribute changed	295
25	Call detail record services	296
25.1	Services	296
25.1.1	Call detail records notification	296
25.1.2	Call detail records report	297
25.1.3	Send stored call detail records	298
25.1.4	Start call detail records transmission	299
25.1.5	Stop call detail records transmission	300
26	Vendor specific extensions services and events	301
26.1	Registration services	301
26.1.1	Escape register	301
26.1.2	Escape register abort	302
26.1.3	Escape register cancel	303
26.2	Services	304
26.2.1	Escape	304
26.2.2	Private data version selection	305
26.3	Events	306
26.3.1	Private event	306
<b>Annex A - Protocol Implementation Conformance Statement (PICS) Proforma</b>		<b>307</b>

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 18052 was prepared by ECMA (as ECMA-285) and was adopted, under a special "fast-track procedure", by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

Annex A forms a normative part of this International Standard.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## Introduction

This International Standard defines Phase III of Protocol for Computer Supported Telecommunications Applications (CSTA) for OSI Layer 7 communication between a computing network and a telecommunications network. This International Standard is part of a Suite of Standards and Technical Reports for Phase III of CSTA. All of the Standards and Technical Reports in the Suite are based on practical experience of ECMA member companies and each one represents a pragmatic and widely-based consensus.

The evolution of this Suite began with CSTA Phase I, which included the CSTA Services and Protocol Standards (ECMA-179 and ECMA-180). In Phase II, Technical Report ECMA TR/68 was added illustrating how CSTA services and events may be used in typical call scenarios.

Phase III of CSTA extends the previous Phase II Standards (ECMA-217 and ECMA-218) in major theme directions as well as numerous details. This incorporates technology based upon the *versit* CTI Encyclopedia (Version 1.0), which was contributed to ECMA by *versit*. Major areas of advancement include:

- New categories of services and events such as capabilities exchange, charging, media attachment services, call data recording (CDR), etc.
- Additional services and events for call and device control.
- Enhancement to existing services and events.
- Organization of services and events to reflect a grouping based on function (call control, device control, etc.).

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

# Information technology - Telecommunications and information exchange between systems - Protocol for Computer Supported Telecommunications Applications (CSTA) Phase III

## 1 Scope

This International Standard specifies application protocol data units (APDUs) for the services described in ISO/IEC 18051, *Information technology - Telecommunications and information exchange between systems - Services for Computer Supported Telecommunications Applications (CSTA) Phase III*.

Clause 5 to clause 7 inclusive describes the concepts underlying the Remote Operations model, notation and service.

Clause 8 to clause 26 inclusive contains CSTA-specific protocol details and forms the main part of this International Standard.

The protocol in this International Standard operates in the context of an application association.

## 2 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this International Standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of IEC and ISO maintain registers of currently valid International Standards.

- ISO/IEC 8649:1996**, *Information technology - Open Systems Interconnection - Service definition for the Association Control Service Element* (this corresponds to ITU-T Rec. X.217 1995).
- ISO/IEC 8650-1:1996**, *Information technology - Open Systems Interconnection - Connection-oriented protocol for the Association Control Service Element: Protocol specification* (this corresponds to ITU-T Rec. X.227 4/95).
- ISO/IEC 9545:1994**, *Information technology - Open Systems Interconnection - Application Layer structure*.
- ISO/IEC 13712-1:1995**, *Information technology - Remote Operations: Concepts, model, and notation* (this corresponds to ITU-T Rec. X.880, 1994).
- ISO/IEC 13712-2:1994**, *Information technology - Remote Operations: OSI realisations - Remote Operations Service Element (ROSE) service definition* (this corresponds to ITU-T Rec. X.881, 1994).
- ISO/IEC 13712-3:1994**, *Information technology - Remote Operations: OSI realisations - Remote Operations Service Element (ROSE) protocol specification* (this corresponds to ITU-T Rec. X.882, 1994).
- ISO/IEC 18051:2000**, *Information technology - Telecommunications and information exchange between systems - Services for Computer Supported Telecommunications Applications (CSTA) Phase III*.
- ISO/IEC TR 18053:2000**, *Information technology - Telecommunications and information exchange between systems - Glossary of definitions and terminology for Computer Supported Telecommunications Applications (CSTA) Phase III*.

- ITU-T X.680,** *Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation.*
- ITU-T X.690,** *Information technology - ASN.1 encoding rules - Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).*

### 3 Conformance

A manufacturer may select any part of the CSTA protocol, as specified in this International Standard, for implementation on a system as long as it satisfies the minimum conformance requirements as specified in clause 3 of ISO/IEC 18051.

A Protocol Implementation Conformance Statement (PICS) shall be used to specify the operations which are provided by a particular implementation. A PICS shall also specify the parameter options which are used.

#### 3.1 Static Requirements

To conform to this International Standard, a system shall support the transfer syntax (derived from the encoding rules specified in ITU-T X.690) named {joint-iso-ccitt asn1(1) basic-encoding(1)}; for the purpose of generating and interpreting CSTA protocol information as defined by the abstract syntax "CSTA-ASN.1-Object-Descriptor" for the operations supported.

#### 3.2 Dynamic Requirements

To conform to this International Standard, a system shall:

- a. follow the procedures as specified in this International Standard, and ISO/IEC 18051, relevant to each CSTA operation that the system claims to implement; and
- b. satisfy the definitions, as specified in ISO/IEC 18051, relevant to each CSTA service that the system claims to implement.

#### 3.3 PICS Requirement

To conform to this International Standard, the following shall be stated by the implementer when defining a PICS corresponding to an application or implementation:

- a. which CSTA operations, as defined in ISO/IEC 18051, are supported by the system for the particular implementation; and
- b. which optional parameters are supported by the PDUs belonging to the supported operations.

A PICS proforma is given in Annex A of this International Standard.

### 4 Definitions and Abbreviations

CSTA-specific terminology is defined in ISO/IEC TR 18053. For the purposes of this International Standard, the following additional definitions, defined in other standards, shall apply:

- Remote Operations (as per ISO/IEC 13712-1)
- Application Association (as per ISO/IEC 8649)
- Application Context (as per ISO/IEC 8649)

### 5 CSTA Service Definition Model

#### 5.1 CSTA Application Layer Structure

The CSTA Application Layer structure conforms to the model described in ISO/IEC 9545.

#### 5.2 Remote Operations

The services of CSTA are modeled as Remote Operations as described in ISO/IEC 13712-1. Typically, one entity requests that a particular operation be performed; the other entity attempts to perform the operation and responds to the requestor. Consequently the operation of the protocol is an elementary request/reply interaction, supported within the OSI Application Layer, and carried out within the context of an application association.

For some of the CSTA services, the entity to which the request is sent generates a reply which can indicate success or failure. For these services, CSTA shall adopt the operations Class 2, defined in ISO/IEC 13712-2 as:

- Asynchronous, reporting success or failure (result or error).

For some of the CSTA services, the entity to which the request is sent generates a reply which can only indicate failure. For these services, CSTA shall adopt the operations Class 3, defined in ISO/IEC 13712-2 as:

- Asynchronous, reporting failure (error) only, if any.

For some of the CSTA services, particularly the ongoing reporting of events, no reply is generated. For these services, CSTA shall adopt the operations Class 5, defined in ISO/IEC 13712-2 as:

- Asynchronous, outcome not reported.

The protocol description for the particular service defines the relevant class of the operation used for that service.

CSTA shall correlate the single response, denoting success or failure, with the originating request by using the mechanisms within the ROSE protocol.

### 5.3 The CSTA Service Response

CSTA employs a generic response mechanism which is, in principle, decoupled from the specifics of the switching activity. The following points describe the operation of the CSTA service response:

- Specific services may have an unconfirmed mode where responses to correct requests are not returned.
- The server shall check the correctness of the request (e.g. syntactical checks) before issuing the response. Incorrect requests shall result in an error response, even in the unconfirmed mode.

### 5.4 Cross Referencing of Event Reports

A computer application process may need to cross reference a CSTAEventReport to one of the following:

- a CSTA Object ID (Call ID or Device ID),
- an earlier Monitor request; or
- one of many Monitor requests (pertaining to the same CSTA Object).

For the above scenarios the necessary cross referencing function may be fulfilled by use of the parameter "MonitorCrossRefID". The content of MonitorCrossRefID depends upon the context and it may be one of the following: Call ID, Device ID or another independently switch managed static identifier. The independent identifier may have a unique correlation to either: one device, one call, or one monitor request.

The switching system limit on the number of monitor requests on one CSTA Object (Call or Device) is an implementation option. This International Standard does not stipulate how many monitor requests on one object are to be supported by the switch. If using Static Device or Call identifiers the limit can only be one.

### 5.5 Handling of Private Data

If an entity receives the parameter CSTAPrivateData, and it can not recognize the information contained, the parameter shall be discarded, and the rest of the message shall be processed.

## 6 Interconnection service boundary

The protocol in this International Standard is an OSI Application Layer protocol and uses the Remote Operations protocol defined in ISO/IEC 13712-3. The Remote Operations protocol assumes certain services are provided by the underlying layers, and these services are also assumed by the protocol for CSTA.

## 7 Security

This protocol also provides a mechanism for secure transmission of CSTA PDUs as defined in this International Standard.

## 8 Association Management

The protocol in this International Standard operates in the context of an application association. This application association can be either:

- an implicit association achieved via off-line agreement; or
- a dynamically negotiated association realized through the use of ACSE.

### 8.1 Implicit association

An a-priori agreement exists between switching and computing functions: the application context is implicit, dynamic negotiation is not possible.

### 8.2 Dynamic association management using ACSE

ACSE requires the use of an application context Name (as defined by ISO 8649). This is an object id that uniquely identifies CSTA independently of the different versions of the protocol.

The CSTA protocol version information is carried within the User Information field of the A-ASSOCIATE request and response PDUs.

An application context is established (using ACSE) as follows:

- the system generating the A-ASSOCIATE request includes the CSTA application context name and a list of all protocol versions that it is prepared to offer in the User Information field;
- on receipt of the A-ASSOCIATE request, the receiving system selects the protocol version to be used by identifying the highest version that is common to both systems;
- the protocol version selected is conveyed to the requestor in the User Information field of the A-ASSOCIATE response.

In addition to negotiating the protocol version, it is necessary for the requesting and responding systems to specify the CSTA services that they support. As with the protocol version information, this is also achieved by carrying additional information in the User Information field of the A-ASSOCIATE request and response PDUs. The application association requestor shall:

- list the services required from the serving application;
- list the services it can supply.

The responder shall include similar information for the responding application. At this point the association requestor will either accept or reject the association.

#### 8.2.1 Encoding of application context name

```
CSTA-application-context-name
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta( 218) }
-- Common to all CSTA protocol versions
```

#### 8.2.2 Encoding of application context information

```
CSTA-application-context-information-csta3
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) application-context-information( 200) }
```

```
DEFINITIONS ::=
BEGIN
```

```
IMPORTS
```

```
CSTAF functionality FROM CSTA-application-context-information
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta2( 218) application-context-information( 200) };
```

```
ACSEUserInformationForCSTA ::= CHOICE
{ oldDefinition          OldACSEUserInformationForCSTA,
  newDefinition [0] IMPLICIT NewACSEUserInformationForCSTA }
```

```

OldACSEUserInformationForCSTA ::= SEQUENCE
    {cSTAVersion
      CSTAVersion,
      CSTAFunctionsRequiredByApplication
      CSTAFunctionality,
      CSTAFunctionsThatCanBeSupplied
      CSTAFunctionality
    }

NewACSEUserInformationForCSTA ::= SEQUENCE
{
    CSTAVersion
    CSTAVersion,
--
--Note that the following two parameters are defined in ECMA-218 (CSTA Phase II).
--They are defined in the context of CSTA Phase II Services and Events for
--backward compatibility considerations.
--For CSTA Phase III (and later), implementations should use CSTA Capability Exchange Services
--to determine the Services and Events supported by a switching function.
--
    CSTAFunctionsRequiredByApplication [0] IMPLICIT CSTAFunctionality OPTIONAL,
    CSTAFunctionsThatCanBeSupplied [1] IMPLICIT CSTAFunctionality OPTIONAL,
    CSTAPrivateDataVersionList [2] IMPLICIT CSTAPrivateDataVersionList OPTIONAL}

CSTAPrivateDataVersionList ::= SEQUENCE OF INTEGER

CSTAVersion ::= BIT STRING
{
    versionOne (0), -- CSTA protocol version defined in ECMA-180
    versionTwo (1), -- CSTA protocol version defined in ECMA-218
    versionThree (2), -- CSTA protocol version defined in ECMA-285
    versionFour (3), -- CSTA protocol version defined in ISO/IEC 18052
    versionFive (4), -- Reserved for future use
    versionSix (5), -- Reserved for future use
    versionSeven (6), -- Reserved for future use
    versionEight (7), -- Reserved for future use
    versionNine (8), -- Reserved for future use
    versionTen (9), -- Reserved for future use
    versionEleven (10), -- Reserved for future use
    versionTwelve (11), -- Reserved for future use
    versionThirteen (12), -- Reserved for future use
    versionFourteen (13), -- Reserved for future use
    versionFifteen (14), -- Reserved for future use
    versionSixteen (15)} -- Reserved for future use

CallControlServices ::= BIT STRING
{
    acceptCall (0),
    alternateCall (1),
    answerCall (2),
    callBack (3),
    callBackMessage (4),
    campOnCall (5),
    clearCall (6),
    clearConnection (7),
    conferenceCall (8),
    consultationCall (9),
    deflectCall (10),
    dialDigits (11),
    directedPickupCall (12),
    groupPickupCall (13),
    holdCall (14),
    intrudeCall (15),
    joinCall (16),
    makeCall (17),
    makePredictiveCall (18),
    parkCall (19),
    reconnectCall (20),
    retrieveCall (21),
    singleStepConference (22),
    singleStepTransfer (23),
    transferCall (24)}

CallAssociatedServices ::= BIT STRING
{
    associateData (0),

```

```

cancelTelephonyTones (1),
generateDigits (2),
generateTelephonyTones (3),
sendUserInformation (4)}

MediaAttachmentServices ::= BIT STRING
{
    attachMediaService (0),
    detachMediaService (1)}

RouteingServices ::= BIT STRING
{
    routeRegister (0),
    routeRegisterCancel (1),
    routeRegisterAbort (2),
    reRoute (3),
    routeEnd (4),
    routeReject (5),
    routeRequest (6),
    routeSelect (7),
    routeUsed (8)}

VoiceUnitServices ::= BIT STRING
{
    concatenateMessage (0),
    deleteMessage (1),
    playMessage (2),
    queryVoiceAttribute (3),
    recordMessage (4),
    reposition (5),
    resume (6),
    review (7),
    setVoiceAttribute (8),
    stop (9),
    suspend (10),
    synthesiseMessage (11)}

CallControlEvents ::= BIT STRING
{bridged (15),
callCleared (0),
conferenced (1),
connectionCleared (2),
delivered (3),
digitsDialed (14),
diverted (4),
established (5),
failed (6),
held (7),
networkCapabilitiesChanged (16),
networkReached (8),
offered (17),
originated (9),
queued (10),
retrieved (11),
serviceInitiated (12),
transferred (13) }

CallAssociatedEvents ::= BIT STRING
{
    callInformation (0),
    charging (1),
    dTMFDigitsDetected (2),
    telephonyTonesDetected (3),
    serviceCompletionFailure (4)}

MediaAttachmentEvents ::= BIT STRING
{
    mediaAttached (0),
    mediaDetached (1)}

PhysicalDeviceFeatureEvents ::= BIT STRING
{buttonInformation (0),
buttonPress (1),

```

STANDARD.PDFISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

```

        displayUpdated          (2),
        hookswitch              (3),
        lampMode                (4),
        messageWaiting          (5),
        microphoneGain          (6),
        microphoneMute          (7),
        ringerStatus            (8),
        speakerMute              (9),
        speakerVolume           (10)}

LogicalDeviceFeatureEvents ::= BIT STRING
{
    agentBusy                    (0),
    agentLoggedOn                (1),
    agentLoggedOff               (2),
    agentNotReady                (3),
    agentReady                   (4),
    agentWorkingAfterCall        (5),
    autoAnswer                   (6),
    autoWorkMode                 (7),
    callBack                     (8),
    callBackMessage              (9),
    callerIDStatus               (10),
    doNotDisturb                 (11),
    forwarding                    (12),
    routeingMode                 (13)}

DeviceMaintenanceEvents ::= BIT STRING
{
    backInService                (0),
    deviceCapabilityChanged       (2),
    outOfService                  (1) }

VoiceUnitEvents ::= BIT STRING
{
    play                         (1),
    record                       (3),
    review                       (5),
    stop                         (0),
    suspendPlay                  (2),
    suspendRecord                 (4),
    voiceAttributesChange        (6) }

VendorSpecEvents ::= BIT STRING
{
    privateEvent                 (0)}

END -- of CSTA-application-context-information

```

## 9 CSTA parameter types

The major parameters have been assigned distinct application tags to facilitate parsing. The data is defined in logical groups in ascending order of application tag. Application tags used are:

- APPLICATION 1 - 8: . . . . . Device identifiers
- APPLICATION 11 - 14: . . . . Connection identifiers and local connection states
- APPLICATION 21 - 24: . . . Status reporting
- APPLICATION 29 - 30: . . . CSTACommonArguments

### 9.1 Switching function objects

```

CSTA-switching-function-objects
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) switching-function-objects( 122) }
DEFINITIONS ::=
BEGIN

EXPORTS
CSTAObject;

IMPORTS
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) };

CSTAObject ::= CHOICE
    {deviceObject    DeviceID,
     callObject     ConnectionID}

END -- of CSTA-switching-function-objects

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.2 Device Identifiers

```

CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }

DEFINITIONS ::=
BEGIN

EXPORTS
DeviceID, NumberDigits, CallingDeviceID, CalledDeviceID,
SubjectDeviceID, RedirectionDeviceID, AssociatedCallingDeviceID,
AssociatedCalledDeviceID, NetworkCallingDeviceID, NetworkCalledDeviceID;

IMPORTS
MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

CallingDeviceID ::= [APPLICATION 1] CHOICE
    { deviceIdentifier      DeviceID,
      notKnown              [7] IMPLICIT          NULL }

CalledDeviceID ::= [APPLICATION 2] CHOICE
    { deviceIdentifier      DeviceID,
      notKnown              [7] IMPLICIT          NULL }

SubjectDeviceID ::= [APPLICATION 3] CHOICE
    { deviceIdentifier      DeviceID,
      notKnown              [7] IMPLICIT          NULL }

RedirectionDeviceID ::= [APPLICATION 4] CHOICE
    { numberDialled        DeviceID,
      notKnown              [7] IMPLICIT          NULL,
      notRequired          [8] IMPLICIT          NULL,
      notSpecified         [9] IMPLICIT          NULL }

AssociatedCallingDeviceID ::= [APPLICATION 5] CHOICE
    { deviceIdentifier      DeviceID,
      notKnown              [7] IMPLICIT          NULL }

AssociatedCalledDeviceID ::= [APPLICATION 6] CHOICE
    { deviceIdentifier      DeviceID,
      notKnown              [7] IMPLICIT          NULL }

NetworkCallingDeviceID ::= [APPLICATION 7] CHOICE
    { deviceIdentifier      DeviceID,
      notKnown              [7] IMPLICIT          NULL }

NetworkCalledDeviceID ::= [APPLICATION 8] CHOICE
    { deviceIdentifier      DeviceID,
      notKnown              [7] IMPLICIT          NULL }

DeviceID ::= SEQUENCE
    {
        deviceIdentifier      CHOICE
        {
            dialingNumber      [0] IMPLICIT          NumberDigits,
            deviceNumber        [1] IMPLICIT          DeviceNumber,
            implicitPublic      [2] IMPLICIT          NumberDigits,
            explicitPublic      [3] IMPLICIT          PublicTON,
            implicitPrivate     [4] IMPLICIT          NumberDigits,
            explicitPrivate     [5] IMPLICIT          PrivateTON,
            other                [6] IMPLICIT          OtherPlan},
        mediaCallCharacteristics MediaCallCharacteristics OPTIONAL }

PublicTON ::= CHOICE
    {
        unknown                [0] IMPLICIT          IA5String,
        international          [1] IMPLICIT          IA5String,
        national                [2] IMPLICIT          IA5String,

```

```

networkspecific      [3] IMPLICIT      IA5String,
subscriber           [4] IMPLICIT      IA5String,
abbreviated          [5] IMPLICIT      IA5String  }

PrivateTON ::= CHOICE
{
    unknown           [0] IMPLICIT      IA5String,
    level3RegionalNumber [1] IMPLICIT      IA5String,
    level2RegionalNumber [2] IMPLICIT      IA5String,
    level1RegionalNumber [3] IMPLICIT      IA5String,
    pTNSpecificNumber  [4] IMPLICIT      IA5String,
    localNumber        [5] IMPLICIT      IA5String,
    abbreviated        [6] IMPLICIT      IA5String  }

OtherPlan ::= OCTET STRING      -- Allows future expansion to cover other numbering
                                -- plans

NumberDigits ::= IA5String

DeviceNumber ::= INTEGER

END -- of CSTA-device-identifiers

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

### 9.3 Call and connection identifiers

```

CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
DEFINITIONS ::=
BEGIN

EXPORTS
CallID, ConnectionID;

IMPORTS
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) };

ConnectionID ::= [APPLICATION 11] CHOICE
{
    callID      [0] IMPLICIT      CallID,
    deviceID    [1]                LocalDeviceID,
    both        SEQUENCE
    {
        callID      [0] IMPLICIT      CallID,
        deviceID    [1]                LocalDeviceID
    }
}

CallID ::= OCTET STRING (SIZE(0..8))

LocalDeviceID ::= CHOICE
{
    staticID      DeviceID,
    dynamicID     [3] IMPLICIT OCTET STRING (SIZE(0..32)) }

END -- of CSTA-call-connection-identifiers

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.4 Connection states

```

CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }

DEFINITIONS ::=
BEGIN

EXPORTS
ConnectionList, LocalConnectionState;

IMPORTS
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) };
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) };
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

ConnectionList ::= SEQUENCE OF SEQUENCE
    {
        newConnection [0] ConnectionID OPTIONAL,
        oldConnection [1] ConnectionID OPTIONAL,
        endpoint [2] CHOICE
            {
                deviceID DeviceID,
                notKnown NULL} OPTIONAL,
        associatedNID [3] CHOICE
            {
                deviceID DeviceID,
                notKnown NULL} OPTIONAL,
        resultingConnectionInfo ConnectionInformation OPTIONAL}

LocalConnectionState ::= [APPLICATION 14] IMPLICIT ENUMERATED
    {
        null (0),
        initiated (1),
        alerting (2),
        connected (3),
        hold (4),
        queued (5),
        fail (6)}

END -- CSTA-connection-states

```

## 9.5 Status reporting

```

CSTA-status-reporting
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) status-reporting( 126) }
DEFINITIONS ::=
BEGIN

EXPORTS
MonitorObject, MonitorCrossRefID, MonitorFilter, MonitorType,
MonitorMediaClass, SnapshotCallData, SnapshotDeviceData;

IMPORTS
CallControlEvents, CallAssociatedEvents, MediaAttachmentEvents,
PhysicalDeviceFeatureEvents, LogicalDeviceFeatureEvents,
DeviceMaintenanceEvents, VoiceUnitEvents, VendorSpecEvents
FROM CSTA-application-context-information-csta3
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) application-context-information( 200) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control(130) }
CSTAObject FROM CSTA-switching-function-objects
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) switching-function-objects( 122) }
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
ConnectionInformation, MediaCallCharacteristics, MediaServiceType,
MediaServiceInstanceID, MediaStreamID FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) }
;

MonitorObject ::= CSTAObject;

MonitorCrossRefID ::= [APPLICATION 21] IMPLICIT OCTET STRING (SIZE(0..4))

MonitorFilter ::= SEQUENCE-- default is no filter (i.e. all events)
    { callControl          [0] IMPLICIT CallControlEvents          DEFAULT {},
      callAssociated       [6] IMPLICIT CallAssociatedEvents        DEFAULT {},
      mediaAttachment      [7] IMPLICIT MediaAttachmentEvents       DEFAULT {},
      physicalDeviceFeature [8] IMPLICIT PhysicalDeviceFeatureEvents DEFAULT {},
      logicalDeviceFeature [9] IMPLICIT LogicalDeviceFeatureEvents  DEFAULT {},
      maintenance         [3] IMPLICIT DeviceMaintenanceEvents     DEFAULT {},
      voiceUnit            [5] IMPLICIT VoiceUnitEvents             DEFAULT {},
      private              [4] IMPLICIT VendorSpecEvents            DEFAULT {} }

-- setting the relevant bit requests the filter for the appropriate events

MonitorType ::= ENUMERATED
    {
        call          (0),
        device        (1)
    }

MonitorMediaClass ::= BIT STRING
    {
        voice          (0),
        data           (1),
        image          (2),
        audio          (4),
        other          (3)
    }

```

```

SnapshotDeviceData ::= [APPLICATION 22] IMPLICIT SEQUENCE OF SnapshotDeviceResponseInfo

SnapshotDeviceResponseInfo ::= SEQUENCE
{
    connectionIdentifier      ConnectionID,
    localCallState            CallState,
    servicesPermitted         [0] IMPLICIT ServicesPermitted      OPTIONAL,
    mediaServiceInfoList     [1] IMPLICIT DeviceMediaInfoList     OPTIONAL,
    mediaCallCharacteristics  [2] IMPLICIT MediaCallCharacteristics OPTIONAL}

DeviceMediaInfoList ::= SEQUENCE OF SEQUENCE
{
    mediaStreamID             MediaStreamID      OPTIONAL,
    connectionInformation    ConnectionInformation OPTIONAL}

SnapshotCallData ::= [APPLICATION 23] IMPLICIT SEQUENCE OF SnapshotCallResponseInfo

SnapshotCallResponseInfo ::= SEQUENCE
{
    deviceOnCall              SubjectDeviceID,
    callIdentifier            ConnectionID      OPTIONAL,
    localConnectionState     LocalConnectionState OPTIONAL,
    servicesPermitted         [0] IMPLICIT ServicesPermitted      OPTIONAL,
    mediaServiceInfoList     [1] IMPLICIT CallMediaInfoList      OPTIONAL
}

CallMediaInfoList ::= SEQUENCE OF SEQUENCE
{
    mediaServiceType         [0] IMPLICIT MediaServiceType      OPTIONAL,
    mediaServiceVersion      [1] IMPLICIT INTEGER                OPTIONAL,
    mediaServiceInstance     [2] IMPLICIT MediaServiceInstanceID OPTIONAL,
    mediaStreamID            [3] IMPLICIT MediaStreamID          OPTIONAL,
    connectionInformation    [4] IMPLICIT ConnectionInformation  OPTIONAL}

CallState ::= CHOICE
{
    compoundCallState        [0] IMPLICIT CompoundCallState,
    simpleCallState          [1] IMPLICIT SimpleCallState,
    unknown                  [2] IMPLICIT NULL}

CompoundCallState ::= SEQUENCE OF LocalConnectionState

SimpleCallState ::= ENUMERATED
{
    callNull                 (0),      -- '00'H - null-null
    callPending              (1),      -- '01'H - null-initiate
    callOriginated           (3),      -- '03'H - null-connect
    callDelivered            (35),     -- '23'H - alerting-connect
    callDeliveredHeld        (36),     -- '24'H - alerting-held
    callReceived             (50),     -- '32'H - connect-alerting
    callEstablished          (51),     -- '33'H - connect-connect
    callEstablishedHeld      (52),     -- '34'H - connected-held
    callReceivedOnHold      (66),     -- '42'H - held-alerting
    callEstablishedOnHold   (67),     -- '43'H - held-connect
    callQueued               (83),     -- '53'H - queued-connect
    callQueuedHeld           (84),     -- '54'H - queued-held
    callFailed               (99),     -- '63'H - failed-connect
    callFailedHeld           (100),    -- '64'H - failed-held
    callBlocked              (96)}     -- '60'H - failed-null

-- This represents the main call states in a simplified encoding. The semantics
-- are identical to the sequence of connection states but they are represented by
-- an item from an enumerated list.

END -- of CSTA-status-reporting

```



## 9.6 Device and feature types and other parameters

```

CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
DEFINITIONS ::=
BEGIN

EXPORTS
AccountInfo, AgentID, AgentPassword, AgentState, AuthCode, CallOrigination, CorrelatorData,
DataPathType, DisplayAttributeList, ForwardList, IOCrossRefID, IORegisterReqID,
MessageID, ParticipationType, PendingAgentState, RetryValue, RouteRegisterReqID,
RouteingCrossRefID, SelectValue, SysStatRegisterID, SystemStatus, TerminatingConditions,
ForwardingType, ForwardDefault, AttributeInfo,
EncodingAlgorithm, ControlData, UserData, TelephonyTone;

IMPORTS
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) };

AccountInfo ::= OCTET STRING (SIZE(0..32))

AgentID ::= OCTET STRING (SIZE(0..32))

AgentPassword ::= OCTET STRING (SIZE(0..32))

AgentState ::= ENUMERATED
{
    agentNotReady      (0),
    agentNull          (1),
    agentReady         (2),
    agentBusy          (3),
    agentWorkingAfterCall (4)
}

AuthCode ::= OCTET STRING (SIZE(0..32))

CorrelatorData ::= OCTET STRING (SIZE(0..32))

IOCrossRefID ::= CHOICE
{
    switchProvided      [0] OCTET STRING (SIZE(0..4)),
    computerProvided    [1] OCTET STRING (SIZE(0..4))
}

IORegisterReqID ::= OCTET STRING (SIZE(0..4))

MessageID ::= OCTET STRING

ParticipationType ::= ENUMERATED
{
    silent      (0),
    active     (1)
}

RouteRegisterReqID ::= OCTET STRING (SIZE(0..4))

RouteingCrossRefID ::= [APPLICATION 24] IMPLICIT OCTET STRING (SIZE(0..4))

SelectValue ::= ENUMERATED
{
    normal      (0),
    leastCost   (1),
    emergency   (2),
    aCD         (3),
    userDefined (4)
}

RetryValue ::= CHOICE-- used in RouteSelect Request service
{
    noListAvailable      [0] IMPLICIT BOOLEAN,
    noCountAvailable     [1] IMPLICIT BOOLEAN,
    retryCount           [2] IMPLICIT INTEGER
}

```

```

SysStatRegisterID ::= OCTET STRING (SIZE(0..4))

SystemStatus ::= ENUMERATED
{
    disabled                (4),
    partiallyDisabled       (8),
    enabled                 (1),
    initializing            (0),
    messagesLost            (3),
    normal                  (2),
    overloadImminent        (5),
    overloadReached         (6),
    overloadRelieved        (7)
}

CallOrigination ::= BIT STRING
{
    internal    (0),
    external    (1)}

ForwardList ::= SEQUENCE OF SEQUENCE
{
    forwardingType    ForwardingType    OPTIONAL,
    forwardStatus     BOOLEAN,
    forwardDN         DeviceID          OPTIONAL,
    forwardDefault    ForwardDefault    OPTIONAL,
    ringCount         INTEGER (1..100)   OPTIONAL}

ForwardingType ::= ENUMERATED
{
    forwardImmediate    (0),
    forwardBusy         (1),
    forwardNoAns        (2),
    forwardDND          (9),
    forwardBusyInt      (3),
    forwardBusyExt      (4),
    forwardNoAnsInt     (5),
    forwardNoAnsExt     (6),
    forwardImmInt       (7),
    forwardImmExt       (8),
    forwardDNDInt       (10),
    forwardDNDExt       (11)}

ForwardDefault ::= ENUMERATED
{
    forwardingTypeAndForwardDN    (0),
    forwardingType                 (1),
    forwardDN                      (2)}

PendingAgentState ::= ENUMERATED
{
    agentNotReady    (0),
    agentNull        (1),
    agentReady       (2),
    agentWorkingAfterCall (3)}

DataPathType ::= ENUMERATED
{
    text    (0),
    voice   (1)}

DisplayAttributeList ::= SEQUENCE
{
    physicalBaseRowNumber    [0] IMPLICIT INTEGER    OPTIONAL,
    physicalBaseColumnNumber [1] IMPLICIT INTEGER    OPTIONAL,
    offset                   [2] IMPLICIT INTEGER    OPTIONAL}

TerminatingConditions ::= BIT STRING
{
    durationExceeded    (0),
    dTMFDigitDetected  (1),
    endOfMessageDetected (2),
    speechDetected      (3)}

AttributeInfo ::= CHOICE
{
    encodingAlgorithm    [0] IMPLICIT EncodingAlgorithm,
    samplingRate         [1] IMPLICIT INTEGER,
}

```

STANDARDSISO.COM · Click to view the full PDF of ISO/IEC 18052:2000

```

duration                [2] IMPLICIT INTEGER,
filename                [3] IMPLICIT IA5String,
currentPosition         [4] IMPLICIT INTEGER,
currentSpeed            [5] IMPLICIT INTEGER,
currentVolume           [6] IMPLICIT INTEGER (0 .. 100),
currentGain             [7] IMPLICIT INTEGER (0 .. 100),
currentState            [8] IMPLICIT CurrentState}

```

EncodingAlgorithm ::= ENUMERATED

```

{
  aDPCM6K      (0),
  aDPCM8K      (1),
  muLawPCM6K   (2),
  aLawPCM6K    (3)}

```

CurrentState ::= ENUMERATED

```

{
  stop          (0),
  play          (1),
  record        (2),
  suspendPlay   (3),
  suspendRecord (4),
  review        (5)}

```

ControlData ::= SEQUENCE

```

{
  gender          ENUMERATED
                  {male      (0),
                   female    (1)},
  language        OCTET STRING}

```

UserData ::= [APPLICATION 29] OCTET STRING (SIZE(0..256))

TelephonyTone ::= ENUMERATED

```

{
  beep           ( 0),
  billing        ( 1),
  busy           ( 2),
  carrier        ( 3),
  confirmation   ( 4),
  dial           ( 5),
  faxCNG         ( 6),
  hold           ( 7),
  howler         ( 8),
  intrusion      ( 9),
  modemCNG       (10),
  park           (11),
  recordWarning  (12),
  reorder        (13),
  ringback       (14),
  silence        (15),
  sitVC          (16),
  sitIC          (17),
  sitRO          (18),
  sitNC          (19),
  switchSpec0    (20),
  switchSpec1    (21),
  switchSpec2    (22),
  switchSpec3    (23),
  switchSpec4    (24),
  switchSpec5    (25),
  switchSpec6    (26),
  switchSpec7    (27),
  switchSpec8    (28),
  switchSpec9    (29),
  switchSpec10   (30),
  switchSpec11   (31),
  switchSpec12   (32),
  switchSpec13   (33),
  switchSpec14   (34),
  switchSpec15   (35),

```

switchSpec16	(36),
switchSpec17	(37),
switchSpec18	(38),
switchSpec19	(39),
switchSpec20	(40),
switchSpec21	(41),
switchSpec22	(42),
switchSpec23	(43),
switchSpec24	(44),
switchSpec25	(45),
switchSpec26	(46),
switchSpec27	(47),
switchSpec28	(48),
switchSpec29	(49),
switchSpec30	(50),
switchSpec31	(51),
switchSpec32	(52),
switchSpec33	(53),
switchSpec34	(54),
switchSpec35	(55),
switchSpec36	(56),
switchSpec37	(57),
switchSpec38	(58),
switchSpec39	(59),
switchSpec40	(60),
switchSpec41	(61),
switchSpec42	(62),
switchSpec43	(63),
switchSpec44	(64),
switchSpec45	(65),
switchSpec46	(66),
switchSpec47	(67),
switchSpec48	(68),
switchSpec49	(69),
switchSpec50	(70),
switchSpec51	(71),
switchSpec52	(72),
switchSpec53	(73),
switchSpec54	(74),
switchSpec55	(75),
switchSpec56	(76),
switchSpec57	(77),
switchSpec58	(78),
switchSpec59	(79),
switchSpec60	(80),
switchSpec61	(81),
switchSpec62	(82),
switchSpec63	(83),
switchSpec64	(84),
switchSpec65	(85),
switchSpec66	(86),
switchSpec67	(87),
switchSpec68	(88),
switchSpec69	(89),
switchSpec70	(90),
switchSpec71	(91),
switchSpec72	(92),
switchSpec73	(93),
switchSpec74	(94),
switchSpec75	(95),
switchSpec76	(96),
switchSpec77	(97),
switchSpec78	(98),
switchSpec79	(99),
switchSpec80	(100),
switchSpec81	(101),
switchSpec82	(102),
switchSpec83	(103),

STANDARDS150.COM . Click to view the full PDF of ISO/IEC 18052:2000

```
switchSpec84      (104),  
switchSpec85      (105),  
switchSpec86      (106),  
switchSpec87      (107),  
switchSpec88      (108),  
switchSpec89      (109),  
switchSpec90      (110),  
switchSpec91      (111),  
switchSpec92      (112),  
switchSpec93      (113),  
switchSpec94      (114),  
switchSpec95      (115),  
switchSpec96      (116),  
switchSpec97      (117),  
switchSpec98      (118),  
switchSpec99      (119),  
switchSpec100     (120)}
```

```
END -- of CSTA-device-feature-types
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.7 Security

```
CSTA-security
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) security( 128) }

DEFINITIONS ::=
BEGIN

EXPORTS
CSTASecurityData,TimeInfo;

CSTASecurityData ::= SEQUENCE
{
  messageSequenceNumber[0] IMPLICIT INTEGER      OPTIONAL,
  timestamp                TimeInfo              OPTIONAL,
  securityInfo              SecurityInfo          OPTIONAL}

SecurityInfo ::= CHOICE
{
  string      OCTET STRING,
  private     NULL}      -- The actual encoding is added here,
                        -- replacing NULL with another valid ASN.1 type.

TimeInfo ::= GeneralizedTime

END -- of CSTA-security
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.8 Common extensions

```

CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
DEFINITIONS ::=
BEGIN

EXPORTS
CSTACommonArguments, CSTAPrivateData;

IMPORTS
CSTASecurityData FROM CSTA-security
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) security( 128) };

CSTACommonArguments ::= [APPLICATION 30] IMPLICIT SEQUENCE
    {security      [0] IMPLICIT  CSTASecurityData      OPTIONAL,
     privateData   [1] IMPLICIT  SEQUENCE OF CSTAPrivateData  OPTIONAL }

CSTAPrivateData ::= CHOICE
{
    string      OCTET STRING,
    private     NULL}
    -- The actual encoding is added here,
    -- replacing NULL with another valid ASN.1 type.

END -- of CSTA-extension-types

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.9 Call control

```

CSTA-call-control
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) call-control( 130) }

DEFINITIONS ::=
BEGIN

EXPORTS
AutoOriginate, CallCharacteristics, CallLinkageData, CallLinkageDataList, CallQualifyingData,
ConsultOptions, NetworkCapability,
ProgressIndicator, ServicesPermitted;

IMPORTS
TimeInfo FROM CSTA-security
{ iso( 1) identified-organization( 3) icd-ecma( 12) standard( 0)
  csta3( 285) security( 128) }
CallControlServices, CallAssociatedServices, MediaAttachmentServices, RoutingServices,
VoiceUnitServices FROM CSTA-application-context-information-csta3
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) application-context-information( 200) };

AutoOriginate ::= ENUMERATED
{
  prompt          (0),
  doNotPrompt     (1) }

CallCharacteristics ::= BIT STRING
{
  acdCall          (0),
  priorityCall     (1),
  maintenanceCall (2),
  directAgent      (3),
  assistCall       (4),
  voiceUnitCall    (5)}

CallQualifyingData ::= OCTET STRING (SIZE(0..32))

ConsultOptions ::= ENUMERATED
{
  unrestricted     (0),
  consultOnly      (1),
  transferOnly     (2),
  conferenceOnly   (3) }

NetworkCapability ::= SEQUENCE
{
  networkType ENUMERATED
  {
    iSDNPublic      (0),
    nonISDNPublic   (1),
    iSDNPrivate     (2),
    nonISDNPrivate  (3),
    other           (4) },
  eventsProvided BIT STRING
  {
    bridged          (0),
    callCleared      (1),
    conferenced      (2),
    connectionCleared (3),
    delivered        (4),
    digitsDialed     (5),
    diverted         (6),
    established      (7),
    failed           (8),
    held             (9),
    networkCapabilitiesChange (10),
    networkReached   (11),
    offered          (12),
    originated       (13),
    queued           (14),
    retrieved        (15),
    serviceInitiated (16),
  }
}

```

```

transferred (17) } OPTIONAL}

ProgressIndicator ::= SEQUENCE
{
    progressLocation ENUMERATED
    {
        user (0),
        privateNetServingLocal (1),
        publicNetServingLocal (2),
        transitNetwork (3),
        publicNetServingRemote (4),
        privateNetServingRemote (5),
        localInterface (6),
        internationalNetwork (7),
        networkBeyondInterwk (8),
        other (9)
    },
    progressDescription ENUMERATED
    {
        iSDNProgressDesc (0),
        qSIGProgressDesc (1),
        other (2)
    }
}

ServicesPermitted ::= SEQUENCE
{
    callControlServices CallControlServices,
    callAssociatedServices CallAssociatedServices,
    mediaAttachmentServices MediaAttachmentServices,
    routingServices RoutingServices,
    voiceUnitServices VoiceUnitServices
}

CallLinkageDataList ::= SEQUENCE
{
    newCallLinkageData CallLinkageData,
    oldCallLinkageData CallLinkageData
}

CallLinkageData ::= SEQUENCE
{
    globalCallData GlobalCallData,
    threadData ThreadData OPTIONAL
}

GlobalCallData ::= SEQUENCE
{
    globalCallSwitchingSubDomainName IA5String (SIZE(1..64)) OPTIONAL,
    globalCallLinkageID GlobalCallLinkageID,
    callLinkageIDTimestamp TimeInfo OPTIONAL
}

GlobalCallLinkageID ::= CHOICE
{
    subDomainCallLinkageID [0] IMPLICIT OCTET STRING (SIZE(1..8)),
    globallyUniqueCallLinkageID [1] IMPLICIT OCTET STRING (SIZE(1..16))
}

ThreadData ::= SEQUENCE
{
    threadSwitchingSubDomainName IA5String (SIZE(1..64)) OPTIONAL,
    threadLinkageID ThreadLinkageID,
    threadIDTimeStamp TimeInfo OPTIONAL
}

ThreadLinkageID ::= CHOICE
{
    subDomainThreadID [0] IMPLICIT OCTET STRING (SIZE(1..8)),
    globallyUniqueThreadID [1] IMPLICIT OCTET STRING (SIZE(1..16))
}

END -- of CSTA-call-control

```

## 9.10 Capability Exchange

```

CSTA-capability-exchange
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) capability-exchange( 131) }
DEFINITIONS ::=
BEGIN

EXPORTS
CapExchangeServList, SystemStatusServList, MonitoringServList, SnapshotServList,
CallControlServList, CallControlEvtsList, CallAssociatedServList,
CallAssociatedEvtsList, MediaServList, MediaEvtsList, RouteingServList, PhysDevEvtsList,
PhysDevServList, LogicalEvtsList, LogicalServList, DeviceMaintEvtsList, IOServicesServList,
DataCollectionServList, VoiceUnitServList, VoiceUnitEvtsList, CDRServList,
VendorSpecificServList, VendorSpecificEvtsList, DeviceIDFormat, SwDomainFeatures,
SwAppearanceAddressability, SwAppearanceTypes, IgnoreUnsupportedParameters, PauseTime,
TimeStampMode, MiscMonitorCaps, MaxLengthParameters, FilterThreshold, ServiceCrossRefID,
DeviceCategory, GroupDeviceAttributes, NamedDeviceTypes, ACDModels, AgentLogOnModels,
AppearanceType, TransAndConfSetup, MediaServiceCapsList, LogDevServList, LogDevEvtsList;

IMPORTS
MediaServiceType, MediaServiceInstanceID, ConnectionMode, ConnectionModeBMap
  FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) };

ServiceCrossRefID ::= OCTET STRING (SIZE(0..4))

-- Capability Bitmaps

LogDevServList ::= SEQUENCE
{
  callControlServList      [0] IMPLICIT CallControlServList      OPTIONAL,
  callAssociatedServList   [1] IMPLICIT CallAssociatedServList   OPTIONAL,
  logicalServList          [2] IMPLICIT LogicalServList           OPTIONAL,
  mediaServList            [3] IMPLICIT MediaServList             OPTIONAL,
  ioservicesServList       [4] IMPLICIT IOServicesServList        OPTIONAL,
  dataCollectionServList   [5] IMPLICIT DataCollectionServList    OPTIONAL,
  voiceUnitServList        [6] IMPLICIT VoiceUnitServList         OPTIONAL}

LogDevEvtsList ::= SEQUENCE
{
  callControlEvtsList     [0] IMPLICIT CallControlEvtsList       OPTIONAL,
  callAssociatedEvtsList  [1] IMPLICIT CallAssociatedEvtsList     OPTIONAL,
  logicalEvtsList         [2] IMPLICIT LogicalEvtsList            OPTIONAL,
  mediaEvtsList           [3] IMPLICIT MediaEvtsList              OPTIONAL,
  voiceUnitEvtsList       [4] IMPLICIT VoiceUnitEvtsList          OPTIONAL}

CapExchangeServList ::= SEQUENCE
{
  getLogicalDeviceInformation [0] IMPLICIT GetLogicalDeviceInformation OPTIONAL,
  getPhysicalDeviceInformation [1] IMPLICIT GetPhysicalDeviceInformation OPTIONAL,
  getSwitchingFunctionCaps    NULL,
  getSwitchingFunctionDevices [2] IMPLICIT GetSwitchingFunctionDevices OPTIONAL,
  switchingFunctionDevices    [3] IMPLICIT SwitchingFunctionDevices OPTIONAL}

GetLogicalDeviceInformation ::= BIT STRING
{
  privateDataInReq          ( 0),
  namedDeviceTypeInAck      ( 1),
  shortFormDeviceIDInAck    ( 2),
  miscMonitorCapsInAck      ( 3),
  maxCallBacksInAck         ( 4),
  maxAutoAnswerRingsInAck  ( 5),
  maxActiveCallsInAck       ( 6),
  maxHeldCallsInAck         ( 7),
  maxFwdSettingsInAck       ( 8),
  maxDevicesInConfInAck     ( 9),
  transAndConfSetupParameter (10),
  transAndConfSetupConsultationInAck (11),

```

```

transAndConfSetupHoldMakeInAck          (12),
transAndConfSetupAlternateInAck          (13),
transAndConfSetupTwoCallsHoldInAck      (14),
transAndConfSetupTwoCallsConnectedInAck (15),
mediaClassSupportInAck                   (16),
connectionRateListInAck                  (17),
delayToleranceListInAck                  (18),
numberOfChannelsInAck                     (19),
maxChannelBindInAck                       (20),
privateDataInAck                          (21)}

```

```

GetPhysicalDeviceInformation ::= BIT STRING
{
    privateDataInReq          ( 0),
    namedDeviceTypesInAck    ( 1),
    otherLogicalDeviceListInAck ( 2),
    deviceModelNameInAck     ( 3),
    maxDisplaysInAck         ( 4),
    maxButtonsInAck          ( 5),
    maxLampsInAck            ( 6),
    maxRingPatternsInAck     ( 7),
    privateDataInAck         ( 8)}

```

```

GetSwitchingFunctionDevices ::= BIT STRING
{
    requestedDeviceID          ( 0),
    requestedDeviceCategoryACD ( 1),
    requestedDeviceCategoryACDGroup ( 2),
    requestedDeviceCategoryHuntGroup ( 3),
    requestedDeviceCategoryPickGroup ( 4),
    requestedDeviceCategoryOtherGroup ( 5),
    requestedDeviceCategoryNetwInterface ( 6),
    requestedDeviceCategoryPark ( 7),
    requestedDeviceCategoryRouteingDevice ( 8),
    requestedDeviceCategoryStation ( 9),
    requestedDeviceCategoryVoiceUnit (10),
    requestedDeviceCategoryOther (11),
    privateData                (12),
    privateDataInAck           (13)}

```

```

SwitchingFunctionDevices ::= BIT STRING
{
    segmentID          ( 0),
    deviceListDeviceCategory ( 1),
    deviceListNamedDeviceTypes ( 2),
    deviceListDeviceAttributes ( 3),
    deviceListDeviceModelName ( 4),
    privateData        ( 5)}

```

```

SystemStatusServList ::= SEQUENCE
{
    changeSystemStatusFilter [ 0] IMPLICIT ChangeSystemStatusFilter OPTIONAL,
    systemRegister           [ 1] IMPLICIT SystemRegister          OPTIONAL,
    systemStatusRegisterAbort [ 2] IMPLICIT SystemStatusRegisterAbort OPTIONAL,
    systemStatusRegisterCancel [ 3] IMPLICIT SystemStatusRegisterCancel OPTIONAL,
    requestSystemStatus      [ 4] IMPLICIT RequestSystemStatus     OPTIONAL,
    systemStatus              [ 5] IMPLICIT SystemStatus            OPTIONAL,
    swFunctionCapsChanged    [ 6] IMPLICIT SwFunctionCapsChanged   OPTIONAL,
    swFunctionDevicesChanged [ 7] IMPLICIT SwFunctionDevicesChanged OPTIONAL}

```

```

ChangeSystemStatusFilter ::= BIT STRING
{
    requestedStatusFilterInitializing ( 0), -- optional parameters
    requestedStatusFilterEnabled      ( 1), -- optional parameters
    requestedStatusFilterNormal       ( 2), -- optional parameters
    requestedStatusFilterMessageLost ( 3), -- optional parameters
    requestedStatusFilterDisabled     ( 4), -- optional parameters
    requestedStatusFilterPartiallyDisabled ( 5), -- optional parameters
    requestedStatusFilterOverloadImminent ( 6), -- optional parameters
    requestedStatusFilterOverloadReached ( 7), -- optional parameters
    requestedStatusFilterOverloadRelieved ( 8), -- optional parameters
    privateData                       ( 9), -- optional parameters
}

```

```

        privateDataInAck                (10)} -- optional parameters

SystemRegister ::= BIT STRING
{
    requestTypesSystemStatus            ( 0), -- optional parameters
    requestTypesRequestSystemStatus     ( 1), -- optional parameters
    requestTypesSwitchingFunctionCapsChanged ( 2), -- optional parameters
    requestTypesSwitchingFunctionDevicesChanged ( 3), -- optional parameters
    statusFilterInitializing             ( 4), -- optional parameters
    statusFilterEnabled                  ( 5), -- optional parameters
    statusFilterNormal                   ( 6), -- optional parameters
    statusFilterMessageLost              ( 7), -- optional parameters
    statusFilterDisabled                 ( 8), -- optional parameters
    statusFilterPartiallyDisabled        ( 9), -- optional parameters
    statusFilterOverloadImminent         (10), -- optional parameters
    statusFilterOverloadReached          (11), -- optional parameters
    statusFilterOverloadRelieved         (12), -- optional parameters
    privateDataOctetString               (13), -- optional parameters
    privateData                          (14), -- optional parameters
    privateDataOctetStringInAck          (15), -- optional parameters
    privateDataInAck                     (16)} -- optional parameters

SystemStatusRegisterCancel ::= BIT STRING
{
    privateData                (0), -- optional parameters
    privateDataInAck           (1)} -- optional parameters

SystemStatusRegisterAbort ::= BIT STRING
{
    privateData                (0), -- optional parameters
    privateDataInAck           (1)} -- optional parameters

RequestSystemStatus ::= BIT STRING
{
    privateDataInReq           ( 0), -- optional parameters
    systemStatusInitializing   ( 1), -- optional parameters
    systemStatusEnabled        ( 2), -- optional parameters
    systemStatusNormal         ( 3), -- optional parameters
    systemStatusMessageLost    ( 4), -- optional parameters
    systemStatusDisabled       ( 5), -- optional parameters
    systemStatusPartiallyDisabled ( 6), -- optional parameters
    systemStatusOverloadImminent ( 7), -- optional parameters
    systemStatusOverloadReached ( 8), -- optional parameters
    systemStatusOverloadRelieved ( 9), -- optional parameters
    privateDataInAck           (10), -- optional parameters
    switchingFunctionSupportsSending (11), -- misc characteristics
    switchingFunctionSupportsReceiving (12)} -- misc characteristics

SystemStatus ::= BIT STRING
{
    systemStatusInitializing   ( 0), -- optional parameters
    systemStatusEnabled        ( 1), -- optional parameters
    systemStatusNormal         ( 2), -- optional parameters
    systemStatusMessageLost    ( 3), -- optional parameters
    systemStatusDisabled       ( 4), -- optional parameters
    systemStatusPartiallyDisabled ( 5), -- optional parameters
    systemStatusOverloadImminent ( 6), -- optional parameters
    systemStatusOverloadReached ( 7), -- optional parameters
    systemStatusOverloadRelieved ( 8), -- optional parameters
    privateData                ( 9), -- optional parameters
    privateDataInAck           (10), -- optional parameters
    switchingFunctionSupportsSending (11), -- misc characteristics
    switchingFunctionSupportsReceiving (12)} -- misc characteristics

SwFunctionCapsChanged ::= BIT STRING
{
    privateData                (0), -- optional parameters
    privateDataInAck           (1)} -- optional parameters

SwFunctionDevicesChanged ::= BIT STRING
{
    privateData                (0), -- optional parameters
    privateDataInAck           (1)} -- optional parameters

MonitoringServList ::= SEQUENCE

```

```

{      changeMonitorFilter      [ 0 ] IMPLICIT ChangeMonitorFilter      OPTIONAL,
      monitorStart              [ 1 ] IMPLICIT MonitorStart          OPTIONAL,
      monitorStop               [ 2 ] IMPLICIT MonitorStop         OPTIONAL}

ChangeMonitorFilter ::= BIT STRING
{      privateData              ( 0 ),      -- optional parameters
      privateDataInAck         ( 1 )}      -- optional parameters

MonitorStart ::= BIT STRING
{      monitorObjectDevice      ( 0 ),      -- optional parameters
      monitorObjectCall        ( 1 ),      -- optional parameters
      requestedMonitorFilter    ( 2 ),      -- optional parameters
      monitorTypeCall          ( 3 ),      -- optional parameters
      monitorTypeDevice        ( 4 ),      -- optional parameters
      monitorMediaClassParameter ( 5 ),      -- optional parameters
      monitorMediaClassAudio    ( 6 ),      -- optional parameters
      monitorMediaClassData     ( 7 ),      -- optional parameters
      monitorMediaClassImage    ( 8 ),      -- optional parameters
      monitorMediaClassVoice    ( 9 ),      -- optional parameters
      monitorExistingCallsInAck (10),      -- optional parameters
      privateDataInAck         (11),      -- optional parameters
      callIDOnly               (12),      -- miscellaneous characteristics
      swDomainDefaultForMonitorTypeIsDevice (13)} -- miscellaneous characteristics

MonitorStop ::= BIT STRING
{      privateData              ( 0 ),      -- optional parameters
      privateDataInAck         ( 1 ),      -- optional parameters
      switchingFunctionSupportsSending ( 2 ),      -- misc characteristics
      switchingFunctionSupportsReceiving ( 3 )} -- misc characteristics

SnapshotServList ::= SEQUENCE
{      snapshotCall            [0] IMPLICIT SnapshotCall,
      snapshotDevice          [1] IMPLICIT SnapshotDevice,
      snapshotCallData        [2] IMPLICIT SnapshotCallData,
      snapshotDeviceData      [3] IMPLICIT SnapshotDeviceData}

SnapshotCall ::= BIT STRING
{      privateData              ( 0 ),      -- optional parameters
      locaConnectionStateInAck ( 1 ),      -- optional parameters
      mediaServiceInfoListInAck ( 2 ),      -- optional parameters
      mediaServiceVersionInAck ( 3 ),      -- optional parameters
      mediaServiceInstanceInAck ( 4 ),      -- optional parameters
      mediaStreamID           ( 5 ),      -- optional parameters
      connectionInformation    ( 6 ),      -- optional parameters
      mediaCallCharacteristicsInAck ( 7 ),      -- optional parameters
      callCharacteristicsInAck ( 8 ),      -- optional parameters
      callingDeviceInAck       ( 9 ),      -- optional parameters
      calledDeviceInAck        (10),      -- optional parameters
      privateDataInAck         (11),      -- optional parameters
      callIDOnly               (12),      -- miscellaneous characteristics
      reportsWithSnapsCallData (13)} -- miscellaneous characteristics

SnapshotDevice ::= BIT STRING
{      privateData              ( 0 ),      -- optional parameters
      localCallStateCompoundCallState ( 1 ),      -- optional parameters
      localCallStateSimpleCallState ( 2 ),      -- optional parameters
      localCallStateUnknown    ( 3 ),      -- optional parameters
      mediaServiceInfoListInAck ( 4 ),      -- optional parameters
      mediaServiceVersionInAck ( 5 ),      -- optional parameters
      mediaServiceInstanceInAck ( 6 ),      -- optional parameters
      mediaStreamID           ( 7 ),      -- optional parameters
      connectionInformation    ( 8 ),      -- optional parameters
      mediaCallCharacteristicsInAck ( 9 ),      -- optional parameters
      privateDataInAck         (10),      -- optional parameters
      reportsWithSnapsDeviceData (11)} -- miscellaneous characteristics

SnapshotCallData ::= BIT STRING
{      segmentID                ( 0 ),      -- optional parameters

```

```

localConnectionState      ( 1),  -- optional parameters
mediaServiceInformationList ( 2),  -- optional parameters
mediaServiceVersion       ( 3),  -- optional parameters
mediaServiceInstance      ( 4),  -- optional parameters
mediaStreamID             ( 5),  -- optional parameters
connectionInformation     ( 6),  -- optional parameters
privateData               ( 7)}  -- optional parameters

SnapshotDeviceData ::= BIT STRING
{
    segmentID                ( 0),  -- optional parameters
    localCallStateCompoundCallState ( 1),  -- optional parameters
    localCallStateSimpleCallState ( 2),  -- optional parameters
    localCallStateUnknown    ( 3),  -- optional parameters
    mediaServiceInformationList ( 4),  -- optional parameters
    mediaServiceVersion      ( 5),  -- optional parameters
    mediaServiceInstance     ( 6),  -- optional parameters
    mediaStreamID            ( 7),  -- optional parameters
    connectionInformation    ( 8),  -- optional parameters
    mediaCallCharacteristics ( 9),  -- optional parameters
    privateData              (10)}  -- optional parameters

CallControlServList ::= SEQUENCE
{
    acceptCall      [0] IMPLICIT AcceptCall      OPTIONAL,
    alternateCall  [1] IMPLICIT AlternateCall    OPTIONAL,
    answerCall     [2] IMPLICIT AnswerCall      OPTIONAL,
    callBack       [3] IMPLICIT CallBack        OPTIONAL,
    callBackMessage [4] IMPLICIT CallBackMessage OPTIONAL,
    campOnCall    [5] IMPLICIT CampOnCall      OPTIONAL,
    clearCall     [6] IMPLICIT ClearCall        OPTIONAL,
    clearConnection [7] IMPLICIT ClearConnection OPTIONAL,
    conferenceCall [8] IMPLICIT ConferenceCall  OPTIONAL,
    consultationCall [9] IMPLICIT ConsultationCall OPTIONAL,
    deflectCall   [10] IMPLICIT DeflectCall     OPTIONAL,
    dialDigits    [11] IMPLICIT DialDigits     OPTIONAL,
    directedPickupCall [12] IMPLICIT DirectedPickupCall OPTIONAL,
    groupPickupCall [13] IMPLICIT GroupPickupCall OPTIONAL,
    holdCall      [14] IMPLICIT HoldCall       OPTIONAL,
    intrudeCall   [15] IMPLICIT IntrudeCall    OPTIONAL,
    joinCall      [16] IMPLICIT JoinCall       OPTIONAL,
    makeCall      [17] IMPLICIT MakeCall       OPTIONAL,
    makePredictiveCall [18] IMPLICIT MakePredictiveCall OPTIONAL,
    parkCall      [19] IMPLICIT ParkCall       OPTIONAL,
    reconnectCall [20] IMPLICIT ReconnectCall  OPTIONAL,
    retrieveCall  [21] IMPLICIT RetrieveCall   OPTIONAL,
    singleStepConference [22] IMPLICIT SingleStepConference OPTIONAL,
    singleStepTransfer [23] IMPLICIT SingleStepTransfer OPTIONAL,
    transferCall  [24] IMPLICIT TransferCall   OPTIONAL}

AcceptCall ::= BIT STRING
{
    correlatorData ( 0),  -- optional parameters
    userData       ( 1),  -- optional parameters
    privateData    ( 2),  -- optional parameters
    privateDataInAck ( 3),  -- optional parameters
    deviceIDOnly  ( 4),  -- misc characteristics
    ackModelMultiStep ( 5)}  -- misc characteristics

AlternateCall ::= BIT STRING
{
    alerting ( 0),  -- initial states heldCall
    hold     ( 1),  -- initial states heldCall
    queued   ( 2),  -- initial states heldCall
    connectionReservation ( 3),  -- optional parameters
    consultOptionConsultOnly ( 4),  -- optional parameters
    consultOptionTransferOnly ( 5),  -- optional parameters
    consultOptionConferenceOnly ( 6),  -- optional parameters
    consultOptionUnrestricted ( 7),  -- optional parameters
    privateData ( 8),  -- optional parameters
    privateDataInAck ( 9),  -- optional parameters
    deviceIDOnly (10),  -- misc characteristics

```

```

    ackModelMultiStep          (11),  -- misc characteristics
    supportsOfferedModeOfAlerting (12)} -- misc characteristics

AnswerCall ::= BIT STRING
{
    alerting          ( 0),  -- initial states
    initiated         ( 1),  -- initial states
    queued            ( 2),  -- initial states
    correlatorData    ( 3),  -- optional parameters
    userData          ( 4),  -- optional parameters
    privateData       ( 5),  -- optional parameters
    privateDataInAck ( 6),  -- optional parameters
    deviceIDOnly      ( 7),  -- misc characteristics
    supportsOfferedModeOfAlerting ( 8),  -- misc characteristics
    ackModelMultiStep ( 9)}  -- misc characteristics

CallBack ::= BIT STRING
{
    alerting          ( 0),  -- initial states
    null              ( 1),  -- initial states
    failed            ( 2),  -- initial states
    queued            ( 3),  -- initial states
    callCharacteristics ( 4),  -- optional parameters
    privateData       ( 5),  -- optional parameters
    targetDeviceInAck ( 6),  -- optional parameters
    privateDataInAck ( 7),  -- optional parameters
    deviceIDOnly      ( 8),  -- misc characteristics
    moreCallBacksNegAck ( 9),  -- misc characteristics
    ackModelMultiStep (10)}  -- misc characteristics

CallBackMessage ::= BIT STRING
{
    alerting          ( 0),  -- initial states
    null              ( 1),  -- initial states
    failed            ( 2),  -- initial states
    queued            ( 3),  -- initial states
    privateData       ( 4),  -- optional parameters
    targetDeviceInAck ( 5),  -- optional parameters
    privateDataInAck ( 6),  -- optional parameters
    deviceIDOnly      ( 7),  -- misc characteristics
    moreCallBacksNegAck ( 8),  -- misc characteristics
    ackModelMultiStep ( 9)}  -- misc characteristics

CampOnCall ::= BIT STRING
{
    privateData       ( 0),  -- optional parameters
    privateDataInAck ( 1),  -- optional parameters
    deviceIDOnly      ( 2),  -- misc characteristics
    ackModelMultiStep ( 3)}  -- misc characteristics

ClearCall ::= BIT STRING
{
    alerting          ( 0),  -- initial states
    connected         ( 1),  -- initial states
    failed            ( 2),  -- initial states
    queued            ( 3),  -- initial states
    initiated         ( 4),  -- initial states
    hold              ( 5),  -- initial states
    userData          ( 6),  -- optional parameters
    privateData       ( 7),  -- optional parameters
    privateDataInAck ( 8),  -- optional parameters
    deviceIDOnly      ( 9),  -- misc characteristics
    callIDOnly        (10),  -- misc characteristics
    ackModelMultiStep (11)}  -- misc characteristics

ClearConnection ::= BIT STRING
{
    alerting          ( 0),  -- initial states
    connected         ( 1),  -- initial states
    fail              ( 2),  -- initial states
    queued            ( 3),  -- initial states
    initiated         ( 4),  -- initial states
    hold              ( 5),  -- initial states
    correlatorData    ( 6),  -- optional parameters

```

```

        userData                ( 7),  -- optional parameters
        privateData             ( 8),  -- optional parameters
        privateDataInAck        ( 9),  -- optional parameters
        deviceIDOnly            (10),  -- misc characteristics
        ackModelMultiStep       (11)}  -- misc characteristics

ConferenceCall ::= BIT STRING
{
    activeCallConnected        ( 0),  -- initial states
    activeCallHold             ( 1),  -- initial states
    heldCallConnected         ( 2),  -- initial states
    heldCallHold               ( 3),  -- initial states
    privateData                ( 4),  -- optional parameters
    connectionParameterInAck   ( 5),  -- optional parameters
    endpointDeviceID           ( 6),  -- optional parameters
    resultingConnectionInfo    ( 7),  -- optional parameters
    conferenceCallOnfoInAck    ( 8),  -- optional parameters
    privateDataInAck           ( 9),  -- optional parameters
    protectedAgainstClearing   (10),  -- misc characteristics
    deviceIDOnly               (11),  -- misc characteristics
    ackModelMultiStep         (12)}  -- misc characteristics

ConsultationCall ::= BIT STRING
{
    connectionReservation      ( 0),  -- optional parameters
    accountCode                 ( 1),  -- optional parameters
    authCode                    ( 2),  -- optional parameters
    correlatorData              ( 3),  -- optional parameters
    userData                    ( 4),  -- optional parameters
    callCharacteristics         ( 5),  -- optional parameters
    callCharacteristicsACDCall  ( 6),  -- optional parameters
    callCharacteristicsPriorityCall ( 7),  -- optional parameters
    callCharacteristicsMaintenanceCall ( 8),  -- optional parameters
    callCharacteristicsDirectAgent ( 9),  -- optional parameters
    callCharacteristicsAssistCall (10),  -- optional parameters
    callCharacteristicsVoiceUnitCall (11),  -- optional parameters
    mediaCallCharacteristics    (12),  -- optional parameters
    callingConnectionInfo      (13),  -- optional parameters
    flowDirectionTransmit       (14),  -- optional parameters
    flowDirectionReceive        (15),  -- optional parameters
    flowDirectionTransmitAndReceive (16),  -- optional parameters
    numberOfChannels            (17),  -- optional parameters
    consultOptionConsultOnly    (18),  -- optional parameters
    consultOptionTransferOnly   (19),  -- optional parameters
    consultOptionConferenceOnly (20),  -- optional parameters
    consultOptionUnrestricted   (21),  -- optional parameters
    privateData                 (22),  -- optional parameters
    initiatedCallInfoInAck      (23),  -- optional parameters
    privateDataInAck            (24),  -- optional parameters
    deviceIDOnly                (25),  -- misc characteristics
    multiStage                  (26),  -- misc characteristics
    supportsAdjustmentOfMediaCharacteristics (27),  -- misc characteristics
    ackModelMultiStep          (28)}  -- misc characteristics

DeflectCall ::= BIT STRING
{
    alerting                    ( 0),  -- initial states
    connected                   ( 1),  -- initial states
    failed                       ( 2),  -- initial states
    hold                         ( 3),  -- initial states
    queued                       ( 4),  -- initial states
    correlatorData               ( 5),  -- optional parameters
    userData                     ( 6),  -- optional parameters
    privateData                  ( 7),  -- optional parameters
    privateDataInAck             ( 8),  -- optional parameters
    deviceIDOnly                 ( 9),  -- misc characteristics
    ackModelMultiStep           (10)}  -- misc characteristics

DialDigits ::= BIT STRING
{
    correlatorData              ( 0),  -- optional parameters
    privateData                 ( 1),  -- optional parameters

```

```

privateDataInAck          ( 2),  -- optional parameters
deviceIDOnly              ( 3),  -- misc characteristics
ackModelMultiStep        ( 4)}  -- misc characteristics

DirectedPickupCall ::= BIT STRING
{
    alerting                ( 0),  -- initial states
    hold                    ( 1),  -- initial states
    queued                  ( 2),  -- initial states
    connected               ( 3),  -- initial states
    correlatorData          ( 4),  -- optional parameters
    userData                ( 5),  -- optional parameters
    privateData             ( 6),  -- optional parameters
    pickedCallInAck         ( 7),  -- optional parameters
    pickedCallInfoInAck     ( 8),  -- optional parameters
    privateDataInAck        ( 9),  -- optional parameters
    deviceIDOnly            (10),  -- misc characteristics
    supportsOfferedModeOfAlerting (11), -- misc characteristics
    supportsPrompting       (12),  -- misc characteristics
    promptingMode           (13),  -- misc characteristics
    ackModelMultiStep       (14)}  -- misc characteristics

GroupPickupCall ::= BIT STRING
{
    alerting                ( 0),  -- initial states
    connected               ( 1),  -- initial states
    hold                    ( 2),  -- initial states
    queued                  ( 3),  -- initial states
    pickGroup               ( 4),  -- optional parameters
    correlatorData          ( 5),  -- optional parameters
    userData                ( 6),  -- optional parameters
    privateData             ( 7),  -- optional parameters
    pickedCallInAck         ( 8),  -- optional parameters
    pickedCallInfoInAck     ( 9),  -- optional parameters
    privateDataInAck        (10),  -- optional parameters
    supportsPrompting       (11),  -- misc characteristics
    promptingMode           (12),  -- misc characteristics
    supportsOfferedModeOfAlerting (13), -- misc characteristics
    ackModelMultiStep       (14)}  -- misc characteristics

HoldCall ::= BIT STRING
{
    connectionReservation   ( 0),  -- optional parameters
    privateData             ( 1),  -- optional parameters
    privateDataInAck        ( 2),  -- optional parameters
    deviceIDOnly            ( 3),  -- misc characteristics
    ackModelMultiStep       ( 4)}  -- misc characteristics

IntrudeCall ::= BIT STRING
{
    participationTypeSilent ( 0),  -- optional parameters
    participationTypeActive ( 1),  -- optional parameters
    userData                ( 2),  -- optional parameters
    privateData             ( 3),  -- optional parameters
    conferencedCallInfoInAck ( 4),  -- optional parameters
    privateDataInAck        ( 5),  -- optional parameters
    deviceIDOnly            ( 6),  -- misc characteristics
    supportsConference       ( 7),  -- misc characteristics
    supportsAlternate       ( 8),  -- misc characteristics
    ackModelMultiStep       ( 9)}  -- misc characteristics

JoinCall ::= BIT STRING
{
    autoOriginatePrompt     ( 0),  -- optional parameters
    autoOriginateDoNotPrompt ( 1),  -- optional parameters
    participationTypeSilent ( 2),  -- optional parameters
    participationTypeActive ( 3),  -- optional parameters
    accountCode              ( 4),  -- optional parameters
    authCode                 ( 5),  -- optional parameters
    correlatorData          ( 6),  -- optional parameters
    userData                ( 7),  -- optional parameters
    conferencedCallInAck     ( 8),  -- optional parameters
    conferencedCallInfoInAck ( 9),  -- optional parameters

```

```

privateDataInAck          (10), -- optional parameters
deviceIDOnly              (11), -- misc characteristics
supportsPrompting         (12), -- misc characteristics
promptingMode             (13), -- misc characteristics
ackModelMultiStep        (14)} -- misc characteristics

MakeCall ::= BIT STRING
{
    initiated              ( 0), -- initials states
    null                   ( 1), -- initials states
    accountCode           ( 2), -- optional parameters
    authCode              ( 3), -- optional parameters
    autoOriginatePrompt   ( 4), -- optional parameters
    autoOriginateDoNotPrompt ( 5), -- optional parameters
    correlatorData        ( 6), -- optional parameters
    userData              ( 7), -- optional parameters
    callCharacteristics    ( 8), -- optional parameters
    callCharacteristicsACDCall ( 9), -- optional parameters
    callCharacteristicsPriorityCall (10), -- optional parameters
    callCharacteristicsMaintenanceCall (11), -- optional parameters
    callCharacteristicsDirectAgent (12), -- optional parameters
    callCharacteristicsAssistCall (13), -- optional parameters
    callCharacteristicsVoiceUnitCall (14), -- optional parameters
    mediaCallCharacteristics (15), -- optional parameters
    callingConnectionInfo (16), -- optional parameters
    privateData           (17), -- optional parameters
    initiatedCallInfoInAck (18), -- optional parameters
    privateDataInAck      (19), -- optional parameters
    multiStage            (20), -- misc characteristics
    supportsPrompting     (21), -- misc characteristics
    promptingMode         (22), -- misc characteristics
    offHook               (23), -- misc characteristics
    mediaCharacteristicsAdjustable (24), -- misc characteristics
    ackModelMultiStep    (25)} -- misc characteristics

MakePredictiveCall ::= BIT STRING
{
    signallingDetection          ( 0), -- optional parameters
    signallingConditionCallDelivered ( 1), -- optional parameters
    signallingConditionCallEstablished ( 2), -- optional parameters
    signallingConditionActionDestinationDetection ( 3), -- optional parameters
    signallingConditionActionRemainConnected ( 4), -- optional parameters
    destinationDetection        ( 5), -- optional parameters
    destinationConditionHumanVoice ( 6), -- optional parameters
    destinationConditionAnsweringMachine ( 7), -- optional parameters
    destinationConditionFax      ( 8), -- optional parameters
    destinationActionClearConnection ( 9), -- optional parameters
    destinationActionRemainConnected (10), -- optional parameters
    defaultActionClearConnection (11), -- optional parameters
    defaultActionRemainConnected (12), -- optional parameters
    accountCode                  (13), -- optional parameters
    authCode                     (14), -- optional parameters
    autoOriginatePrompt          (15), -- optional parameters
    autoOriginateDoNotPrompt     (16), -- optional parameters
    alertTime                    (17), -- optional parameters
    correlatorData               (18), -- optional parameters
    callCharacteristics           (19), -- optional parameters
    callCharacteristicsACDCall    (20), -- optional parameters
    callCharacteristicsPriorityCall (21), -- optional parameters
    callCharacteristicsMaintenanceCall (22), -- optional parameters
    callCharacteristicsDirectAgent (23), -- optional parameters
    callCharacteristicsAssistCall (24), -- optional parameters
    callCharacteristicsVoiceUnitCall (25), -- optional parameters
    userData                     (26), -- optional parameters
    privateData                  (27), -- optional parameters
    initiatedCallInfoInAck       (28), -- optional parameters
    privateDataInAck             (29), -- optional parameters
    deviceIDOnly                 (30), -- misc characteristics
    supportsPrompting            (31), -- misc characteristics
    promptingMode                (32), -- misc characteristics
}

```

```

        reservesCallingDevice                (33), -- misc characteristics
        ackModelMultiStep                    (34)} -- misc characteristics

ParkCall ::= BIT STRING
{
    hold                ( 0), -- initial states
    connected            ( 1), -- initial states
    correlatorData      ( 2), -- optional parameters
    privateData          ( 3), -- optional parameters
    parkedToInAck       ( 4), -- optional parameters
    privateDataInAck    ( 5), -- optional parameters
    deviceIDOnly        ( 6), -- misc characteristics
    ackModelMultiStep   ( 7)} -- misc characteristics

ReconnectCall ::= BIT STRING
{
    alerting            ( 0), -- initial states
    connected            ( 1), -- initial states
    fail                ( 2), -- initial states
    initiated            ( 3), -- initial states
    queued              ( 4), -- initial states
    privateData          ( 5), -- optional parameters
    privateDataInAck    ( 6), -- optional parameters
    deviceIDOnly        ( 7), -- misc characteristics
    ackModelMultiStep   ( 8)} -- misc characteristics

RetrieveCall ::= BIT STRING
{
    privateData          ( 0), -- optional parameters
    privateDataInAck    ( 1), -- optional parameters
    deviceIDOnly        ( 2), -- misc characteristics
    ackModelMultiStep   ( 3)} -- misc characteristics

SingleStepConference ::= BIT STRING
{
    participationTypeActive ( 0), -- optional parameters
    participationTypeSilent ( 1), -- optional parameters
    accountCode            ( 2), -- optional parameters
    authCode               ( 3), -- optional parameters
    correlatorData         ( 4), -- optional parameters
    userData               ( 5), -- optional parameters
    privateData            ( 6), -- optional parameters
    conferencedCallInfoInAck ( 7), -- optional parameters
    privateDataInAck      ( 8), -- optional parameters
    deviceIDOnly          ( 9), -- misc characteristics
    ackModelMultiStep     (10)} -- misc characteristics

SingleStepTransfer ::= BIT STRING
{
    accountCode          ( 0), -- optional parameters
    authCode             ( 1), -- optional parameters
    correlatorData       ( 2), -- optional parameters
    userData             ( 3), -- optional parameters
    privateData          ( 4), -- optional parameters
    connectionParameterInAck ( 5), -- optional parameters
    endpointDeviceID     ( 6), -- optional parameters
    resultionConnectionInformation ( 7), -- optional parameters
    transferredCollInfoInAck ( 8), -- optional parameters
    privateDataInAck    ( 9), -- optional parameters
    deviceIDOnly        (10), -- misc characteristics
    multipleDevices      (11), -- misc characteristics
    ackModelMultiStep   (12)} -- misc characteristics

TransferCall ::= BIT STRING
{
    activeCallConnected ( 0), -- initial states
    activeCallHold      ( 1), -- initial states
    heldCallConnected   ( 2), -- initial states
    heldCallHold        ( 3), -- initial states
    privateData         ( 4), -- optional parameters
    connectionsParameterInAck ( 5), -- optional parameters
    endpointDeviceID    ( 6), -- optional parameters
    resultingConnectionInformation ( 7), -- optional parameters
    transferredCallInfoInAck ( 8), -- optional parameters

```

```

privateDataInAck          ( 9),  -- optional parameters
deviceIDOnly              (10),  -- misc characteristics
multipleDevices           (11),  -- misc characteristics
ackModelMultiStep        (12)} -- misc characteristics

```

CallControlEvtsList ::= SEQUENCE

```

{
    bridged                [0] IMPLICIT Bridged                OPTIONAL,
    callCleared            [1] IMPLICIT CallCleared            OPTIONAL,
    conferenced            [2] IMPLICIT Conferenced            OPTIONAL,
    connectionCleared     [3] IMPLICIT ConnectionCleared     OPTIONAL,
    delivered              [4] IMPLICIT Delivered              OPTIONAL,
    digitsDialed           [5] IMPLICIT DigitsDialed           OPTIONAL,
    diverted               [6] IMPLICIT Diverted               OPTIONAL,
    established            [7] IMPLICIT Established            OPTIONAL,
    failed                 [8] IMPLICIT Failed                 OPTIONAL,
    held                   [9] IMPLICIT Held                   OPTIONAL,
    netwCapsChanged       [10] IMPLICIT NetwCapsChanged       OPTIONAL,
    netwReached            [11] IMPLICIT NetwReached            OPTIONAL,
    offered                [12] IMPLICIT Offered                OPTIONAL,
    originated             [13] IMPLICIT Originated             OPTIONAL,
    queued                [14] IMPLICIT Queued                OPTIONAL,
    retrieved              [15] IMPLICIT Retrieved              OPTIONAL,
    serviceInitiated      [16] IMPLICIT ServiceInitiated      OPTIONAL,
    transferred           [17] IMPLICIT Transferred           OPTIONAL }

```

Bridged ::= BIT STRING

```

{
    correlatorData        ( 0),  -- optional parameters
    userData              ( 1),  -- optional parameters
    servicesPermitted     ( 2),  -- optional parameters
    mediaCallCharacteristics ( 3), -- optional parameters
    callCharacteristics   ( 4),  -- optional parameters
    bridgedConnectionInfo ( 5),  -- optional parameters
    callLinkageData       ( 7),  -- optional parameters
    privateData           ( 6)} -- optional parameters

```

CallCleared ::= BIT STRING

```

{
    correlatorData        ( 0),  -- optional parameters
    userData              ( 1),  -- optional parameters
    mediaCallCharacteristics ( 2), -- optional parameters
    callCharacteristics   ( 3),  -- optional parameters
    callLinkageData       ( 6),  -- optional parameters
    privateData           ( 4),  -- optional parameters
    callIDOnly            ( 5)} -- miscellaneous characteristics

```

Conferenced ::= BIT STRING

```

{
    confereceConnectionsEndpointDeviceID ( 0),  -- optional parameters
    confereceConnectionsresultingConnectionInfo ( 1), -- optional parameters
    userData              ( 2),  -- optional parameters
    servicesPermitted     ( 3),  -- optional parameters
    mediaCallCharacteristics ( 4), -- optional parameters
    callCharacteristics   ( 5),  -- optional parameters
    bridgedConnectionInfo ( 6),  -- optional parameters
    privateData           ( 7)} -- optional parameters

```

ConnectionCleared ::= BIT STRING

```

{
    correlatorData        ( 0),  -- optional parameters
    userData              ( 1),  -- optional parameters
    chargingInfo          ( 2),  -- optional parameters
    numberUnitsNumberOfChargingUnits ( 3), -- optional parameters
    numberUnitsTypeOfUnits ( 4),  -- optional parameters
    numberUnitsNumberOfCurrencyUnits ( 5), -- optional parameters
    typeOfChargingInfoSubTotal ( 6), -- optional parameters
    typeOfChargingInfoTotal ( 7),  -- optional parameters
    chargingMultiplierATHousandth ( 8), -- optional parameters
    chargingMultiplierAHundredth ( 9), -- optional parameters
    chargingMultiplierATenth (10),  -- optional parameters
    chargingMultiplierOne (11),  -- optional parameters

```

chargingMultiplierTen (12), -- optional parameters  
chargingMultiplierHundred (13), -- optional parameters  
chargingMultiplierThousand (14), -- optional parameters  
servicesPermitted (15), -- optional parameters  
mediaCallCharacteristics (16), -- optional parameters  
callCharacteristics (17), -- optional parameters  
droppedConnectionInfo (18), -- optional parameters  
callLinkageData (20), -- optional parameters  
privateData (19)} -- optional parameters

Delivered ::= BIT STRING  
{  
originatingNIDConnection ( 0), -- optional parameters  
userData ( 1), -- optional parameters  
servicesPermitted ( 2), -- optional parameters  
netwCallingDevice ( 3), -- optional parameters  
netwCalledDevice ( 4), -- optional parameters  
mediaCallCharacteristics ( 5), -- optional parameters  
callCharacteristics ( 6), -- optional parameters  
connectionInfo ( 7), -- optional parameters  
privateData ( 8)} -- optional parameters

DigitsDialed ::= BIT STRING  
{  
servicesPermitted ( 0), -- optional parameters  
netwCallingDevice ( 1), -- optional parameters  
netwCalledDevice ( 2), -- optional parameters  
diallingConnectionInfo ( 3), -- optional parameters  
callCharacteristics ( 4), -- optional parameters  
privateData ( 5)} -- optional parameters

Diverted ::= BIT STRING  
{  
callingDevice ( 0), -- optional parameters  
calledDevice ( 1), -- optional parameters  
userData ( 2), -- optional parameters  
servicesPermitted ( 3), -- optional parameters  
mediaCallCharacteristics ( 4), -- optional parameters  
callCharacteristics ( 5), -- optional parameters  
connectionInfo ( 6), -- optional parameters  
netwCallingDevice ( 7), -- optional parameters  
netwCalledDevice ( 8), -- optional parameters  
privateData ( 9), -- optional parameters  
sendsDivertedToAll (10)} -- optional parameters

Established ::= BIT STRING  
{  
originatingNIDConnection ( 0), -- optional parameters  
userData ( 1), -- optional parameters  
servicesPermitted ( 2), -- optional parameters  
netwCallingDevice ( 3), -- optional parameters  
netwCalledDevice ( 4), -- optional parameters  
mediaCallCharacteristics ( 5), -- optional parameters  
callCharacteristics ( 6), -- optional parameters  
establishedConnectionInfo ( 7), -- optional parameters  
privateData ( 8)} -- optional parameters

Failed ::= BIT STRING  
{  
originatingNIDConnection ( 0), -- optional parameters  
userData ( 1), -- optional parameters  
servicesPermitted ( 2), -- optional parameters  
netwCallingDevice ( 3), -- optional parameters  
netwCalledDevice ( 4), -- optional parameters  
mediaCallCharacteristics ( 5), -- optional parameters  
callCharacteristics ( 6), -- optional parameters  
failedConnectionInfo ( 7), -- optional parameters  
privateData ( 8), -- optional parameters  
callIDOnly ( 9)} -- miscellaneous characteristics

Held ::= BIT STRING  
{  
correlatorData ( 0), -- optional parameters  
servicesPermitted ( 1), -- optional parameters

```

mediaCallCharacteristics      ( 2),  -- optional parameters
callCharacteristics           ( 3),  -- optional parameters
heldConnectionInfo           ( 4),  -- optional parameters
callLinkageData               ( 6),  -- optional parameters
privateData                   ( 5)}  -- optional parameters

NetwCapsChanged ::= BIT STRING
{
  progressLocationUser        ( 0),  -- optional parameters
  progressLocationPrivateServLocal ( 1),  -- optional parameters
  progressLocationPublicServLocal ( 2),  -- optional parameters
  progressLocationTransit     ( 3),  -- optional parameters
  progressLocationPublicServRemote ( 4),  -- optional parameters
  progressLocationPrivateServRemote ( 5),  -- optional parameters
  progressLocationLocal       ( 6),  -- optional parameters
  progressLocationInternational ( 7),  -- optional parameters
  progressLocationNetwBeyondInterworking ( 8),  -- optional parameters
  progressLocationOther       ( 9),  -- optional parameters
  progressDescriptionISDN     (10),  -- optional parameters
  progressDescriptionQSIG     (11),  -- optional parameters
  progressDescriptionOther    (12),  -- optional parameters
  userData                    (13),  -- optional parameters
  typeOfNetworkISDNPublic     (14),  -- optional parameters
  typeOfNetworkNonISDNPublic  (15),  -- optional parameters
  typeOfNetworkISDNPrivate    (16),  -- optional parameters
  typeOfNetworkNonISDNPrivate (17),  -- optional parameters
  typeOfNetworkOther          (18),  -- optional parameters
  eventsProvidedParameter     (19),  -- optional parameters
  eventsProvidedBridged       (20),  -- optional parameters
  eventsProvidedCallCleared   (21),  -- optional parameters
  eventsProvidedConferenced   (22),  -- optional parameters
  eventsProvidedConnectionCleared (23),  -- optional parameters
  eventsProvidedDelivered     (24),  -- optional parameters
  eventsProvidedDigitsDialed  (25),  -- optional parameters
  eventsProvidedDiverted      (26),  -- optional parameters
  eventsProvidedEstablished   (27),  -- optional parameters
  eventsProvidedFailed        (28),  -- optional parameters
  eventsProvidedHeld          (29),  -- optional parameters
  eventsProvidedNetwCapsChanged (30),  -- optional parameters
  eventsProvidedNetwReached   (31),  -- optional parameters
  eventsProvidedOffered       (32),  -- optional parameters
  eventsProvidedOriginated    (33),  -- optional parameters
  eventsProvidedQueued        (34),  -- optional parameters
  eventsProvidedRetrieved     (35),  -- optional parameters
  eventsProvidedServiceInitiated (36),  -- optional parameters
  eventsProvidedTransferred   (37),  -- optional parameters
  servicesPermitted           (38),  -- optional parameters
  mediaCallCharacteristics    (39),  -- optional parameters
  callCharacteristics         (40),  -- optional parameters
  outboundConnectionInfo     (41),  -- optional parameters
  privateData                 (42)}  -- optional parameters

NetwReached ::= BIT STRING
{
  originatingNIDConnection    ( 0),  -- optional parameters
  userData                    ( 1),  -- optional parameters
  typeOfNetworkISDNPublic     ( 2),  -- optional parameters
  typeOfNetworkNonISDNPublic  ( 3),  -- optional parameters
  typeOfNetworkISDNPrivate    ( 4),  -- optional parameters
  typeOfNetworkNonISDNPrivate ( 5),  -- optional parameters
  typeOfNetworkOther          ( 6),  -- optional parameters
  eventsProvidedParameter     ( 7),  -- optional parameters
  eventsProvidedBridged       ( 8),  -- optional parameters
  eventsProvidedCallCleared   ( 9),  -- optional parameters
  eventsProvidedConferenced   (10),  -- optional parameters
  eventsProvidedConnectionCleared (11),  -- optional parameters
  eventsProvidedDelivered     (12),  -- optional parameters
  eventsProvidedDigitsDialed  (13),  -- optional parameters
  eventsProvidedDiverted      (14),  -- optional parameters
  eventsProvidedEstablished   (15),  -- optional parameters

```

```

eventsProvidedFailed          (16),    -- optional parameters
eventsProvidedHeld            (17),    -- optional parameters
eventsProvidedNetwCapsChanged (18),    -- optional parameters
eventsProvidedNetwReached     (19),    -- optional parameters
eventsProvidedOffered         (20),    -- optional parameters
eventsProvidedOriginated      (21),    -- optional parameters
eventsProvidedQueued          (22),    -- optional parameters
eventsProvidedRetrieved       (23),    -- optional parameters
eventsProvidedServiceInitiated (24),    -- optional parameters
eventsProvidedTransferred     (25),    -- optional parameters
servicesPermitted             (26),    -- optional parameters
mediaCallCharacteristics      (27),    -- optional parameters
callCharacteristics           (28),    -- optional parameters
outboundConnectionInfo        (29),    -- optional parameters
netwCallingDevice             (30),    -- optional parameters
netwCalledDevice              (31),    -- optional parameters
privateData                   (32)}    -- optional parameters

Offered ::= BIT STRING
{
  originatingNIDConnection    ( 0),    -- optional parameters
  userData                    ( 1),    -- optional parameters
  servicesPermitted            ( 2),    -- optional parameters
  netwCallingDevice            ( 3),    -- optional parameters
  netwCalledDevice            ( 4),    -- optional parameters
  mediaCallCharacteristics     ( 5),    -- optional parameters
  callCharacteristics          ( 6),    -- optional parameters
  offeredConnectionInfo        ( 7),    -- optional parameters
  privateData                  ( 8)}    -- optional parameters

Originated ::= BIT STRING
{
  originatingDevice           ( 0),    -- optional parameters
  servicesPermitted           ( 1),    -- optional parameters
  netwCallingDevice           ( 2),    -- optional parameters
  netwCalledDevice            ( 3),    -- optional parameters
  mediaCallCharacteristics     ( 4),    -- optional parameters
  callCharacteristics          ( 5),    -- optional parameters
  originatedConnectionInfo     ( 6),    -- optional parameters
  privateData                  ( 7)}    -- optional parameters

Queued ::= BIT STRING
{
  numberQueued                ( 0),    -- optional parameters
  callsInFront                 ( 1),    -- optional parameters
  userData                     ( 2),    -- optional parameters
  servicesPermitted            ( 3),    -- optional parameters
  netwCallingDevice            ( 4),    -- optional parameters
  netwCalledDevice            ( 5),    -- optional parameters
  mediaCallCharacteristics     ( 6),    -- optional parameters
  callCharacteristics          ( 7),    -- optional parameters
  queuedConnectionInfo         ( 8),    -- optional parameters
  privateData                  ( 9)}    -- optional parameters

Retrieved ::= BIT STRING
{
  correlatorData              ( 0),    -- optional parameters
  servicesPermitted           ( 1),    -- optional parameters
  mediaCallCharacteristics     ( 2),    -- optional parameters
  callCharacteristics          ( 3),    -- optional parameters
  retrievedConnectionInfo      ( 4),    -- optional parameters
  callLinkageData             ( 6),    -- optional parameters
  privateData                  ( 5)}    -- optional parameters

ServiceInitiated ::= BIT STRING
{
  servicesPermitted            ( 0),    -- optional parameters
  mediaCallCharacteristics     ( 1),    -- optional parameters
  callCharacteristics          ( 2),    -- optional parameters
  initiatedConnectionInfo      ( 3),    -- optional parameters
  netwCallingDevice            ( 4),    -- optional parameters
  netwCalledDevice            ( 5),    -- optional parameters
  privateData                  ( 6)}    -- optional parameters

```

```

Transferred ::= BIT STRING
{
    transferredConnectionsEndpointDeviceID      ( 0), -- optional parameters
    transferredConnectionsResultingConnectionInfo ( 1), -- optional parameters
    userData                                    ( 2), -- optional parameters
    chargingInfo                                ( 3), -- optional parameters
    numberUnitsNumberOfChargingUnits           ( 4), -- optional parameters
    numberUnitsTypeOfUnits                     ( 5), -- optional parameters
    numberUnitsNumberOfCurrencyUnits           ( 6), -- optional parameters
    typeOfChargingInfoSubTotal                 ( 7), -- optional parameters
    typeOfChargingInfoTotal                   ( 8), -- optional parameters
    chargingMultiplierAThousandth             ( 9), -- optional parameters
    chargingMultiplierAHundredth              (10), -- optional parameters
    chargingMultiplierATenth                  (11), -- optional parameters
    chargingMultiplierOne                      (12), -- optional parameters
    chargingMultiplierTen                     (13), -- optional parameters
    chargingMultiplierHundred                 (14), -- optional parameters
    chargingMultiplierThousand                (15), -- optional parameters
    servicesPermitted                          (16), -- optional parameters
    mediaCallCharacteristics                   (17), -- optional parameters
    callCharacteristics                        (18), -- optional parameters
    connectionInfo                            (19), -- optional parameters
    privateData                                (20)} -- optional parameters

CallAssociatedServList ::= SEQUENCE
{
    associateData          [0] IMPLICIT AssociateData          OPTIONAL,
    cancelTelephonyTones  [1] IMPLICIT CancelTelephonyTones   OPTIONAL,
    generateDigits         [2] IMPLICIT GenerateDigits          OPTIONAL,
    generateTelephonyTones [3] IMPLICIT GenerateTelephonyTones OPTIONAL,
    sendUserInformation    [4] IMPLICIT SendUserInformation     OPTIONAL}

AssociateData ::= BIT STRING
{
    accountCode      ( 0), -- optional parameters
    authCode         ( 1), -- optional parameters
    correlatorData   ( 2), -- optional parameters
    callQualifyingData ( 3), -- optional parameters
    privateData      ( 4), -- optional parameters
    privateDataInAck ( 5), -- optional parameters
    deviceIDOnly     ( 6), -- misc characteristics
    rejectsRequestsWithholdConnectionID ( 7), -- misc characteristics
    ackModelMultiStep ( 8)} -- misc characteristics

CancelTelephonyTones ::= BIT STRING
{
    privateData      ( 0), -- optional parameters
    privateDataInAck ( 1), -- optional parameters
    ackModelMultiStep ( 2)} -- misc characteristics

GeneratedDigits ::= BIT STRING
{
    digitModeDTMF      ( 0), -- optional parameters
    digitModePulse     ( 1), -- optional parameters
    toneDuration       ( 2), -- optional parameters
    pulseRate          ( 3), -- optional parameters
    pauseDuration      ( 4), -- optional parameters
    privateData        ( 5), -- optional parameters
    privateDataInAck   ( 6), -- optional parameters
    deviceIDOnly       ( 7), -- misc characteristics
    supportsDTMFTonesABCD ( 8), -- misc characteristics
    supportsPauseToneChar ( 9), -- misc characteristics
    ackModelMultiStep  (10)} -- misc characteristics

GenerateTelephonyTones ::= BIT STRING
{
    toneToSendBeep      ( 0), -- optional parameters
    toneToSendBilling   ( 1), -- optional parameters
    toneToSendBusy      ( 2), -- optional parameters
    toneToSendCarrier   ( 3), -- optional parameters
    toneToSendConfirmation ( 4), -- optional parameters

```

toneToSendDial	( 5 ),	-- optional parameters
toneToSendFaxCNG	( 6 ),	-- optional parameters
toneToSendHold	( 7 ),	-- optional parameters
toneToSendHowler	( 8 ),	-- optional parameters
toneToSendIntrusion	( 9 ),	-- optional parameters
toneToSendModemCNG	(10),	-- optional parameters
toneToSendPark	(11),	-- optional parameters
toneToSendRecordWarning	(12),	-- optional parameters
toneToSendReorder	(13),	-- optional parameters
toneToSendRingback	(14),	-- optional parameters
toneToSendSilence	(15),	-- optional parameters
toneToSendSitVC	(16),	-- optional parameters
toneToSendSitIC	(17),	-- optional parameters
toneToSendSitRO	(18),	-- optional parameters
toneToSendSitNC	(19),	-- optional parameters
toneToSendSf0	(20),	-- optional parameters
toneToSendSf1	(21),	-- optional parameters
toneToSendSf2	(22),	-- optional parameters
toneToSendSf3	(23),	-- optional parameters
toneToSendSf4	(24),	-- optional parameters
toneToSendSf5	(25),	-- optional parameters
toneToSendSf6	(26),	-- optional parameters
toneToSendSf7	(27),	-- optional parameters
toneToSendSf8	(28),	-- optional parameters
toneToSendSf9	(29),	-- optional parameters
toneToSendSf10	(30),	-- optional parameters
toneToSendSf11	(31),	-- optional parameters
toneToSendSf12	(32),	-- optional parameters
toneToSendSf13	(33),	-- optional parameters
toneToSendSf14	(34),	-- optional parameters
toneToSendSf15	(35),	-- optional parameters
toneToSendSf16	(36),	-- optional parameters
toneToSendSf17	(37),	-- optional parameters
toneToSendSf18	(38),	-- optional parameters
toneToSendSf19	(39),	-- optional parameters
toneToSendSf20	(40),	-- optional parameters
toneToSendSf21	(41),	-- optional parameters
toneToSendSf22	(42),	-- optional parameters
toneToSendSf23	(43),	-- optional parameters
toneToSendSf24	(44),	-- optional parameters
toneToSendSf25	(45),	-- optional parameters
toneToSendSf26	(46),	-- optional parameters
toneToSendSf27	(47),	-- optional parameters
toneToSendSf28	(48),	-- optional parameters
toneToSendSf29	(49),	-- optional parameters
toneToSendSf30	(50),	-- optional parameters
toneToSendSf31	(51),	-- optional parameters
toneToSendSf32	(52),	-- optional parameters
toneToSendSf33	(53),	-- optional parameters
toneToSendSf34	(54),	-- optional parameters
toneToSendSf35	(55),	-- optional parameters
toneToSendSf36	(56),	-- optional parameters
toneToSendSf37	(57),	-- optional parameters
toneToSendSf38	(58),	-- optional parameters
toneToSendSf39	(59),	-- optional parameters
toneToSendSf40	(60),	-- optional parameters
toneToSendSf41	(61),	-- optional parameters
toneToSendSf42	(62),	-- optional parameters
toneToSendSf43	(63),	-- optional parameters
toneToSendSf44	(64),	-- optional parameters
toneToSendSf45	(65),	-- optional parameters
toneToSendSf46	(66),	-- optional parameters
toneToSendSf47	(67),	-- optional parameters
toneToSendSf48	(68),	-- optional parameters
toneToSendSf49	(69),	-- optional parameters
toneToSendSf50	(70),	-- optional parameters
toneToSendSf51	(71),	-- optional parameters
toneToSendSf52	(72),	-- optional parameters

toneToSendSf53 (73), -- optional parameters  
toneToSendSf54 (74), -- optional parameters  
toneToSendSf55 (75), -- optional parameters  
toneToSendSf56 (76), -- optional parameters  
toneToSendSf57 (77), -- optional parameters  
toneToSendSf58 (78), -- optional parameters  
toneToSendSf59 (79), -- optional parameters  
toneToSendSf60 (80), -- optional parameters  
toneToSendSf61 (81), -- optional parameters  
toneToSendSf62 (82), -- optional parameters  
toneToSendSf63 (83), -- optional parameters  
toneToSendSf64 (84), -- optional parameters  
toneToSendSf65 (85), -- optional parameters  
toneToSendSf66 (86), -- optional parameters  
toneToSendSf67 (87), -- optional parameters  
toneToSendSf68 (88), -- optional parameters  
toneToSendSf69 (89), -- optional parameters  
toneToSendSf70 (90), -- optional parameters  
toneToSendSf71 (91), -- optional parameters  
toneToSendSf72 (92), -- optional parameters  
toneToSendSf73 (93), -- optional parameters  
toneToSendSf74 (94), -- optional parameters  
toneToSendSf75 (95), -- optional parameters  
toneToSendSf76 (96), -- optional parameters  
toneToSendSf77 (97), -- optional parameters  
toneToSendSf78 (98), -- optional parameters  
toneToSendSf79 (99), -- optional parameters  
toneToSendSf80 (100), -- optional parameters  
toneToSendSf81 (101), -- optional parameters  
toneToSendSf82 (102), -- optional parameters  
toneToSendSf83 (103), -- optional parameters  
toneToSendSf84 (104), -- optional parameters  
toneToSendSf85 (105), -- optional parameters  
toneToSendSf86 (106), -- optional parameters  
toneToSendSf87 (107), -- optional parameters  
toneToSendSf88 (108), -- optional parameters  
toneToSendSf89 (109), -- optional parameters  
toneToSendSf90 (110), -- optional parameters  
toneToSendSf91 (111), -- optional parameters  
toneToSendSf92 (112), -- optional parameters  
toneToSendSf93 (113), -- optional parameters  
toneToSendSf94 (114), -- optional parameters  
toneToSendSf95 (115), -- optional parameters  
toneToSendSf96 (116), -- optional parameters  
toneToSendSf97 (117), -- optional parameters  
toneToSendSf98 (118), -- optional parameters  
toneToSendSf99 (119), -- optional parameters  
toneToSendSf100 (120), -- optional parameters  
toneDuration (121), -- optional parameters  
privateData (122), -- optional parameters  
privateDataInAck (123), -- optional parameters  
deviceIDOnly (124), -- misc characteristics  
ackModelMultiStep (125)} -- misc characteristics

SendUserInformation ::= BIT STRING  
{ privateData ( 0), -- optional parameters  
privateDataInAck ( 1), -- optional parameters  
deviceIDOnly ( 2), -- misc characteristics  
ackModelMultiStep ( 3)} -- misc characteristics

CallAssociatedEvtsList ::= SEQUENCE  
{ callInformation [0] IMPLICIT CallInformation OPTIONAL,  
charging [1] IMPLICIT Charging OPTIONAL,  
digitsGenerated [2] IMPLICIT DigitsGenerated OPTIONAL,  
telephonyTonesGenerated [3] IMPLICIT TelephonyTonesGenerated OPTIONAL,  
serviceCompletionFailure [4] IMPLICIT ServiceCompletionFailure OPTIONAL}

CallInformation ::= BIT STRING

```

{
    callingDevice                ( 0), -- optional parameters
    accountInfo                  ( 1), -- optional parameters
    authorisatinonCode           ( 2), -- optional parameters
    correlatorData               ( 3), -- optional parameters
    servicesPermitted            ( 4), -- optional parameters
    userData                     ( 5), -- optional parameters
    callQualifyingData          ( 6), -- optional parameters
    connectionInfo              ( 7), -- optional parameters
    callLinkageData              (10), -- optional parameters
    privateData                  ( 8), -- optional parameters
    genCallInfoForOutdatedConnID ( 9)} -- misc characteristics

Charging ::= BIT STRING
{
    numberUnitsNumberOfChargingUnits ( 0), -- optional parameters
    numberUnitsTypeOfUnits            ( 1), -- optional parameters
    numberUnitsNumberOfCurrencyUnits ( 2), -- optional parameters
    typeOfChargingInfoSubTotal        ( 3), -- optional parameters
    typeOfChargingInfoTotal           ( 4), -- optional parameters
    chargingMultiplierATHousandth     ( 5), -- optional parameters
    chargingMultiplierAHundredth      ( 6), -- optional parameters
    chargingMultiplierATenth           ( 7), -- optional parameters
    chargingMultiplierOne              ( 8), -- optional parameters
    chargingMultiplierTen              ( 9), -- optional parameters
    chargingMultiplierHundred         (10), -- optional parameters
    chargingMultiplierThousand        (11), -- optional parameters
    privateData                        (12)} -- optional parameters

DigitsGenerated ::= BIT STRING
{
    digitsDurationList            ( 0), -- optional parameters
    pauseDurationList             ( 1), -- optional parameters
    connectionInfo                ( 2), -- optional parameters
    privateData                    ( 3)} -- optional parameters

TelephonyTonesGenerated ::= BIT STRING
{
    toneToSendBeep                ( 0), -- optional parameters
    toneToSendBilling              ( 1), -- optional parameters
    toneToSendBusy                 ( 2), -- optional parameters
    toneToSendCarrier              ( 3), -- optional parameters
    toneToSendConfirmation         ( 4), -- optional parameters
    toneToSendDial                 ( 5), -- optional parameters
    toneToSendFaxCNG              ( 6), -- optional parameters
    toneToSendHold                 ( 7), -- optional parameters
    toneToSendHowler               ( 8), -- optional parameters
    toneToSendIntrusion            ( 9), -- optional parameters
    toneToSendModemCNG            (10), -- optional parameters
    toneToSendPark                 (11), -- optional parameters
    toneToSendRecordWarning        (12), -- optional parameters
    toneToSendReorder              (13), -- optional parameters
    toneToSendRingback             (14), -- optional parameters
    toneToSendSilence              (15), -- optional parameters
    toneToSendSitVC                (16), -- optional parameters
    toneToSendSitIC                (17), -- optional parameters
    toneToSendSitRO                (18), -- optional parameters
    toneToSendSitNC                (19), -- optional parameters
    toneToSendSf0                  (20), -- optional parameters
    toneToSendSf1                  (21), -- optional parameters
    toneToSendSf2                  (22), -- optional parameters
    toneToSendSf3                  (23), -- optional parameters
    toneToSendSf4                  (24), -- optional parameters
    toneToSendSf5                  (25), -- optional parameters
    toneToSendSf6                  (26), -- optional parameters
    toneToSendSf7                  (27), -- optional parameters
    toneToSendSf8                  (28), -- optional parameters
    toneToSendSf9                  (29), -- optional parameters
    toneToSendSf10                 (30), -- optional parameters
    toneToSendSf11                 (31), -- optional parameters
    toneToSendSf12                 (32), -- optional parameters
    toneToSendSf13                 (33), -- optional parameters

```

toneToSendSf14	(34),	-- optional parameters
toneToSendSf15	(35),	-- optional parameters
toneToSendSf16	(36),	-- optional parameters
toneToSendSf17	(37),	-- optional parameters
toneToSendSf18	(38),	-- optional parameters
toneToSendSf19	(39),	-- optional parameters
toneToSendSf20	(40),	-- optional parameters
toneToSendSf21	(41),	-- optional parameters
toneToSendSf22	(42),	-- optional parameters
toneToSendSf23	(43),	-- optional parameters
toneToSendSf24	(44),	-- optional parameters
toneToSendSf25	(45),	-- optional parameters
toneToSendSf26	(46),	-- optional parameters
toneToSendSf27	(47),	-- optional parameters
toneToSendSf28	(48),	-- optional parameters
toneToSendSf29	(49),	-- optional parameters
toneToSendSf30	(50),	-- optional parameters
toneToSendSf31	(51),	-- optional parameters
toneToSendSf32	(52),	-- optional parameters
toneToSendSf33	(53),	-- optional parameters
toneToSendSf34	(54),	-- optional parameters
toneToSendSf35	(55),	-- optional parameters
toneToSendSf36	(56),	-- optional parameters
toneToSendSf37	(57),	-- optional parameters
toneToSendSf38	(58),	-- optional parameters
toneToSendSf39	(59),	-- optional parameters
toneToSendSf40	(60),	-- optional parameters
toneToSendSf41	(61),	-- optional parameters
toneToSendSf42	(62),	-- optional parameters
toneToSendSf43	(63),	-- optional parameters
toneToSendSf44	(64),	-- optional parameters
toneToSendSf45	(65),	-- optional parameters
toneToSendSf46	(66),	-- optional parameters
toneToSendSf47	(67),	-- optional parameters
toneToSendSf48	(68),	-- optional parameters
toneToSendSf49	(69),	-- optional parameters
toneToSendSf50	(70),	-- optional parameters
toneToSendSf51	(71),	-- optional parameters
toneToSendSf52	(72),	-- optional parameters
toneToSendSf53	(73),	-- optional parameters
toneToSendSf54	(74),	-- optional parameters
toneToSendSf55	(75),	-- optional parameters
toneToSendSf56	(76),	-- optional parameters
toneToSendSf57	(77),	-- optional parameters
toneToSendSf58	(78),	-- optional parameters
toneToSendSf59	(79),	-- optional parameters
toneToSendSf60	(80),	-- optional parameters
toneToSendSf61	(81),	-- optional parameters
toneToSendSf62	(82),	-- optional parameters
toneToSendSf63	(83),	-- optional parameters
toneToSendSf64	(84),	-- optional parameters
toneToSendSf65	(85),	-- optional parameters
toneToSendSf66	(86),	-- optional parameters
toneToSendSf67	(87),	-- optional parameters
toneToSendSf68	(88),	-- optional parameters
toneToSendSf69	(89),	-- optional parameters
toneToSendSf70	(90),	-- optional parameters
toneToSendSf71	(91),	-- optional parameters
toneToSendSf72	(92),	-- optional parameters
toneToSendSf73	(93),	-- optional parameters
toneToSendSf74	(94),	-- optional parameters
toneToSendSf75	(95),	-- optional parameters
toneToSendSf76	(96),	-- optional parameters
toneToSendSf77	(97),	-- optional parameters
toneToSendSf78	(98),	-- optional parameters
toneToSendSf79	(99),	-- optional parameters
toneToSendSf80	(100),	-- optional parameters
toneToSendSf81	(101),	-- optional parameters

STANDARDSDOWNLOAD.COM : Click to view the full PDF of ISO/IEC 18052:2000

```

toneToSendSf82          (102), -- optional parameters
toneToSendSf83          (103), -- optional parameters
toneToSendSf84          (104), -- optional parameters
toneToSendSf85          (105), -- optional parameters
toneToSendSf86          (106), -- optional parameters
toneToSendSf87          (107), -- optional parameters
toneToSendSf88          (108), -- optional parameters
toneToSendSf89          (109), -- optional parameters
toneToSendSf90          (110), -- optional parameters
toneToSendSf91          (111), -- optional parameters
toneToSendSf92          (112), -- optional parameters
toneToSendSf93          (113), -- optional parameters
toneToSendSf94          (114), -- optional parameters
toneToSendSf95          (115), -- optional parameters
toneToSendSf96          (116), -- optional parameters
toneToSendSf97          (117), -- optional parameters
toneToSendSf98          (118), -- optional parameters
toneToSendSf99          (119), -- optional parameters
toneToSendSf100         (120), -- optional parameters
toneFrequency           (121), -- optional parameters
toneDuration            (122), -- optional parameters
pauseDurationList       (123), -- optional parameters
connectionInfo          (124), -- optional parameters
privateData             (125)} -- optional parameters

ServiceCompletionFailure ::= BIT STRING
{
    primaryCallConnectionInfo      ( 0), -- optional parameters
    secondaryCallConnectionInfo    ( 1), -- optional parameters
    otherDevicesPrimaryCallList    ( 2), -- optional parameters
    otherDevicesSecondaryCallList  ( 3), -- optional parameters
    mediaCallCharacteristics       ( 4), -- optional parameters
    privateData                    ( 5)} -- optional parameters

MediaServList ::= SEQUENCE
{
    attachMediaService      [0] IMPLICIT AttachMediaService      OPTIONAL,
    detachMediaService      [1] IMPLICIT DetachMediaService      OPTIONAL}

AttachMediaService ::= BIT STRING
{
    mediaServiceVersion          ( 0), -- optional parameters
    mediaServiceInstanceID       ( 1), -- optional parameters
    connectionModeConsultConference ( 2), -- optional parameters
    connectionModeConsultConferenceHold ( 3), -- optional parameters
    connectionModeDeflect        ( 4), -- optional parameters
    connectionModeDirectedPickup ( 5), -- optional parameters
    connectionModeJoin           ( 6), -- optional parameters
    connectionModeSingleStepConference ( 7), -- optional parameters
    connectionModeSingleStepConferenceHold ( 8), -- optional parameters
    connectionModeSingleStepTransfer ( 9), -- optional parameters
    connectionModeTransfer       (10), -- optional parameters
    connectionModeDirect         (11), -- optional parameters
    requestedConnectionState      (12), -- optional parameters
    privateData                   (13), -- optional parameters
    mediaServiceInstanceIDInAck   (14), -- optional parameters
    mediaConnectionInfoInAck      (15), -- optional parameters
    privateDataInAck              (16), -- optional parameters
    deviceIDOnly                  (17), -- misc characteristics
    ackModelMultiStep             (18)} -- misc characteristics

DetachMediaService ::= BIT STRING
{
    alerting          ( 0), -- initial states
    connected         ( 1), -- initial states
    fail              ( 2), -- initial states
    hold              ( 3), -- initial states
    queued            ( 4), -- initial states
    privateData       ( 5), -- optional parameters
    privateDataInAck ( 6), -- optional parameters
    deviceIDOnly      ( 7), -- misc characteristics
    ackModelMultiStep ( 8)} -- misc characteristics

```

```

MediaEvtsList ::= SEQUENCE
{
    mediaAttached  [0] IMPLICIT   MediaAttached  OPTIONAL,
    mediaDetached  [1] IMPLICIT   MediaDetached  OPTIONAL}

MediaAttached ::= BIT STRING
{
    mediaServiceVersion      ( 0),    -- optional parameters
    mediaServiceInstanceID   ( 1),    -- optional parameters
    mediaStreamID            ( 2),    -- optional parameters
    mediaCallCharacteristics ( 3),    -- optional parameters
    callCharacteristics      ( 4),    -- optional parameters
    mediaConnectionInfo     ( 5),    -- optional parameters
    privateData              ( 6)}    -- optional parameters

MediaDetached ::= BIT STRING
{
    mediaServiceVersion      ( 0),    -- optional parameters
    mediaServiceInstanceID   ( 1),    -- optional parameters
    mediaStreamID            ( 2),    -- optional parameters
    mediaCallCharacteristics ( 3),    -- optional parameters
    callCharacteristics      ( 4),    -- optional parameters
    mediaConnectionInfo     ( 5),    -- optional parameters
    privateData              ( 6)}    -- optional parameters

RoutingServList ::= SEQUENCE
{
    routeRegister      [ 0] IMPLICIT RouteRegister      OPTIONAL,
    routeRegisterCancel [ 1] IMPLICIT RouteRegisterCancel OPTIONAL,
    routeRegisterAbort [ 2] IMPLICIT RouteRegisterAbort  OPTIONAL,
    reRoute            [ 3] IMPLICIT ReRoute            OPTIONAL,
    routeEnd           [ 4] IMPLICIT RouteEnd           OPTIONAL,
    routeReject        [ 5] IMPLICIT RouteReject        OPTIONAL,
    routeRequest       [ 6] IMPLICIT RouteRequest       OPTIONAL,
    routeSelect        [ 7] IMPLICIT RouteSelect        OPTIONAL,
    routeUsed          [ 8] IMPLICIT RouteUsed          OPTIONAL}

RouteRegister ::= BIT STRING
{
    routingDevice          ( 0),    -- optional parameters
    requestedMonitorMediaClass ( 1), -- optional parameters
    requestedMonitorMediaClassAudio ( 2), -- optional parameters
    requestedMonitorMediaClassData ( 3), -- optional parameters
    requestedMonitorMediaClassImage ( 4), -- optional parameters
    requestedMonitorMediaClassVoice ( 5), -- optional parameters
    privateData            ( 6),    -- optional parameters
    actualRoutingMediaClassInAck ( 7), -- optional parameters
    privateDataInAck      ( 8),    -- optional parameters
    allRoutingDevices     ( 9)}    -- misc characteristics

RouteRegisterAbort ::= BIT STRING
{
    privateData          ( 0)}    -- optional parameters

RouteRegisterCancel ::= BIT STRING
{
    privateData          ( 0),    -- optional parameters
    privateDataInAck    ( 1)}    -- optional parameters

ReRoute ::= BIT STRING
{
    replyTimeout        ( 1),    -- optional parameters
    correlatorData     ( 2),    -- optional parameters
    privateData         ( 3)}    -- optional parameters

RouteEnd ::= BIT STRING
{
    errorValue          ( 1),    -- optional parameters
    correlatorData     ( 2),    -- optional parameters
    privateData         ( 3),    -- optional parameters
    supportsSending     ( 4),    -- misc characteristics
    supportsReceiving  ( 5)}    -- misc characteristics

RouteReject ::= BIT STRING
{
    rejectCauseBusyOverflow ( 0), -- optional parameters
    rejectCauseQueueTimeOverflow ( 1), -- optional parameters
}

```

```

rejectCauseCapacityOverflow      ( 2),  -- optional parameters
rejectCauseCalendarOverflow      ( 3),  -- optional parameters
rejectCauseUnknownOverflow      ( 4),  -- optional parameters
correlatorData                   ( 5),  -- optional parameters
privateData                       ( 6)}  -- optional parameters

RouteRequest ::= BIT STRING
{
    callingDevice                 ( 0),  -- optional parameters
    calledDevice                  ( 1),  -- optional parameters
    routeSelAlgorithmACD          ( 2),  -- optional parameters
    routeSelAlgorithmEmergency    ( 3),  -- optional parameters
    routeSelAlgorithmLeastCost    ( 4),  -- optional parameters
    routeSelAlgorithmNormal       ( 5),  -- optional parameters
    routeSelAlgorithmUserDefined  ( 6),  -- optional parameters
    priority                      ( 7),  -- optional parameters
    replyTimeout                  ( 8),  -- optional parameters
    correlatorData                ( 9),  -- optional parameters
    mediaCallCharacteristics      (10),  -- optional parameters
    callCharacteristics           (11),  -- optional parameters
    routedCallInfo                (12),  -- optional parameters
    privateData                   (13),  -- optional parameters
    nonCallRelatedRouteing       (14)}  -- misc characteristics

RouteSelect ::= BIT STRING
{
    alternateRoutes               ( 0),  -- optional parameters
    remainRetriesNoListAvailable  ( 1),  -- optional parameters
    remainRetriesNoCountAvailable ( 2),  -- optional parameters
    remainRetriesRetryCount       ( 3),  -- optional parameters
    routeUsed                     ( 4),  -- optional parameters
    correlatorData                ( 5),  -- optional parameters
    privateData                   ( 6)}  -- optional parameters

RouteUsed ::= BIT STRING
{
    callingDevice                 ( 0),  -- optional parameters
    domain                        ( 1),  -- optional parameters
    correlatorData                ( 2),  -- optional parameters
    privateData                   ( 3)}  -- optional parameters

PhysDevServList ::= SEQUENCE
{
    buttonPress                   [ 0] IMPLICIT ButtonPress           OPTIONAL,
    getAuditoryApparatusInfo      [ 1] IMPLICIT GetAuditoryApparatusInfo  OPTIONAL,
    getButtonInformation          [ 2] IMPLICIT GetButtonInformation  OPTIONAL,
    getDisplay                    [ 3] IMPLICIT GetDisplay            OPTIONAL,
    getHookSwitchStatus          [ 4] IMPLICIT GetHookSwitchStatus    OPTIONAL,
    getLampInfo                  [ 5] IMPLICIT GetLampInfo            OPTIONAL,
    getLampMode                  [ 6] IMPLICIT GetLampMode            OPTIONAL,
    getMessageWaitingIndicator    [ 7] IMPLICIT GetMessageWaitingIndicator  OPTIONAL,
    getMicrophoneGain            [ 8] IMPLICIT GetMicrophoneGain      OPTIONAL,
    getMicrophoneMute            [ 9] IMPLICIT GetMicrophoneMute      OPTIONAL,
    getRingerStatus              [10] IMPLICIT GetRingerStatus        OPTIONAL,
    getSpeakerMute               [11] IMPLICIT GetSpeakerMute        OPTIONAL,
    getSpeakerVolume             [12] IMPLICIT GetSpeakerVolume      OPTIONAL,
    setButtonInformation          [13] IMPLICIT SetButtonInformation  OPTIONAL,
    setDisplay                    [14] IMPLICIT SetDisplay            OPTIONAL,
    setHookSwitchStatus          [15] IMPLICIT SetHookSwitchStatus    OPTIONAL,
    setLampMode                  [16] IMPLICIT SetLampMode            OPTIONAL,
    setMessageWaitingIndicator    [17] IMPLICIT SetMessageWaitingIndicator  OPTIONAL,
    setMicrophoneGain            [18] IMPLICIT SetMicrophoneGain      OPTIONAL,
    setMicrophoneMute            [19] IMPLICIT SetMicrophoneMute      OPTIONAL,
    setRingerStatus              [20] IMPLICIT SetRingerStatus        OPTIONAL,
    setSpeakerMute               [21] IMPLICIT SetSpeakerMute        OPTIONAL,
    setSpeakerVolume             [22] IMPLICIT SetSpeakerVolume      OPTIONAL}

ButtonPress ::= BIT STRING
{
    privateData                   ( 0),  -- optional parameters
    privateDataInAck              ( 1),  -- optional parameters
    ackModelMultiStep             ( 2)}  -- misc characteristics

```

```

GetAuditoryApparatusInfo ::= BIT STRING
{
    auditoryApparatus          ( 0), -- optional parameters
    privateData                ( 1), -- optional parameters
    auditoryApparatusTypeSpeakerphone ( 2), -- optional parameters
    auditoryApparatusTypeHandset    ( 3), -- optional parameters
    auditoryApparatusTypeHeadset    ( 4), -- optional parameters
    auditoryApparatusTypeSpeakerOnlyPhone ( 5), -- optional parameters
    auditoryApparatusTypeoother     ( 6), -- optional parameters
    speakerPresent              ( 7), -- optional parameters
    speakerVolumeSettable        ( 8), -- optional parameters
    speakerVolumeReadable        ( 9), -- optional parameters
    speakerMuteSettable          (10), -- optional parameters
    speakerMuteReadable          (11), -- optional parameters
    microphonePresent            (12), -- optional parameters
    microphoneGainSettable       (13), -- optional parameters
    microphoneGainReadable       (14), -- optional parameters
    microphoneMuteSettable       (15), -- optional parameters
    microphoneMuteReadable       (16), -- optional parameters
    hookswitchSettable           (17), -- optional parameters
    hookswitchOnHook             (18), -- optional parameters
    privateDataInAck             (19)} -- optional parameters

GetButtonInformation ::= BIT STRING
{
    button                      ( 0), -- optional parameters
    privateData                 ( 1), -- optional parameters
    buttonLabelInAck            ( 2), -- optional parameters
    buttonLabelSettableInAck    ( 3), -- optional parameters
    buttonFunctionInAck         ( 4), -- optional parameters
    buttonAssociatedNumberInAck ( 5), -- optional parameters
    buttonAssociatedNumberSettableInAck ( 6), -- optional parameters
    listOfLampsInAck           ( 7), -- optional parameters
    privateDataInAck            ( 8)} -- optional parameters

GetDisplay ::= BIT STRING
{
    displayID                   ( 0), -- optional parameters
    privateData                 ( 1), -- optional parameters
    characterSetASCII           ( 2), -- optional parameters
    characterSetUnicode         ( 3), -- optional parameters
    characterSetProprietary     ( 4), -- optional parameters
    privateDataInAck            ( 5)} -- optional parameters

GetHookSwitchStatus ::= BIT STRING
{
    hookSwitch                  ( 0), -- optional parameters
    privateData                 ( 1), -- optional parameters
    privateDataInAck            ( 2)} -- optional parameters

GetLampInfo ::= BIT STRING
{
    lamp                        ( 0), -- optional parameters
    privateData                 ( 1), -- optional parameters
    lampLabelInAck              ( 2), -- optional parameters
    buttonInAck                  ( 3), -- optional parameters
    lampColorInAck              ( 4), -- optional parameters
    privateDataInAck            ( 5)} -- optional parameters

GetLampMode ::= BIT STRING
{
    lamp                        ( 0), -- optional parameters
    privateData                 ( 1), -- optional parameters
    lampModeInAck               ( 2), -- optional parameters
    lampBrightnessNormal        ( 3), -- optional parameters
    lampBrightnessDim           ( 4), -- optional parameters
    lampBrightnessBright        ( 5), -- optional parameters
    lampColorInAck              ( 6), -- optional parameters
    buttonInAck                  ( 7), -- optional parameters
    privateDataInAck            ( 8)} -- optional parameters

GetMessageWaitingIndicator ::= BIT STRING
{
    privateData                 ( 0), -- optional parameter
    deviceForMsgInAck           ( 1), -- optional parameter

```

```

        lampIsPresentInAck          ( 2),  -- optional parameter
        privateDataInAck          ( 3)}  -- optional parameters

GetMicrophoneGain ::= BIT STRING
{
    auditoryApparatus          ( 0),  -- optional parameters
    privateData                ( 1),  -- optional parameters
    micGainAbsInAck           ( 2),  -- optional parameters
    privateDataInAck          ( 3)}  -- optional parameters

GetMicrophoneMute ::= BIT STRING
{
    auditoryApparatus          ( 0),  -- optional parameters
    privateData                ( 1),  -- optional parameters
    privateDataInAck          ( 2)}  -- optional parameters

GetRingerStatus ::= BIT STRING
{
    ringer                     ( 0),  -- optional parameters
    privateData                ( 1),  -- optional parameters
    ringCountInAck            ( 2),  -- optional parameters
    ringPatternInAck          ( 3),  -- optional parameters
    ringVolumeInAck           ( 4),  -- optional parameters
    ringVolumeAbsInAck        ( 5),  -- optional parameters
    privateDataInAck          ( 6)}  -- optional parameters

GetSpeakerMute ::= BIT STRING
{
    auditoryApparatus          ( 0),  -- optional parameters
    privateDataInAck          ( 1),  -- optional parameters
    privateData                ( 2)}  -- optional parameters

GetSpeakerVolume ::= BIT STRING
{
    auditoryApparatus          ( 0),  -- optional parameters
    privateData                ( 1),  -- optional parameters
    speakerVolAbsInAck        ( 2),  -- optional parameters
    privateDataInAck          ( 3)}  -- optional parameters

SetButtonInformation ::= BIT STRING
{
    buttonLabel                ( 0),  -- optional parameters
    buttonAssociatedNumber      ( 1),  -- optional parameters
    privateData                ( 2),  -- optional parameters
    privateDataInAck          ( 3),  -- optional parameters
    ackModelMultiStep          ( 4)}  -- misc characteristics

SetDisplay ::= BIT STRING
{
    physBaseRowNumber          ( 0),  -- optional parameters
    physColumnRowNumber        ( 1),  -- optional parameters
    offset                     ( 2),  -- optional parameters
    privateData                ( 3),  -- optional parameters
    privateDataInAck          ( 4),  -- optional parameters
    supportsModifyingPosition ( 5),  -- misc characteristics
    ackModelMultiStep          ( 6)}  -- misc characteristics

SetHookSwitchStatus ::= BIT STRING
{
    privateData                ( 0),  -- optional parameters
    ackModelMultiStep          ( 1)}  -- misc characteristics

SetLampMode ::= BIT STRING
{
    lampModeBrokenFlutter      ( 0),  -- optional parameter
    lampModeFlutter            ( 1),  -- optional parameter
    lampModeOff                ( 2),  -- optional parameter
    lampModeSteady             ( 3),  -- optional parameter
    lampModeWink               ( 4),  -- optional parameter
    lampModeReserved           ( 5),  -- optional parameter
    lampModeSf0                ( 6),  -- optional parameters
    lampModeSf1                ( 7),  -- optional parameters
    lampModeSf2                ( 8),  -- optional parameters
    lampModeSf3                ( 9),  -- optional parameters
    lampModeSf4                (10),  -- optional parameters
    lampModeSf5                (11),  -- optional parameters
    lampModeSf6                (12),  -- optional parameters

```

lampModeSf7	(13),	-- optional parameters
lampModeSf8	(14),	-- optional parameters
lampModeSf9	(15),	-- optional parameters
lampModeSf10	(16),	-- optional parameters
lampModeSf11	(17),	-- optional parameters
lampModeSf12	(18),	-- optional parameters
lampModeSf13	(19),	-- optional parameters
lampModeSf14	(20),	-- optional parameters
lampModeSf15	(21),	-- optional parameters
lampModeSf16	(22),	-- optional parameters
lampModeSf17	(23),	-- optional parameters
lampModeSf18	(24),	-- optional parameters
lampModeSf19	(25),	-- optional parameters
lampModeSf20	(26),	-- optional parameters
lampModeSf21	(27),	-- optional parameters
lampModeSf22	(28),	-- optional parameters
lampModeSf23	(29),	-- optional parameters
lampModeSf24	(30),	-- optional parameters
lampModeSf25	(31),	-- optional parameters
lampModeSf26	(32),	-- optional parameters
lampModeSf27	(33),	-- optional parameters
lampModeSf28	(34),	-- optional parameters
lampModeSf29	(35),	-- optional parameters
lampModeSf30	(36),	-- optional parameters
lampModeSf31	(37),	-- optional parameters
lampModeSf32	(38),	-- optional parameters
lampModeSf33	(39),	-- optional parameters
lampModeSf34	(40),	-- optional parameters
lampModeSf35	(41),	-- optional parameters
lampModeSf36	(42),	-- optional parameters
lampModeSf37	(43),	-- optional parameters
lampModeSf38	(44),	-- optional parameters
lampModeSf39	(45),	-- optional parameters
lampModeSf40	(46),	-- optional parameters
lampModeSf41	(47),	-- optional parameters
lampModeSf42	(48),	-- optional parameters
lampModeSf43	(49),	-- optional parameters
lampModeSf44	(50),	-- optional parameters
lampModeSf45	(51),	-- optional parameters
lampModeSf46	(52),	-- optional parameters
lampModeSf47	(53),	-- optional parameters
lampModeSf48	(54),	-- optional parameters
lampModeSf49	(55),	-- optional parameters
lampModeSf50	(56),	-- optional parameters
lampModeSf51	(57),	-- optional parameters
lampModeSf52	(58),	-- optional parameters
lampModeSf53	(59),	-- optional parameters
lampModeSf54	(60),	-- optional parameters
lampModeSf55	(61),	-- optional parameters
lampModeSf56	(62),	-- optional parameters
lampModeSf57	(63),	-- optional parameters
lampModeSf58	(64),	-- optional parameters
lampModeSf59	(65),	-- optional parameters
lampModeSf60	(66),	-- optional parameters
lampModeSf61	(67),	-- optional parameters
lampModeSf62	(68),	-- optional parameters
lampModeSf63	(69),	-- optional parameters
lampModeSf64	(70),	-- optional parameters
lampModeSf65	(71),	-- optional parameters
lampModeSf66	(72),	-- optional parameters
lampModeSf67	(73),	-- optional parameters
lampModeSf68	(74),	-- optional parameters
lampModeSf69	(75),	-- optional parameters
lampModeSf70	(76),	-- optional parameters
lampModeSf71	(77),	-- optional parameters
lampModeSf72	(78),	-- optional parameters
lampModeSf73	(79),	-- optional parameters
lampModeSf74	(80),	-- optional parameters

STANDARD 350.COM : Click to view the FULL PDF of ISO/IEC 18052:2000

lampModeSf75	(81),	-- optional parameters
lampModeSf76	(82),	-- optional parameters
lampModeSf77	(83),	-- optional parameters
lampModeSf78	(84),	-- optional parameters
lampModeSf79	(85),	-- optional parameters
lampModeSf80	(86),	-- optional parameters
lampModeSf81	(87),	-- optional parameters
lampModeSf82	(88),	-- optional parameters
lampModeSf83	(89),	-- optional parameters
lampModeSf84	(90),	-- optional parameters
lampModeSf85	(91),	-- optional parameters
lampModeSf86	(92),	-- optional parameters
lampModeSf87	(93),	-- optional parameters
lampModeSf88	(94),	-- optional parameters
lampModeSf89	(95),	-- optional parameters
lampModeSf90	(96),	-- optional parameters
lampModeSf91	(97),	-- optional parameters
lampModeSf92	(98),	-- optional parameters
lampModeSf93	(99),	-- optional parameters
lampModeSf94	(100),	-- optional parameter
lampBrightnessNormal	(101),	-- optional parameter
lampBrightnessDim	(102),	-- optional parameter
lampBrightnessBright	(103),	-- optional parameter
lampColorNoColor	(104),	-- optional parameter
lampColorRed	(105),	-- optional parameter
lampColorYellow	(106),	-- optional parameter
lampColorGreen	(107),	-- optional parameter
lampColorBlue	(108),	-- optional parameter
lampColorReserved	(109),	-- optional parameter
lampColorSf0	(110),	-- optional parameters
lampColorSf1	(111),	-- optional parameters
lampColorSf2	(112),	-- optional parameters
lampColorSf3	(113),	-- optional parameters
lampColorSf4	(114),	-- optional parameters
lampColorSf5	(115),	-- optional parameters
lampColorSf6	(116),	-- optional parameters
lampColorSf7	(117),	-- optional parameters
lampColorSf8	(118),	-- optional parameters
lampColorSf9	(119),	-- optional parameters
lampColorSf10	(120),	-- optional parameters
lampColorSf11	(121),	-- optional parameters
lampColorSf12	(122),	-- optional parameters
lampColorSf13	(123),	-- optional parameters
lampColorSf14	(124),	-- optional parameters
lampColorSf15	(125),	-- optional parameters
lampColorSf16	(126),	-- optional parameters
lampColorSf17	(127),	-- optional parameters
lampColorSf18	(128),	-- optional parameters
lampColorSf19	(129),	-- optional parameters
lampColorSf20	(130),	-- optional parameters
lampColorSf21	(131),	-- optional parameters
lampColorSf22	(132),	-- optional parameters
lampColorSf23	(133),	-- optional parameters
lampColorSf24	(134),	-- optional parameters
lampColorSf25	(135),	-- optional parameters
lampColorSf26	(136),	-- optional parameters
lampColorSf27	(137),	-- optional parameters
lampColorSf28	(138),	-- optional parameters
lampColorSf29	(139),	-- optional parameters
lampColorSf30	(140),	-- optional parameters
lampColorSf31	(141),	-- optional parameters
lampColorSf32	(142),	-- optional parameters
lampColorSf33	(143),	-- optional parameters
lampColorSf34	(144),	-- optional parameters
lampColorSf35	(145),	-- optional parameters
lampColorSf36	(146),	-- optional parameters
lampColorSf37	(147),	-- optional parameters
lampColorSf38	(148),	-- optional parameters

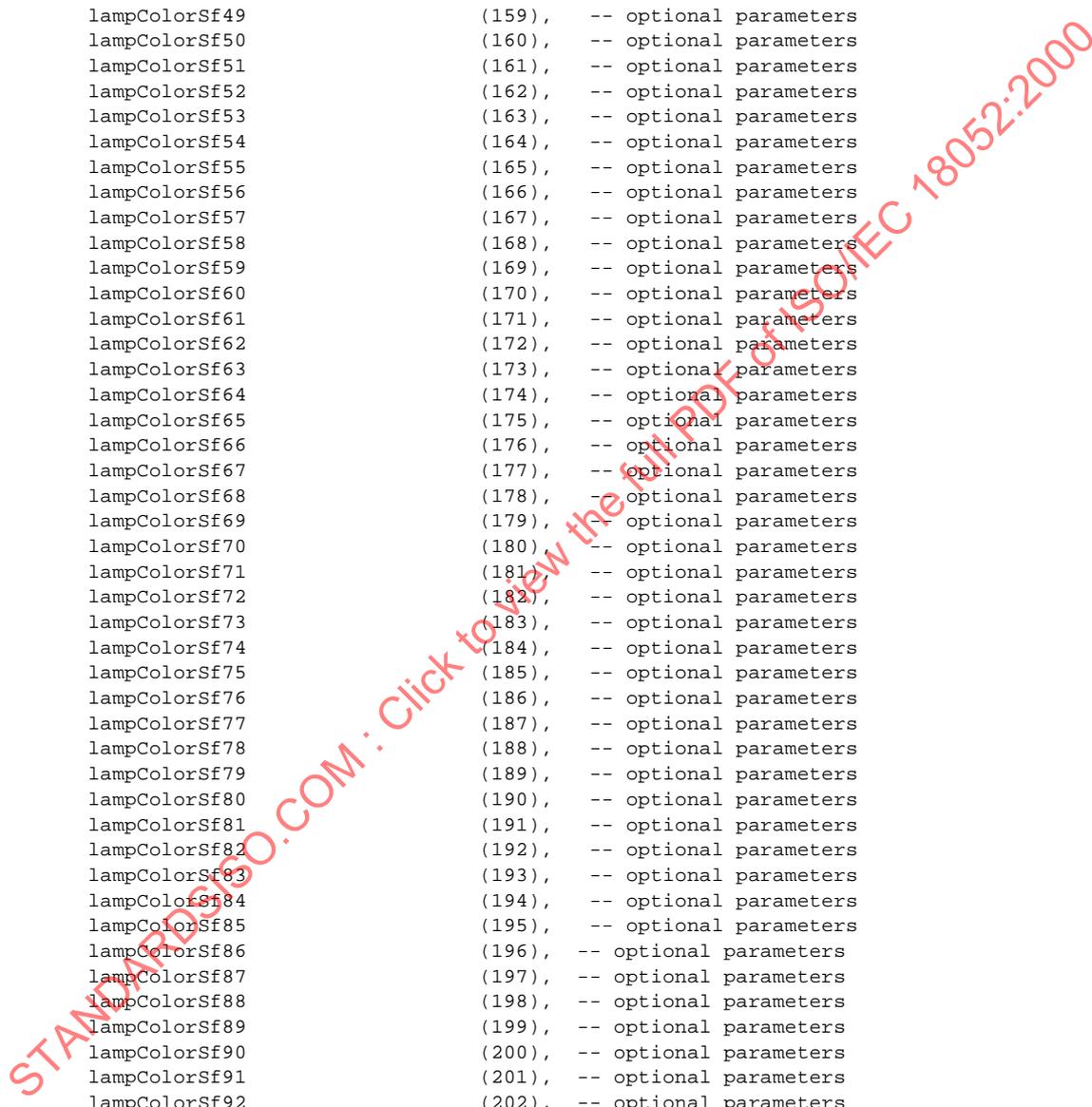
```

lampColorSf39          (149), -- optional parameters
lampColorSf40          (150), -- optional parameters
lampColorSf41          (151), -- optional parameters
lampColorSf42          (152), -- optional parameters
lampColorSf43          (153), -- optional parameters
lampColorSf44          (154), -- optional parameters
lampColorSf45          (155), -- optional parameters
lampColorSf46          (156), -- optional parameters
lampColorSf47          (157), -- optional parameters
lampColorSf48          (158), -- optional parameters
lampColorSf49          (159), -- optional parameters
lampColorSf50          (160), -- optional parameters
lampColorSf51          (161), -- optional parameters
lampColorSf52          (162), -- optional parameters
lampColorSf53          (163), -- optional parameters
lampColorSf54          (164), -- optional parameters
lampColorSf55          (165), -- optional parameters
lampColorSf56          (166), -- optional parameters
lampColorSf57          (167), -- optional parameters
lampColorSf58          (168), -- optional parameters
lampColorSf59          (169), -- optional parameters
lampColorSf60          (170), -- optional parameters
lampColorSf61          (171), -- optional parameters
lampColorSf62          (172), -- optional parameters
lampColorSf63          (173), -- optional parameters
lampColorSf64          (174), -- optional parameters
lampColorSf65          (175), -- optional parameters
lampColorSf66          (176), -- optional parameters
lampColorSf67          (177), -- optional parameters
lampColorSf68          (178), -- optional parameters
lampColorSf69          (179), -- optional parameters
lampColorSf70          (180), -- optional parameters
lampColorSf71          (181), -- optional parameters
lampColorSf72          (182), -- optional parameters
lampColorSf73          (183), -- optional parameters
lampColorSf74          (184), -- optional parameters
lampColorSf75          (185), -- optional parameters
lampColorSf76          (186), -- optional parameters
lampColorSf77          (187), -- optional parameters
lampColorSf78          (188), -- optional parameters
lampColorSf79          (189), -- optional parameters
lampColorSf80          (190), -- optional parameters
lampColorSf81          (191), -- optional parameters
lampColorSf82          (192), -- optional parameters
lampColorSf83          (193), -- optional parameters
lampColorSf84          (194), -- optional parameters
lampColorSf85          (195), -- optional parameters
lampColorSf86          (196), -- optional parameters
lampColorSf87          (197), -- optional parameters
lampColorSf88          (198), -- optional parameters
lampColorSf89          (199), -- optional parameters
lampColorSf90          (200), -- optional parameters
lampColorSf91          (201), -- optional parameters
lampColorSf92          (202), -- optional parameters
lampColorSf93          (203), -- optional parameters
lampColorSf94          (204), -- optional parameters
privateData            (205), -- optional parameters
privateDataInAck      (206), -- optional parameters
ackModelMultiStep     (207)} -- misc characteristics

SetMessageWaitingIndicator ::= BIT STRING
{
    deviceForMsg          ( 0), -- optional parameters
    privateData           ( 1), -- optional parameters
    privateDataInAck      ( 2), -- optional parameters
    ackModelMultiStep     ( 3)} -- misc characteristics

SetMicrophoneGain ::= BIT STRING
{
    microphoneGainAbs     (0), -- optional parameters

```



```

microphoneGainInc          (1),    -- optional parameters
privateData                (2),    -- optional parameters
privateDataInAck          (3),    -- optional parameters
ackModelMultiStep         (4)}   -- misc characteristics

SetMicrophoneMute ::= BIT STRING
{
  privateData              ( 0),    -- optional parameters
  privateDataInAck        ( 1),    -- optional parameters
  ackModelMultiStep       ( 2)}   -- misc characteristics

SetRingerStatus ::= BIT STRING
{
  ringerModeRinging       ( 0),    -- optional parameters
  ringerModeNotRinging    ( 1),    -- optional parameters
  ringVolumeAbs           ( 2),    -- optional parameters
  ringVolumeInc           ( 3),    -- optional parameters
  privateData             ( 4),    -- optional parameters
  privateDataInAck        ( 5),    -- optional parameters
  ackModelMultiStep       ( 6)}   -- misc characteristics

SetSpeakerMute ::= BIT STRING
{
  privateData              ( 0),    -- optional parameters
  privateDataInAck        ( 1),    -- optional parameters
  ackModelMultiStep       ( 2)}   -- misc characteristics

SetSpeakerVolume ::= BIT STRING
{
  speakerVolumeAbs        ( 0),    -- optional parameters
  speakerVolumeInc        ( 1),    -- optional parameters
  privateData             ( 2),    -- optional parameters
  privateDataInAck        ( 3),    -- optional parameters
  ackModelMultiStep       ( 4),    -- misc characteristics
  resettedAfterCall       ( 5),    -- misc characteristics
  notSettableWhileActive ( 6)}   -- misc characteristics

PhysDevEvtsList ::= SEQUENCE
{
  buttonInformation      [0] IMPLICIT ButtonInformation      OPTIONAL,
  buttonPress           [1] IMPLICIT ButtonPressEvent       OPTIONAL,
  displayUpdated        [2] IMPLICIT DisplayUpdated         OPTIONAL,
  hookswitch            [3] IMPLICIT Hookswitch             OPTIONAL,
  lampMode              [4] IMPLICIT LampMode               OPTIONAL,
  messageWaiting        [5] IMPLICIT MessageWaiting         OPTIONAL,
  microphoneGain        [6] IMPLICIT MicrophoneGain         OPTIONAL,
  microphoneMute        [7] IMPLICIT MicrophoneMute         OPTIONAL,
  ringerStatus          [8] IMPLICIT RingerStatus           OPTIONAL,
  speakerMute           [9] IMPLICIT SpeakerMute            OPTIONAL,
  speakerVolume         [10] IMPLICIT SpeakerVolume         OPTIONAL}

ButtonInformation ::= BIT STRING
{
  buttonLabel           ( 0),    -- optional parameters
  buttonAssociatedNumber ( 1),    -- optional parameters
  buttonPressIndicator  ( 2),    -- optional parameters
  privateData           ( 3)}   -- optional parameters

ButtonPressEvent ::= BIT STRING
{
  buttonLabel           ( 0),    -- optional parameters
  buttonAssociatedNumber ( 1),    -- optional parameters
  privateData           ( 2)}   -- optional parameters

DisplayUpdated ::= BIT STRING
{
  characterSetASCII     ( 0),    -- optional parameters
  characterSetUnicode   ( 1),    -- optional parameters
  characterSetProprietary ( 2),  -- optional parameters
  privateData           ( 3)}   -- optional parameters

Hookswitch ::= BIT STRING
{
  privateData           ( 0)}   -- optional parameters

LampMode ::= BIT STRING
{
  lampModeBrokenFlutter ( 0),    -- optional parameter

```

lampModeFlutter	( 1),	-- optional parameter
lampModeOff	( 2),	-- optional parameter
lampModeSteady	( 3),	-- optional parameter
lampModeWink	( 4),	-- optional parameter
lampModeReserved	( 5),	-- optional parameter
lampModeSf0	( 6),	-- optional parameters
lampModeSf1	( 7),	-- optional parameters
lampModeSf2	( 8),	-- optional parameters
lampModeSf3	( 9),	-- optional parameters
lampModeSf4	(10),	-- optional parameters
lampModeSf5	(11),	-- optional parameters
lampModeSf6	(12),	-- optional parameters
lampModeSf7	(13),	-- optional parameters
lampModeSf8	(14),	-- optional parameters
lampModeSf9	(15),	-- optional parameters
lampModeSf10	(16),	-- optional parameters
lampModeSf11	(17),	-- optional parameters
lampModeSf12	(18),	-- optional parameters
lampModeSf13	(19),	-- optional parameters
lampModeSf14	(20),	-- optional parameters
lampModeSf15	(21),	-- optional parameters
lampModeSf16	(22),	-- optional parameters
lampModeSf17	(23),	-- optional parameters
lampModeSf18	(24),	-- optional parameters
lampModeSf19	(25),	-- optional parameters
lampModeSf20	(26),	-- optional parameters
lampModeSf21	(27),	-- optional parameters
lampModeSf22	(28),	-- optional parameters
lampModeSf23	(29),	-- optional parameters
lampModeSf24	(30),	-- optional parameters
lampModeSf25	(31),	-- optional parameters
lampModeSf26	(32),	-- optional parameters
lampModeSf27	(33),	-- optional parameters
lampModeSf28	(34),	-- optional parameters
lampModeSf29	(35),	-- optional parameters
lampModeSf30	(36),	-- optional parameters
lampModeSf31	(37),	-- optional parameters
lampModeSf32	(38),	-- optional parameters
lampModeSf33	(39),	-- optional parameters
lampModeSf34	(40),	-- optional parameters
lampModeSf35	(41),	-- optional parameters
lampModeSf36	(42),	-- optional parameters
lampModeSf37	(43),	-- optional parameters
lampModeSf38	(44),	-- optional parameters
lampModeSf39	(45),	-- optional parameters
lampModeSf40	(46),	-- optional parameters
lampModeSf41	(47),	-- optional parameters
lampModeSf42	(48),	-- optional parameters
lampModeSf43	(49),	-- optional parameters
lampModeSf44	(50),	-- optional parameters
lampModeSf45	(51),	-- optional parameters
lampModeSf46	(52),	-- optional parameters
lampModeSf47	(53),	-- optional parameters
lampModeSf48	(54),	-- optional parameters
lampModeSf49	(55),	-- optional parameters
lampModeSf50	(56),	-- optional parameters
lampModeSf51	(57),	-- optional parameters
lampModeSf52	(58),	-- optional parameters
lampModeSf53	(59),	-- optional parameters
lampModeSf54	(60),	-- optional parameters
lampModeSf55	(61),	-- optional parameters
lampModeSf56	(62),	-- optional parameters
lampModeSf57	(63),	-- optional parameters
lampModeSf58	(64),	-- optional parameters
lampModeSf59	(65),	-- optional parameters
lampModeSf60	(66),	-- optional parameters
lampModeSf61	(67),	-- optional parameters
lampModeSf62	(68),	-- optional parameters

STANDARD 350.COM : Click to view the full PDF of ISO/IEC 18052:2000

lampModeSf63	(69),	-- optional parameters
lampModeSf64	(70),	-- optional parameters
lampModeSf65	(71),	-- optional parameters
lampModeSf66	(72),	-- optional parameters
lampModeSf67	(73),	-- optional parameters
lampModeSf68	(74),	-- optional parameters
lampModeSf69	(75),	-- optional parameters
lampModeSf70	(76),	-- optional parameters
lampModeSf71	(77),	-- optional parameters
lampModeSf72	(78),	-- optional parameters
lampModeSf73	(79),	-- optional parameters
lampModeSf74	(80),	-- optional parameters
lampModeSf75	(81),	-- optional parameters
lampModeSf76	(82),	-- optional parameters
lampModeSf77	(83),	-- optional parameters
lampModeSf78	(84),	-- optional parameters
lampModeSf79	(85),	-- optional parameters
lampModeSf80	(86),	-- optional parameters
lampModeSf81	(87),	-- optional parameters
lampModeSf82	(88),	-- optional parameters
lampModeSf83	(89),	-- optional parameters
lampModeSf84	(90),	-- optional parameters
lampModeSf85	(91),	-- optional parameters
lampModeSf86	(92),	-- optional parameters
lampModeSf87	(93),	-- optional parameters
lampModeSf88	(94),	-- optional parameters
lampModeSf89	(95),	-- optional parameters
lampModeSf90	(96),	-- optional parameters
lampModeSf91	(97),	-- optional parameters
lampModeSf92	(98),	-- optional parameters
lampModeSf93	(99),	-- optional parameters
lampModeSf94	(100),	-- optional parameter
lampBrightnessNormal	(101),	-- optional parameter
lampBrightnessDim	(102),	-- optional parameter
lampBrightnessBright	(103),	-- optional parameter
lampColorNoColor	(104),	-- optional parameter
lampColorRed	(105),	-- optional parameter
lampColorYellow	(106),	-- optional parameter
lampColorGreen	(107),	-- optional parameter
lampColorBlue	(108),	-- optional parameter
lampColorReserved	(109),	-- optional parameter
lampColorSf0	(110),	-- optional parameters
lampColorSf1	(111),	-- optional parameters
lampColorSf2	(112),	-- optional parameters
lampColorSf3	(113),	-- optional parameters
lampColorSf4	(114),	-- optional parameters
lampColorSf5	(115),	-- optional parameters
lampColorSf6	(116),	-- optional parameters
lampColorSf7	(117),	-- optional parameters
lampColorSf8	(118),	-- optional parameters
lampColorSf9	(119),	-- optional parameters
lampColorSf10	(120),	-- optional parameters
lampColorSf11	(121),	-- optional parameters
lampColorSf12	(122),	-- optional parameters
lampColorSf13	(123),	-- optional parameters
lampColorSf14	(124),	-- optional parameters
lampColorSf15	(125),	-- optional parameters
lampColorSf16	(126),	-- optional parameters
lampColorSf17	(127),	-- optional parameters
lampColorSf18	(128),	-- optional parameters
lampColorSf19	(129),	-- optional parameters
lampColorSf20	(130),	-- optional parameters
lampColorSf21	(131),	-- optional parameters
lampColorSf22	(132),	-- optional parameters
lampColorSf23	(133),	-- optional parameters
lampColorSf24	(134),	-- optional parameters
lampColorSf25	(135),	-- optional parameters
lampColorSf26	(136),	-- optional parameters

lampColorSf27	(137),	-- optional parameters
lampColorSf28	(138),	-- optional parameters
lampColorSf29	(139),	-- optional parameters
lampColorSf30	(140),	-- optional parameters
lampColorSf31	(141),	-- optional parameters
lampColorSf32	(142),	-- optional parameters
lampColorSf33	(143),	-- optional parameters
lampColorSf34	(144),	-- optional parameters
lampColorSf35	(145),	-- optional parameters
lampColorSf36	(146),	-- optional parameters
lampColorSf37	(147),	-- optional parameters
lampColorSf38	(148),	-- optional parameters
lampColorSf39	(149),	-- optional parameters
lampColorSf40	(150),	-- optional parameters
lampColorSf41	(151),	-- optional parameters
lampColorSf42	(152),	-- optional parameters
lampColorSf43	(153),	-- optional parameters
lampColorSf44	(154),	-- optional parameters
lampColorSf45	(155),	-- optional parameters
lampColorSf46	(156),	-- optional parameters
lampColorSf47	(157),	-- optional parameters
lampColorSf48	(158),	-- optional parameters
lampColorSf49	(159),	-- optional parameters
lampColorSf50	(160),	-- optional parameters
lampColorSf51	(161),	-- optional parameters
lampColorSf52	(162),	-- optional parameters
lampColorSf53	(163),	-- optional parameters
lampColorSf54	(164),	-- optional parameters
lampColorSf55	(165),	-- optional parameters
lampColorSf56	(166),	-- optional parameters
lampColorSf57	(167),	-- optional parameters
lampColorSf58	(168),	-- optional parameters
lampColorSf59	(169),	-- optional parameters
lampColorSf60	(170),	-- optional parameters
lampColorSf61	(171),	-- optional parameters
lampColorSf62	(172),	-- optional parameters
lampColorSf63	(173),	-- optional parameters
lampColorSf64	(174),	-- optional parameters
lampColorSf65	(175),	-- optional parameters
lampColorSf66	(176),	-- optional parameters
lampColorSf67	(177),	-- optional parameters
lampColorSf68	(178),	-- optional parameters
lampColorSf69	(179),	-- optional parameters
lampColorSf70	(180),	-- optional parameters
lampColorSf71	(181),	-- optional parameters
lampColorSf72	(182),	-- optional parameters
lampColorSf73	(183),	-- optional parameters
lampColorSf74	(184),	-- optional parameters
lampColorSf75	(185),	-- optional parameters
lampColorSf76	(186),	-- optional parameters
lampColorSf77	(187),	-- optional parameters
lampColorSf78	(188),	-- optional parameters
lampColorSf79	(189),	-- optional parameters
lampColorSf80	(190),	-- optional parameters
lampColorSf81	(191),	-- optional parameters
lampColorSf82	(192),	-- optional parameters
lampColorSf83	(193),	-- optional parameters
lampColorSf84	(194),	-- optional parameters
lampColorSf85	(195),	-- optional parameters
lampColorSf86	(196),	-- optional parameters
lampColorSf87	(197),	-- optional parameters
lampColorSf88	(198),	-- optional parameters
lampColorSf89	(199),	-- optional parameters
lampColorSf90	(200),	-- optional parameters
lampColorSf91	(201),	-- optional parameters
lampColorSf92	(202),	-- optional parameters
lampColorSf93	(203),	-- optional parameters
lampColorSf94	(204),	-- optional parameters

STANDARDSDO.COM : Click to view the PDF of ISO/IEC 18052:2000

```

        privateData                (205)} -- optional parameters

MessageWaiting ::= BIT STRING
{
    deviceForMsg                ( 0), -- optional parameters
    privateData                 ( 1)} -- optional parameters

MicrophoneGain ::= BIT STRING
{
    microphoneGainAbs           ( 0), -- optional parameters
    microphoneGainInc           ( 1), -- optional parameters
    privateData                 ( 2)} -- optional parameters

MicrophoneMute ::= BIT STRING
{
    privateData                 ( 0)} -- optional parameters

RingerStatus ::= BIT STRING
{
    ringerModeRinging           ( 0), -- optional parameters
    ringerModeNotRinging        ( 1), -- optional parameters
    ringCount                   ( 2), -- optional parameters
    ringPattern                 ( 3), -- optional parameters
    ringVolumeAbs               ( 4), -- optional parameters
    ringVolumeInc               ( 5), -- optional parameters
    privateData                 ( 6)} -- optional parameters

SpeakerMute ::= BIT STRING
{
    privateData                 ( 0)} -- optional parameters

SpeakerVolume ::= BIT STRING
{
    speakerVolumeAbs            ( 0), -- optional parameters
    speakerVolumeInc            ( 1), -- optional parameters
    privateData                 ( 2)} -- optional parameters

LogicalServList ::= SEQUENCE
{
    callBackNonCallRel          [ 0] IMPLICIT CallBackNonCallRel          OPTIONAL,
    callBackMsgNonCallRel       [ 1] IMPLICIT CallBackMsgNonCallRel       OPTIONAL,
    cancelCallBack              [ 2] IMPLICIT CancelCallBack              OPTIONAL,
    cancelCallBackMsg           [ 3] IMPLICIT CancelCallBackMsg           OPTIONAL,
    getAgentState               [ 4] IMPLICIT GetAgentState               OPTIONAL,
    getAutoAnswer               [ 5] IMPLICIT GetAutoAnswer               OPTIONAL,
    getAutoWorkMode             [ 6] IMPLICIT GetAutoWorkMode             OPTIONAL,
    getCallerIDStatus           [ 7] IMPLICIT GetCallerIDStatus           OPTIONAL,
    getDoNotDisturb             [ 8] IMPLICIT GetDoNotDisturb             OPTIONAL,
    getForwarding               [ 9] IMPLICIT GetForwarding               OPTIONAL,
    getLastNumberDialed          [10] IMPLICIT GetLastNumberDialed          OPTIONAL,
    getRouteingMode             [11] IMPLICIT GetRouteingMode             OPTIONAL,
    setAgentState               [12] IMPLICIT SetAgentState               OPTIONAL,
    setAutoAnswer               [13] IMPLICIT SetAutoAnswer               OPTIONAL,
    setAutoWorkMode             [14] IMPLICIT SetAutoWorkMode             OPTIONAL,
    setCallerIDStatus           [15] IMPLICIT SetCallerIDStatus           OPTIONAL,
    setDoNotDisturb             [16] IMPLICIT SetDoNotDisturb             OPTIONAL,
    setForwarding               [17] IMPLICIT SetForwarding               OPTIONAL,
    setRouteingMode             [18] IMPLICIT SetRouteingMode             OPTIONAL}

CallBackNonCallRel ::= BIT STRING
{
    privateData                 ( 0), -- optional parameters
    privateDataInAck            ( 1), -- optional parameters
    additionalReqForbidden       ( 2), -- optional parameters
    ackModelMultiStep           ( 3)} -- misc characteristics

CallBackMsgNonCallRel ::= BIT STRING
{
    privateData                 ( 0), -- optional parameters
    privateDataInAck            ( 1), -- optional parameters
    additionalReqForbidden       ( 2), -- optional parameters
    ackModelMultiStep           ( 3)} -- misc characteristics

CancelCallBack ::= BIT STRING
{
    privateData                 ( 0), -- optional parameters
    privateDataInAck            ( 1), -- optional parameters
    ackModelMultiStep           ( 2), -- misc characteristics

```

```

        supportsClearing                ( 3)} -- misc characteristics

CancelCallBackMsg ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    privateDataInAck           ( 1), -- optional parameters
    ackModelMultiStep          ( 2), -- misc characteristics
    supportsClearing           ( 3)} -- misc characteristics

GetAgentState ::= BIT STRING
{
    acdGroup                    ( 0), -- optional parameters
    privateData                ( 1), -- optional parameters
    agentStateListAgentIDInAck ( 2), -- optional parameters
    agentGroupInAck            ( 3), -- optional parameters
    pendingAgentStateInAck     ( 4), -- optional parameters
    agentStateConditionForcedPauseInAck ( 5), -- optional parameters
    agentStateConditionPauseInAck ( 6), -- optional parameters
    privateDataInAck           ( 7)} -- optional parameters

GetAutoAnswer ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    numberOfRingsInAck         ( 1), -- optional parameters
    privateDataInAck           ( 2)} -- optional parameters

GetAutoWorkMode ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    autoWorkIntervalInAck     ( 1), -- optional parameters
    privateDataInAck           ( 2)} -- optional parameters

GetCallerIDStatus ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    privateDataInAck           ( 1)} -- optional parameters

GetDoNotDisturb ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    callOriginParameterInAck  ( 1), -- optional parameters
    callOriginInternInAck     ( 2), -- optional parameters
    callOriginExternInAck     ( 3), -- optional parameters
    callingDeviceListInAck    ( 4), -- optional parameters
    privateDataInAck           ( 5)} -- optional parameters

GetForwarding ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    forwardListInAck           ( 1), -- optional parameters
    forwardListImmediateInAck ( 2), -- optional parameters
    forwardListBusyInAck      ( 3), -- optional parameters
    forwardListDNDInAck       ( 4), -- optional parameters
    forwardListNoAnsInAck     ( 5), -- optional parameters
    forwardListBusyIntInAck   ( 6), -- optional parameters
    forwardListBusyExtInAck   ( 7), -- optional parameters
    forwardListDNDIntInAck    ( 8), -- optional parameters
    forwardListDNDExtInAck    ( 9), -- optional parameters
    forwardListNoAnsIntInAck  (10), -- optional parameters
    forwardListNoAnsExtInAck  (11), -- optional parameters
    forwardListImmIntInAck    (12), -- optional parameters
    forwardListImmExtInAck    (13), -- optional parameters
    forwardDNInAck            (14), -- optional parameters
    forwardDefaultInAck       (15), -- optional parameters
    forwardDefaultTypeAndDNInAck (16), -- optional parameters
    forwardDefaultTypeInAck   (17), -- optional parameters
    forwardDefaultDNInAck     (18), -- optional parameters
    ringCountInAck            (19), -- optional parameters
    privateDataInAck           (20)} -- optional parameters

GetLastNumberDialed ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    privateDataInAck           ( 1)} -- optional parameters

GetRoutingMode ::= BIT STRING

```



```

{      privateData          ( 0),  -- optional parameters
      privateDataInAck     ( 1)}  -- optional parameters

SetAgentState ::= BIT STRING
{
  requestedAgentStateLoggedOn      ( 0),  -- optional parameters
  requestedAgentStateLoggedOff     ( 1),  -- optional parameters
  requestedAgentStateNotReady      ( 2),  -- optional parameters
  requestedAgentStateReady         ( 3),  -- optional parameters
  requestedAgentStateWorkingAfterCall ( 4),  -- optional parameters
  agentID                          ( 5),  -- optional parameters
  password                         ( 6),  -- optional parameters
  group                            ( 7),  -- optional parameters
  privateData                      ( 8),  -- optional parameters
  pendingAgentStateWorkingAfterCallInAck ( 9),  -- optional parameters
  pendingAgentStateNotReadyInAck   (10),  -- optional parameters
  pendingAgentStateNullInAck       (11),  -- optional parameters
  privateDataInAck                 (12),  -- optional parameters
  ackModelMultiStep                (13),  -- misc characteristics
  groupDeviceAllowedInReq          (14),  -- misc characteristics
  aCDDDeviceAllowedInReq           (15),  -- misc characteristics
  delayTransitionIfBusy             (16),  -- misc characteristics
  delayTransitionIfWorkingAfterCall (17)}  -- misc characteristics

SetAutoAnswer ::= BIT STRING
{
  numberOfRings          ( 0),  -- optional parameters
  privateData            ( 1),  -- optional parameters
  privateDataInAck       ( 2),  -- optional parameters
  ackModelMultiStep      ( 3)}  -- misc characteristics

SetAutoWorkMode ::= BIT STRING
{
  autoWorkInterval      ( 0),  -- optional parameters
  privateData           ( 1),  -- optional parameters
  privateDataInAck      ( 2),  -- optional parameters
  ackModelMultiStep     ( 3),  -- misc characteristics
  groupDeviceAllowedInReq ( 4),  -- misc characteristics
  aCDDDeviceAllowedInReq ( 5)}  -- misc characteristics

SetCallerIDStatus ::= BIT STRING
{
  privateData          ( 0),  -- optional parameters
  privateDataInAck     ( 1),  -- optional parameters
  ackModelMultiStep    ( 2)}  -- misc characteristics

SetDoNotDisturb ::= BIT STRING
{
  callOriginationInternal      ( 0),  -- optional parameters
  callOriginationExternal     ( 1),  -- optional parameters
  callingDeviceList            ( 2),  -- optional parameters
  privateData                  ( 3),  -- optional parameters
  privateDataInAck             ( 4),  -- optional parameters
  ackModelMultiStep            ( 5)}  -- misc characteristics

SetForwarding ::= BIT STRING
{
  forwardingTypeBusy      ( 0),  -- optional parameters
  forwardingTypeBusyInt   ( 1),  -- optional parameters
  forwardingTypeBusyExt   ( 2),  -- optional parameters
  forwardingTypeDND       ( 3),  -- optional parameters
  forwardingTypeDNDInt    ( 4),  -- optional parameters
  forwardingTypeDNDExt    ( 5),  -- optional parameters
  forwardingTypeNoAns     ( 6),  -- optional parameters
  forwardingTypeNoAnsInt  ( 7),  -- optional parameters
  forwardingTypeNoAnsExt  ( 8),  -- optional parameters
  forwardingTypeImmediate ( 9),  -- optional parameters
  forwardingTypeImmInt     (10),  -- optional parameters
  forwardingTypeImmExt     (11),  -- optional parameters
  forwardDN                (12),  -- optional parameters
  ringCount                (13),  -- optional parameters
  privateData              (14),  -- optional parameters
  privateDataInAck         (15),  -- optional parameters
  ackModelMultiStep        (16)}  -- misc characteristics

```

```

SetRouteingMode ::= BIT STRING
{
    privateData                ( 0), -- optional parameters
    privateDataInAck          ( 1), -- optional parameters
    ackModelMultiStep         ( 2)} -- misc characteristics

LogicalEvtsList ::= SEQUENCE
{
    agentBusy                [0] IMPLICIT AgentBusy                OPTIONAL,
    agentLoggedOff           [1] IMPLICIT AgentLoggedOff           OPTIONAL,
    agentLoggedOn           [2] IMPLICIT AgentLoggedOn           OPTIONAL,
    agentNotReady           [3] IMPLICIT AgentNotReady           OPTIONAL,
    agentReady              [4] IMPLICIT AgentReady              OPTIONAL,
    agentWorkingAfterCall   [5] IMPLICIT AgentWorkingAfterCall   OPTIONAL,
    autoAnswer              [6] IMPLICIT AutoAnswer              OPTIONAL,
    autoWorkMode            [7] IMPLICIT AutoWorkMode            OPTIONAL,
    callBack                [8] IMPLICIT CallBackEvent           OPTIONAL,
    callBackMessage         [9] IMPLICIT CallBackMessageEvent    OPTIONAL,
    callerIDStatus          [10] IMPLICIT CallerIDStatus          OPTIONAL,
    doNotDisturb            [11] IMPLICIT DoNotDisturb            OPTIONAL,
    forwarding              [12] IMPLICIT Forwarding              OPTIONAL,
    routeingMode            [13] IMPLICIT RouteingMode            OPTIONAL}

AgentBusy ::= BIT STRING
{
    agentID                  ( 0), -- optional parameters
    acdGroup                 ( 1), -- optional parameters
    pendingAgentStateWorkingAfterCall ( 2), -- optional parameters
    pendingAgentStateNotReady ( 3), -- optional parameters
    pendingAgentStateReady  ( 4), -- optional parameters
    pendingAgentStateNull   ( 5), -- optional parameters
    cause                    ( 6), -- optional parameters
    privateData              ( 7)} -- optional parameters

AgentLoggedOff ::= BIT STRING
{
    agentID                  ( 0), -- optional parameters
    acdGroup                 ( 1), -- optional parameters
    agentPassword            ( 2), -- optional parameters
    cause                    ( 3), -- optional parameters
    privateData              ( 4)} -- optional parameters

AgentLoggedOn ::= BIT STRING
{
    agentID                  ( 0), -- optional parameters
    acdGroup                 ( 1), -- optional parameters
    agentPassword            ( 2), -- optional parameters
    cause                    ( 3), -- optional parameters
    privateData              ( 4)} -- optional parameters

AgentNotReady ::= BIT STRING
{
    agentID                  ( 0), -- optional parameters
    acdGroup                 ( 1), -- optional parameters
    cause                    ( 2), -- optional parameters
    privateData              ( 3)} -- optional parameters

AgentReady ::= BIT STRING
{
    agentID                  ( 0), -- optional parameters
    acdGroup                 ( 1), -- optional parameters
    cause                    ( 2), -- optional parameters
    privateData              ( 3)} -- optional parameters

AgentWorkingAfterCall ::= BIT STRING
{
    agentID                  ( 0), -- optional parameters
    acdGroup                 ( 1), -- optional parameters
    pendingAgentStateNotReady ( 2), -- optional parameters
    pendingAgentStateReady  ( 3), -- optional parameters
    pendingAgentStateNull   ( 4), -- optional parameters
    cause                    ( 5), -- optional parameters
    privateData              ( 6)} -- optional parameters

```

```

AutoAnswer ::= BIT STRING
{
    numberOfRings          ( 0),  -- optional parameters
    privateData            ( 1)}  -- optional parameters

AutoWorkMode ::= BIT STRING
{
    privateData            ( 0)}  -- optional parameters

CallBackEvent ::= BIT STRING
{
    privateData            ( 0)}  -- optional parameters

CallBackMessageEvent ::= BIT STRING
{
    privateData            ( 0)}  -- optional parameters

CallerIDStatus ::= BIT STRING
{
    privateData            ( 0)}  -- optional parameters

DoNotDisturb ::= BIT STRING
{
    callOriginationParameter ( 0),  -- optional parameters
    callOriginationInternal  ( 1),  -- optional parameters
    callOriginationExternal  ( 2),  -- optional parameters
    callingDeviceList        ( 3),  -- optional parameters
    privateData              ( 4)}  -- optional parameters

Forwarding ::= BIT STRING
{
    forwardingTypeBusy        ( 0),
    forwardingTypeBusyInt     ( 1),
    forwardingTypeBusyExt     ( 2),
    forwardingTypeDND         ( 3),
    forwardingTypeDNDInt      ( 4),
    forwardingTypeDNDExt      ( 5),
    forwardingTypeNoAns       ( 6),
    forwardingTypeNoAnsInt    ( 7),
    forwardingTypeNoAnsExt    ( 8),
    forwardingTypeImmediate   ( 9),
    forwardingTypeImmInt      (10),
    forwardingTypeImmExt      (11),
    forwardTo                  (12),
    forwardDefaultTypeAndDN   (13),
    forwardDefaultType        (14),
    forwardDefaultDN          (15),
    ringCount                  (16),
    privateData                (17)}

RouteingMode ::= BIT STRING
{
    privateData            ( 0)}  -- optional parameters

DeviceMaintEvtsList ::= SEQUENCE
{
    backInService          [0] IMPLICIT BackInService          OPTIONAL,
    deviceCapsChanged      [1] IMPLICIT DeviceCapsChanged      OPTIONAL,
    outOfService           [2] IMPLICIT OutOfService           OPTIONAL}

BackInService ::= BIT STRING
{
    cause                  ( 0),  -- optional parameters
    privateData            ( 1)}  -- optional parameters

DeviceCapsChanged ::= BIT STRING
{
    cause                  ( 0),  -- optional parameters
    privateData            ( 1)}  -- optional parameters

OutOfService ::= BIT STRING
{
    cause                  ( 0),  -- optional parameters
    privateData            ( 1)}  -- optional parameters

IOServicesServList ::= SEQUENCE
{
    ioRegister             [ 0] IMPLICIT IoRegister            OPTIONAL,
    ioRegisterAbort        [ 1] IMPLICIT IoRegisterAbort       OPTIONAL,
    ioRegisterCancel       [ 2] IMPLICIT IoRegisterCancel       OPTIONAL,
    dataPathResumed        [ 3] IMPLICIT DataPathResumed        OPTIONAL,

```

```

dataPathSuspended      [ 4] IMPLICIT DataPathSuspended OPTIONAL,
fastData                [ 5] IMPLICIT FastData          OPTIONAL,
resumeDataPath          [ 6] IMPLICIT ResumeDataPath      OPTIONAL,
sendBroadcastData       [ 7] IMPLICIT SendBroadcastData   OPTIONAL,
sendData                [ 8] IMPLICIT SendData          OPTIONAL,
sendMulticastData       [ 9] IMPLICIT SendMulticastData  OPTIONAL,
startDataPath           [10] IMPLICIT StartDataPath     OPTIONAL,
stopDataPath            [11] IMPLICIT StopDataPath      OPTIONAL,
suspendDataPath         [12] IMPLICIT SuspendDataPath    OPTIONAL}

IoRegister ::= BIT STRING
{
    ioDevice              ( 0), -- optional parameters
    privateData           ( 1), -- optional parameters
    privateDataInAck      ( 2), -- optional parameters
    allIODevices          ( 3)} -- misc characteristics

IoRegisterAbort ::= BIT STRING
{
    privateData           ( 0)} -- optional parameters

IoRegisterCancel ::= BIT STRING
{
    privateData           ( 0), -- optional parameters
    privateDataInAck      ( 1)} -- optional parameters

DataPathResumed ::= BIT STRING
{
    privateData           ( 0), -- optional parameters
    privateDataInAck      ( 1)} -- optional parameters

DataPathSuspended ::= BIT STRING
{
    privateData           ( 0), -- optional parameters
    privateDataInAck      ( 1)} -- optional parameters

FastData ::= BIT STRING
{
    objectDevice          ( 0), -- optional parameters
    objectCall            ( 1), -- optional parameters
    dataPathTypeText      ( 2), -- optional parameters
    dataPathTypeVoice     ( 3), -- optional parameters
    displayAttribPhyBaseRowNumber ( 4), -- optional parameters
    displayAttribPhyBaseColumnNumber ( 5), -- optional parameters
    displayAttribOffset   ( 6), -- optional parameters
    privateData           ( 7), -- optional parameters
    privateDataInAck      ( 8), -- optional parameters
    supportsModifyingPosition ( 9)} -- misc characteristics

ResumeDataPath ::= BIT STRING
{
    privateData           ( 0), -- optional parameters
    privateDataInAck      ( 1), -- optional parameters
    sendsDataPathResumed ( 2)} -- misc characteristics

SendBroadcastData ::= BIT STRING
{
    privateData           ( 0), -- optional parameters
    dataPathTypeText      ( 1), -- optional parameters
    dataPathTypeVoice     ( 2), -- optional parameters
    displayAttribPhyBaseRowNumber ( 3), -- optional parameters
    displayAttribPhyBaseColumnNumber ( 4), -- optional parameters
    displayAttribOffset   ( 5), -- optional parameters
    privateDataInAck      ( 6), -- optional parameters
    supportsModifyingPosition ( 7)} -- misc characteristics

SendData ::= BIT STRING
{
    displayAttribPhyBaseRowNumber ( 0), -- optional parameters
    displayAttribPhyBaseColumnNumber ( 1), -- optional parameters
    displayAttribOffset   ( 2), -- optional parameters
    ioCauseTerminationCharReceived ( 3), -- optional parameters
    ioCauseCharCountReached ( 4), -- optional parameters
    ioCauseTimeout        ( 5), -- optional parameters
    ioCauseSfTerminated   ( 6), -- optional parameters
    privateData           ( 7), -- optional parameters
    privateDataInAck      ( 8), -- optional parameters

```



```

        supportsModifyingPosition          ( 9) } -- misc characteristics

SendMulticastData ::= BIT STRING
{
    ioData                                ( 0), -- optional parameters
    displayAttribPhyBaseRowNumber         ( 1), -- optional parameters
    displayAttribPhyBaseColumnNumber      ( 2), -- optional parameters
    displayAttribOffset                   ( 3), -- optional parameters
    privateData                            ( 4), -- optional parameters
    privateDataInAck                      ( 5), -- optional parameters
    supportsModifyingPosition             ( 6) } -- misc characteristics

StartDataPath ::= BIT STRING
{
    objectDevice                           ( 0), -- optional parameters
    objectCall                             ( 1), -- optional parameters
    dataPathDirectionCfToObject            ( 2), -- optional parameters
    dataPathDirectionObjectToCf           ( 3), -- optional parameters
    dataPathDirectionBidirectional        ( 4), -- optional parameters
    dataPathTypeText                      ( 5), -- optional parameters
    dataPathTypeVoice                     ( 6), -- optional parameters
    numberOfCharsToCollect                 ( 7), -- optional parameters
    terminationChar                       ( 8), -- optional parameters
    timeout                                ( 9), -- optional parameters
    privateData                            (10), -- optional parameters
    numberOfCharsToCollectInAck           (11), -- optional parameters
    terminationCharInAck                  (12), -- optional parameters
    timeoutInAck                          (13), -- optional parameters
    privateDataInAck                      (14) } -- optional parameters

StopDataPath ::= BIT STRING
{
    privateData                            ( 0), -- optional parameters
    privateDataInAck                      ( 1) } -- optional parameters

SuspendDataPath ::= BIT STRING
{
    privateData                            ( 0), -- optional parameters
    privateDataInAck                      ( 1) } -- optional parameters

DataCollectionServList ::= SEQUENCE
{
    dataCollected                        [ 0] IMPLICIT DataCollected           OPTIONAL,
    dataCollectionResumed                 [ 1] IMPLICIT DataCollectionResumed     OPTIONAL,
    dataCollectionSuspended               [ 2] IMPLICIT DataCollectionSuspended   OPTIONAL,
    resumeDataCollection                  [ 3] IMPLICIT ResumeDataCollection      OPTIONAL,
    startDataCollection                   [ 4] IMPLICIT StartDataCollection       OPTIONAL,
    stopDataCollection                    [ 5] IMPLICIT StopDataCollection        OPTIONAL,
    suspendDataCollection                  [ 6] IMPLICIT SuspendDataCollection     OPTIONAL}

DataCollected ::= BIT STRING
{
    digitsDuration                        ( 0), -- optional parameters
    digitsPauseDuration                   ( 1), -- optional parameters
    toneDetectedBeep                      ( 2), -- optional parameters
    toneDetectedBilling                    ( 3), -- optional parameters
    toneDetectedBusy                       ( 4), -- optional parameters
    toneDetectedCarrier                    ( 5), -- optional parameters
    toneDetectedConfirmation                ( 6), -- optional parameters
    toneDetectedDial                       ( 7), -- optional parameters
    toneDetectedFaxCNG                     ( 8), -- optional parameters
    toneDetectedHold                       ( 9), -- optional parameters
    toneDetectedHowler                     (10), -- optional parameters
    toneDetectedIntrusion                  (11), -- optional parameters
    toneDetectedModemCNG                   (12), -- optional parameters
    toneDetectedPark                       (13), -- optional parameters
    toneDetectedRecordWarning               (14), -- optional parameters
    toneDetectedReorder                    (15), -- optional parameters
    toneDetectedRingback                   (16), -- optional parameters
    toneDetectedSilence                    (17), -- optional parameters
    toneDetectedSitVC                      (18), -- optional parameters
    toneDetectedSitIC                      (19), -- optional parameters
    toneDetectedSitRO                      (20), -- optional parameters
    toneDetectedSitNC                      (21), -- optional parameters

```

```

toneDetectedSf0          (22),  -- optional parameters
toneDetectedSf1          (23),  -- optional parameters
toneDetectedSf2          (24),  -- optional parameters
toneDetectedSf3          (25),  -- optional parameters
toneDetectedSf4          (26),  -- optional parameters
toneDetectedSf5          (27),  -- optional parameters
toneDetectedSf6          (28),  -- optional parameters
toneDetectedSf7          (29),  -- optional parameters
toneDetectedSf8          (30),  -- optional parameters
toneDetectedSf9          (31),  -- optional parameters
toneDetectedSf10         (32),  -- optional parameters
toneDetectedOther        (33),  -- optional parameters
toneFrequency            (34),  -- optional parameters
toneDuration             (35),  -- optional parameters
tonePauseDuration        (36),
connectionInfo           (37),  -- optional parameters
dcollCauseFushCharReceived (38),  -- optional parameters
dcollCauseCharCountReached (39),  -- optional parameters
dcollCauseTimeout        (40),  -- optional parameters
dcollCauseSFTerminated   (41),  -- optional parameters
privateData              (42),  -- optional parameters
privateDataInAck         (43)}  -- optional parameters

DataCollectionResumed ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck     ( 1)}  -- optional parameters

DataCollectionSuspended ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck     ( 1)}  -- optional parameters

ResumeDataCollection ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck     ( 1)}  -- optional parameters

StartDataCollection ::= BIT STRING
{
    objectDevice         ( 0),  -- optional parameters
    objectCall           ( 1),  -- optional parameters
    dataCollectionTypeDigits ( 2),  -- optional parameters
    dataCollectionTypeTones ( 3),  -- optional parameters
    digitsReportingCriteriaNumChars ( 4),  -- optional parameters
    digitsReportingCriteriaFlushChar ( 5),  -- optional parameters
    digitsReportingCriteriaTimeout ( 6),  -- optional parameters
    privateData          ( 7),  -- optional parameters
    privateDataInAck     ( 8)}  -- optional parameters

StopDataCollection ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck     ( 1)}  -- optional parameters

SuspendDataCollection ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck     ( 1)}  -- optional parameters

VoiceUnitServList ::= SEQUENCE
{
    concatenateMsg       [ 0] IMPLICIT ConcatenateMsg  OPTIONAL,
    deleteMsg           [ 1] IMPLICIT DeleteMsg        OPTIONAL,
    playMsg              [ 2] IMPLICIT PlayMsg          OPTIONAL,
    queryVoiceAttrib     [ 3] IMPLICIT QueryVoiceAttrib OPTIONAL,
    recordMsg            [ 4] IMPLICIT RecordMsg        OPTIONAL,
    reposition           [ 5] IMPLICIT Reposition        OPTIONAL,
    resume               [ 6] IMPLICIT Resume           OPTIONAL,
    review               [ 7] IMPLICIT Review            OPTIONAL,
    setVoiceAttrib       [ 8] IMPLICIT SetVoiceAttrib    OPTIONAL,
    stop                 [ 9] IMPLICIT Stop              OPTIONAL,
    suspend              [10] IMPLICIT Suspend           OPTIONAL,
    synthesizeMsg       [11] IMPLICIT SynthesizeMsg     OPTIONAL}

```

```

ConcatenateMsg ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck    ( 1)}  -- optional parameters

DeleteMsg ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck    ( 1)}  -- optional parameters

PlayMsg ::= BIT STRING
{
    duration              ( 0),  -- optional parameters
    terminationParameter ( 1),  -- optional parameters
    terminationDurationExceeded ( 2), -- optional parameters
    terminationDTMFDigitDetected ( 3), -- optional parameters
    terminationEndOfSpeechDetected ( 4), -- optional parameters
    terminationSpeech    ( 5),  -- optional parameters
    privateData          ( 6),  -- optional parameters
    privateDataInAck    ( 7),  -- optional parameters
    multipleMsgsSimultaneously ( 8)} -- miscellaneous characteristics

QueryVoiceAttrib ::= BIT STRING
{
    attribToQueryEncodingAlgorithm ( 0),  -- optional parameters
    attribToQuerySamplingRate      ( 1),  -- optional parameters
    attribToQueryDuration          ( 2),  -- optional parameters
    attribToQueryFilename         ( 3),  -- optional parameters
    attribToQueryCurrentPosition  ( 4),  -- optional parameters
    attribToQueryCurrentSpeed     ( 5),  -- optional parameters
    attribToQueryCurrentVolume    ( 6),  -- optional parameters
    attribToQueryCurrentLevel     ( 7),  -- optional parameters
    attribToQueryCurrentState     ( 8),  -- optional parameters
    connection                    ( 9),  -- optional parameters
    duration                      (10),  -- optional parameters
    terminationParameter          (11),  -- optional parameters
    terminationDurationExceeded  (12),  -- optional parameters
    terminationDTMFDigitDetected (13),  -- optional parameters
    terminationEndOfSpeechDetected (14),  -- optional parameters
    terminationSpeech            (15),  -- optional parameters
    privateData                  (16),  -- optional parameters
    attribInAckEncodingAlgorithmADPCM6K (17), -- optional parameters
    attribInAckEncodingAlgorithmADPCM8K (18), -- optional parameters
    attribInAckEncodingAlgorithmMuLawPCM6K (19), -- optional parameters
    attribInAckEncodingAlgorithmALawPCM6K (20), -- optional parameters
    attribInAckSamplingRate      (21),  -- optional parameters
    attribInAckDuration          (22),  -- optional parameters
    attribInAckFilename         (23),  -- optional parameters
    attribInAckCurrentPosition  (24),  -- optional parameters
    attribInAckCurrentSpeed     (25),  -- optional parameters
    attribInAckCurrentVolumeAbs (26),  -- optional parameters
    attribInAckCurrentGain      (27),  -- optional parameters
    attribInAckCurrentState     (28),  -- optional parameters
    privateDataInAck           (29)}  -- optional parameters

RecordMsg ::= BIT STRING
{
    samplingRate          ( 0),  -- optional parameters
    encodingAlgorithmADPCM6K ( 1),  -- optional parameters
    encodingAlgorithmADPCM8K ( 2),  -- optional parameters
    encodingAlgorithmMuLawPCM6K ( 3),  -- optional parameters
    encodingAlgorithmALawPCM6K ( 4),  -- optional parameters
    maxDuration          ( 5),  -- optional parameters
    terminationParameter ( 6),  -- optional parameters
    terminationDurationExceeded ( 7),  -- optional parameters
    terminationDTMFDigitDetected ( 8),  -- optional parameters
    terminationEndOfDataDetected ( 9),  -- optional parameters
    terminationSpeechDetected (10),  -- optional parameters
    privateData          (11),  -- optional parameters
    privateDataInAck    (12)}  -- optional parameters

Reposition ::= BIT STRING
{
    periodOfRepositionStartOfMsg ( 0),  -- optional parameters

```

```

        periodOfRepositionEndOfMsg          ( 1),  -- optional parameters
        periodOfRepositionRelativePointer   ( 2),  -- optional parameters
        msgToReposition                     ( 3),  -- optional parameters
        privateData                         ( 4),  -- optional parameters
        privateDataInAck                    ( 5)}  -- optional parameters

Resume ::= BIT STRING
{
    msgToResume          ( 0),  -- optional parameters
    duration             ( 1),  -- optional parameters
    privateData          ( 2),  -- optional parameters
    privateDataInAck    ( 3)}  -- optional parameters

Review ::= BIT STRING
{
    periodToResumeStartOfMsg   ( 0),  -- optional parameters
    periodToResumeLengthOfReview ( 1),  -- optional parameters
    privateData                ( 2),  -- optional parameters
    privateDataInAck          ( 3)}  -- optional parameters

SetVoiceAttrib ::= BIT STRING
{
    currentSpeed          ( 0),  -- optional parameters
    currentVolumeAbs     ( 1),  -- optional parameters
    currentVolumeInc     ( 2),  -- optional parameters
    periodToResumeStartOfMsg ( 3),  -- optional parameters
    periodToResumeLengthOfReview ( 4),  -- optional parameters
    currentGain          ( 5),  -- optional parameters
    message              ( 6),  -- optional parameters
    privateData          ( 7),  -- optional parameters
    privateDataInAck    ( 8)}  -- optional parameters

Stop ::= BIT STRING
{
    privateData          ( 0),  -- optional parameters
    privateDataInAck    ( 1)}  -- optional parameters

Suspend ::= BIT STRING
{
    message              ( 0),  -- optional parameters
    privateData          ( 1),  -- optional parameters
    privateDataInAck    ( 2)}  -- optional parameters

SynthesizeMsg ::= BIT STRING
{
    genderMale          ( 0),  -- optional parameters
    genderFemale        ( 1),  -- optional parameters
    privateData         ( 2),  -- optional parameters
    privateDataInAck    ( 3)}  -- optional parameters

VoiceUnitEvtsList ::= SEQUENCE
{
    play                [ 0] IMPLICIT Play          OPTIONAL,
    record              [ 1] IMPLICIT Record        OPTIONAL,
    review              [ 2] IMPLICIT ReviewEvent    OPTIONAL,
    stop                [ 3] IMPLICIT StopEvent      OPTIONAL,
    suspendPlay        [ 4] IMPLICIT SuspendPlay    OPTIONAL,
    suspendRecord      [ 5] IMPLICIT SuspendRecord  OPTIONAL,
    voiceAttribChanged [ 6] IMPLICIT VoiceAttribChanged OPTIONAL}

Play ::= BIT STRING
{
    length              ( 0),  -- optional parameters
    currentPosition    ( 1),  -- optional parameters
    speed              ( 2),  -- optional parameters
    cause              ( 3),  -- optional parameters
    servicesPermitted ( 4),  -- optional parameters
    privateData        ( 5)}  -- optional parameters

Record ::= BIT STRING
{
    length              ( 0),  -- optional parameters
    currentPosition    ( 1),  -- optional parameters
    speed              ( 2),  -- optional parameters
    cause              ( 3),  -- optional parameters
    servicesPermitted ( 4),  -- optional parameters
    privateData        ( 5)}  -- optional parameters

```

```

ReviewEvent ::= BIT STRING
{
    length                ( 0), -- optional parameters
    currentPosition      ( 1), -- optional parameters
    cause                 ( 2), -- optional parameters
    servicesPermitted    ( 3), -- optional parameters
    privateData          ( 4)} -- optional parameters

StopEvent ::= BIT STRING
{
    length                ( 0), -- optional parameters
    currentPosition      ( 1), -- optional parameters
    speed                 ( 2), -- optional parameters
    cause                 ( 3), -- optional parameters
    servicesPermitted    ( 4), -- optional parameters
    privateData          ( 5)} -- optional parameters

SuspendPlay ::= BIT STRING
{
    length                ( 0), -- optional parameters
    currentPosition      ( 1), -- optional parameters
    cause                 ( 2), -- optional parameters
    servicesPermitted    ( 3), -- optional parameters
    privateData          ( 4)} -- optional parameters

SuspendRecord ::= BIT STRING
{
    length                ( 0), -- optional parameters
    currentPosition      ( 1), -- optional parameters
    cause                 ( 2), -- optional parameters
    servicesPermitted    ( 3), -- optional parameters
    privateData          ( 4)} -- optional parameters

VoiceAttribChanged ::= BIT STRING
{
    playVolumeAbs        ( 0), -- optional parameters
    playVolumeInc        ( 1), -- optional parameters
    recordGain           ( 2), -- optional parameters
    speed                ( 3), -- optional parameters
    currentPosition      ( 4), -- optional parameters
    cause                 ( 5), -- optional parameters
    privateData          ( 6)} -- optional parameters

CDRServList ::= SEQUENCE
{
    cdrNotification      [ 0] IMPLICIT CdrNotification      OPTIONAL,
    cdrReport            [ 1] IMPLICIT CdrReport            OPTIONAL,
    sendStoredCDRs       [ 2] IMPLICIT SendStoredCDRs       OPTIONAL,
    startCDRTransmission [ 3] IMPLICIT StartCDRTransmission OPTIONAL,
    stopCDRTransmission [ 4] IMPLICIT StopCDRTransmission  OPTIONAL }

CdrNotification ::= BIT STRING
{
    cdrReasonTimeout      ( 0), -- optional parameters
    cdrReasonThresholdReached ( 1), -- optional parameters
    cdrReasonOther        ( 2), -- optional parameters
    privateData           ( 3), -- optional parameters
    privateDataInAck      ( 4)} -- optional parameters

CdrReport ::= BIT STRING
{
    cdrReasonTimeout      ( 0), -- optional parameters
    cdrReasonThresholdReached ( 1), -- optional parameters
    cdrReasonOther        ( 2), -- optional parameters
    recordNumber          ( 3), -- optional parameter
    recordCreationTime    ( 4), -- optional parameter
    callingDevice         ( 5), -- optional parameter
    calledDevice          ( 6), -- optional parameter
    assocCallingDevice    ( 7), -- optional parameter
    assocCalledDevice     ( 8), -- optional parameter
    netwCallingDevice     ( 9), -- optional parameter
    netwCalledDevice      (10), -- optional parameter
    callCharacteristics   (11), -- optional parameter
    mediaCallCharacteristics (12), -- optional parameter
    chargedDeviceOperator (13), -- optional parameter

```

chargedDeviceNonOperator (14), -- optional parameters  
 recordedCall (15), -- optional parameters  
 nodeNumberArea0 (16), -- optional parameters  
 nodeNumberArea1 (17), -- optional parameters  
 nodeNumberArea2 (18), -- optional parameters  
 tarifTable (19), -- optional parameters  
 connectionStart (20), -- optional parameters  
 connectionEnd (21), -- optional parameters  
 connectionDuration (22), -- optional parameters  
 accessCode (23), -- optional parameters  
 carrier (24), -- optional parameters  
 selectedRoute (25), -- optional parameters  
 billingIndicatorNormalCharging (26), -- optional parameters  
 billingIndicatorReverseCharging (27), -- optional parameters  
 billingIndicatorCreditCardCharging (28), -- optional parameters  
 billingIndicatorCallForwarding (29), -- optional parameters  
 billingIndicatorCallDeflection (30), -- optional parameters  
 billingIndicatorCallTransfer (31), -- optional parameters  
 billingIndicatorOther (32), -- optional parameters  
 chargingInfo (33), -- optional parameters  
 suppServiceInfoNormalCall (34), -- optional parameters  
 suppServiceInfoConsultationCall (35), -- optional parameters  
 suppServiceInfoTransferCall (36), -- optional parameters  
 suppServiceInfoCallCompletion (37), -- optional parameters  
 suppServiceInfoCallForwarding (38), -- optional parameters  
 suppServiceInfoCallDiversion (39), -- optional parameters  
 suppServiceInfoConferencing (40), -- optional parameters  
 suppServiceInfoIntrusion (41), -- optional parameters  
 suppServiceInfoUserUserInfo (42), -- optional parameters  
 suppServiceInfoOther (43), -- optional parameters  
 reasonForTermNormalClearing (44), -- optional parameters  
 reasonForTermUnsuccessfulCallAttempt (45), -- optional parameters  
 reasonForTermAbnormalTermination (46), -- optional parameters  
 reasonForTermCallTransferred (47), -- optional parameters  
 reasonForTermOther (48), -- optional parameters  
 authCode (49), -- optional parameters  
 accountInfo (50), -- optional parameters  
 deviceCategory (51), -- optional parameters  
 namedDeviceTypes (52), -- optional parameters  
 operatorDevice (53), -- optional parameters  
 lastStoredCDRReportSent (54), -- optional parameters  
 privateData (55), -- optional parameters  
 privateDataInAck (56)} -- optional parameters

SendStoredCDRs ::= BIT STRING  
 {  
     timePeriod ( 0), -- optional parameters  
     privateData ( 1), -- optional parameters  
     privateDataInAck ( 2)} -- optional parameters

StartCDRTransmission ::= BIT STRING  
 {  
     transferModeTransferAtEndOfCall ( 0), -- optional parameters  
     transferModeTransferOnRequest ( 1), -- optional parameters  
     transferModeTransferOnThresholdReached ( 2), -- optional parameters  
     privateData ( 3), -- optional parameters  
     privateDataInAck ( 4)} -- optional parameters

StopCDRTransmission ::= BIT STRING  
 {  
     cdrTermReasonEndOfData ( 0), -- optional parameters  
     cdrTermReasonError ( 1), -- optional parameters  
     cdrTermReasonThresholdReached ( 2), -- optional parameters  
     cdrTermReasonOther ( 3), -- optional parameter  
     privateData ( 4), -- optional parameter  
     privateDataInAck ( 5), -- optional parameter  
     swFunctionSupportsSending ( 6), -- miscellaneous characteristics  
     swFunctionSupportsReceiving ( 7)} -- miscellaneous characteristics

VendorSpecificServList ::= SEQUENCE  
 {  
     escapeRegister [ 0] IMPLICIT EscapeRegister OPTIONAL,

```

        escapeRegisterCancel      [ 1] IMPLICIT EscapeRegisterCancel      OPTIONAL,
        escapeRegisterAbort      [ 2] IMPLICIT EscapeRegisterAbort      OPTIONAL,
        escape                    [ 3] IMPLICIT Escape                    OPTIONAL,
        privateDataVersionSelection [ 4] IMPLICIT PrivateDataVersionSelection OPTIONAL }

EscapeRegister ::= BIT STRING
{
    privateData          (0),      -- optional parameters
    privateDataInAck    (1)}      -- optional parameters

EscapeRegisterCancel ::= BIT STRING
{
    privateData          (0),      -- optional parameters
    privateDataInAck    (1)}      -- optional parameters

EscapeRegisterAbort ::= BIT STRING
{
    privateData          (0),      -- optional parameters
    privateDataInAck    (1)}      -- optional parameters

Escape ::= BIT STRING
{
    privateDataInAck      (0),      -- optional parameters
    swFunctionSupportsSending (1),  -- misc characteristics
    swFunctionSupportsReceiving (2)} -- misc characteristics

PrivateDataVersionSelection ::= BIT STRING
{
    privateDataInAck      (0)}      -- optional parameters

VendorSpecificEvtsList ::= SEQUENCE
{
    privateEvent [0] IMPLICIT PrivateEvent OPTIONAL}

PrivateEvent ::= BIT STRING

-- other Types

DeviceIDFormat ::= BIT STRING
{
    dialableDigitsAsterix      (0),
    dialableDigitsHash         (1),
    dialableDigitsABCD         (2),
    dialableDigitsExclamation  (3),
    dialableDigitsP            (4),
    dialableDigitsT            (5),
    dialableDigitsComma        (6),
    dialableDigitsW            (7),
    dialableDigitsAt           (8),
    dialableDigitsDollar       (9),
    dialableDigitsSemicolon    (10),
    sFReprExclamation          (11),
    sFReprEt                   (12),
    sFReprSlash                (13),
    sFReprPercent              (14),
    sFReprNM                   (15),
    sFReprGeneric              (16),
    sFReprImplicitTON          (17),
    sFReprPubTONUnkown        (18),
    sFReprPubTONInternal      (19),
    sFReprPubTONNational      (20),
    sFReprPubTONSubscriber    (21),
    sFReprPubTONAbbreviated    (22),
    sFReprPriTONUnknown       (23),
    sFReprPriTONLevel3        (24),
    sFReprPriTONLevel2        (25),
    sFReprPriTONLevel1        (26),
    sFReprPriTONLocal         (27),
    sFReprPriTONAbbreviated    (28),
    sFReprOther                (29),
    deviceNumber               (30)}

SwDomainFeatures ::= BIT STRING

```

```

{
    isForwardingBefore           ( 0),
    isForwardingAfter           ( 1),
    swFunctionDefaultSettings    ( 2),
    userSpecific                 ( 3),
    userSpecificDefaultFowardingType ( 4),
    userSpecificDefaultForwardDestination ( 5),
    negativeAcknowledgment      ( 6),
    supportFailedWithAssConn     ( 7),
    supportFailedWithoutAssConn  ( 8),
    supportFailedWithAssConnNotReportet ( 9),
    recall                       (10),
    callBack                    (11),
    extCallsIncoming            (12),
    extCallsOutgoing            (13),
    prompting                    (14)}

```

SwAppearanceAddressability ::= BIT STRING

```

{
    nonAddressable (0),
    addressable   (1)}

```

SwAppearanceTypes ::= BIT STRING

```

{
    selectedStandard           (0),
    basicStandard             (1),
    basicBridged              (2),
    exclusiveBridged          (3),
    independentSharedBridged  (4),
    interDependentSharedBridged (5)}

```

IgnoreUnsupportedParameters ::= ENUMERATED

```

{
    ignoreParameters (0),
    rejectMessage    (1)}

```

PauseTime ::= INTEGER (1..2000)

TimeStampMode ::= BIT STRING

```

{
    allEvents (0),
    allAcks   (1),
    allServReqs (2)}

```

MiscMonitorCaps ::= BIT STRING

```

{
    groupInclusivModel (0),
    groupExclusiveModel (1),
    monitorPhysicalElement (2),
    acdDeviceInclusiv (3),
    acdDeviceExclusiv (4)
}

```

MaxLengthParameters ::= SEQUENCE

```

{
    accountInfo           INTEGER (0..32),
    authCode              INTEGER (0..32),
    agentID               INTEGER (0..32),
    agentPassword         INTEGER (0..32),
    callIDInConnectionID INTEGER (0..8),
    correlatorData       INTEGER (0..32),
    privateData           INTEGER,
    deviceIdentifiers     INTEGER (0..128),
    userData              INTEGER (0..256),
    buttonLabel           INTEGER (0..64),
    lampLabel             INTEGER (0..64),
    charactersToSend     INTEGER (0..64)}

```

FilterThreshold ::= SEQUENCE

```

{
    getLogicalDeviceInformation    INTEGER,
    getPhysicalDeviceInformation   INTEGER,
    getSwitchingFunctionCaps      INTEGER,
    getSwitchingFunctionDevices   INTEGER,
    switchingFunctionDevices      INTEGER,
    changeSystemStatusFilter      INTEGER,

```

systemStatusRegister	INTEGER,
systemStatusRegisterAbort	INTEGER,
systemStatusRegisterCancel	INTEGER,
requestSystemStatus	INTEGER,
systemStatus	INTEGER,
switchingFunctionCapsChanged	INTEGER,
switchingFunctionDevsChanged	INTEGER,
changeMonitorFilter	INTEGER,
monitorStart	INTEGER,
monitorStop	INTEGER,
snapshotCall	INTEGER,
snapshotDevice	INTEGER,
snapshotCallData	INTEGER,
snapshotDeviceData	INTEGER,
acceptCall	INTEGER,
alternateCall	INTEGER,
answerCall	INTEGER,
callBackCallRelated	INTEGER,
callBackMessageCallRelated	INTEGER,
campOnCall	INTEGER,
clearCall	INTEGER,
clearConnection	INTEGER,
conferenceCall	INTEGER,
consultationCall	INTEGER,
deflectCall	INTEGER,
dialDigits	INTEGER,
directedPickupCall	INTEGER,
groupPickupCall	INTEGER,
holdCall	INTEGER,
intrudeCall	INTEGER,
joinCall	INTEGER,
makeCall	INTEGER,
makePredictiveCall	INTEGER,
parkCall	INTEGER,
reconnectCall	INTEGER,
retrieveCall	INTEGER,
singleStepConferenceCall	INTEGER,
singleStepTransferCall	INTEGER,
transferCall	INTEGER,
associateData	INTEGER,
cancelTelephonyTone	INTEGER,
generateDigits	INTEGER,
generateTelephonyTone	INTEGER,
sendUserInfo	INTEGER,
startDTMFDigitsCollection	INTEGER,
startTelephonyTonesCollection	INTEGER,
stopDTMFDigitsCollection	INTEGER,
stopTelephonyTonesCollection	INTEGER,
attachMediaService	INTEGER,
detachMediaService	INTEGER,
routeRegister	INTEGER,
routeRegisterAbort	INTEGER,
routeRegisterCancel	INTEGER,
reRoute	INTEGER,
routeEnd	INTEGER,
routeReject	INTEGER,
routeRequest	INTEGER,
routeSelect	INTEGER,
routeUsed	INTEGER,
buttonPress	INTEGER,
getAuditoryApparatusInfo	INTEGER,
getButtonInformation	INTEGER,
getDisplay	INTEGER,
getHookswitchStatus	INTEGER,
getLampInformation	INTEGER,
getLampMode	INTEGER,
getMicrophoneGain	INTEGER,
getMicrophoneMute	INTEGER,

STANDARD 360.COM : Click to view the full PDF of ISO/IEC 18052:2000

getMessageWaitingIndicator	INTEGER,
getRingerStatus	INTEGER,
getSpeakerMute	INTEGER,
getSpeakerVolume	INTEGER,
setButtonInformation	INTEGER,
setDisplay	INTEGER,
setHookswitchStatus	INTEGER,
setLampMode	INTEGER,
setMsgWaitingIndicator	INTEGER,
setMicrophoneGain	INTEGER,
setMicrophoneMute	INTEGER,
setRingerStatus	INTEGER,
setSpeakerMute	INTEGER,
setSpeakerVolume	INTEGER,
callBackMessageNonCallRelated	INTEGER,
callBackNonCallRelated	INTEGER,
cancelCallBack	INTEGER,
cancelCallBackMessage	INTEGER,
getAgentState	INTEGER,
getAutoAnswer	INTEGER,
getAutoWorkMode	INTEGER,
getCallerIDStatus	INTEGER,
getDoNotDisturb	INTEGER,
getForwarding	INTEGER,
getLastNumberDialed	INTEGER,
getRouteingMode	INTEGER,
setAgentState	INTEGER,
setAutoAnswer	INTEGER,
setAutoWorkMode	INTEGER,
setCallerIDStatus	INTEGER,
setDoNotDisturb	INTEGER,
setForwarding	INTEGER,
setRouteingMode	INTEGER,
backInService	INTEGER,
deviceCaosChanged	INTEGER,
outOfService	INTEGER,
ioRegister	INTEGER,
ioRegisterAbort	INTEGER,
ioRegisterCancel	INTEGER,
dataPathResumed	INTEGER,
dataPathSuspended	INTEGER,
fastData	INTEGER,
resumeDataPath	INTEGER,
sendBroadcastData	INTEGER,
sendData	INTEGER,
sendMulticastData	INTEGER,
startDataPath	INTEGER,
stopDataPath	INTEGER,
suspendDataPath	INTEGER,
concatenateMsg	INTEGER,
deleteMsg	INTEGER,
playMsg	INTEGER,
queryVoiceAttribute	INTEGER,
recordMsg	INTEGER,
reposition	INTEGER,
resume	INTEGER,
review	INTEGER,
setVoiceAttribute	INTEGER,
stop	INTEGER,
suspend	INTEGER,
synthesizeMsg	INTEGER,
cDRNotification	INTEGER,
cDRReport	INTEGER,
sendStoredCDRs	INTEGER,
startCDRTransmission	INTEGER,
stopCDRTransmission	INTEGER,
escapeRegister	INTEGER,
escapeRegisterAbort	INTEGER,

STANDARD.PDFISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

```

        escapeRegisterCancel      INTEGER,
        escape                    INTEGER,
        privateDataVersion        INTEGER}

MediaServiceCapsList ::= SEQUENCE OF SEQUENCE
{
    mediaServiceType      MediaServiceType,
    mediaServiceVersion   INTEGER
    mediaServiceInstance  MediaServiceInstanceID OPTIONAL,
    connectionMode       ConnectionModeBMap   OPTIONAL,
    mediaStreamIDSupported BOOLEAN}

DeviceCategory ::= ENUMERATED
{
    acd                ( 0 ),
    group              ( 1 ),
    networkInterface   ( 2 ),
    park               ( 3 ),
    routeingDevice     ( 4 ),
    station            ( 5 ),
    voiceUnit          ( 6 ),
    other              ( 7 )}

GroupDeviceAttributes ::= BIT STRING
{
    acd      ( 0 ),
    hunt    ( 1 ),
    pick    ( 2 ),
    other   ( 3 )}

NamedDeviceTypes ::= ENUMERATED
{
    acd                ( 0 ),
    acdGroup           ( 1 ),
    button             ( 2 ),
    buttonGroup        ( 3 ),
    conferenceBridge   ( 4 ),
    line               ( 5 ),
    lineGroup          ( 6 ),
    operator           ( 7 ),
    operatorGroup      ( 8 ),
    parkingDevice      ( 9 ),
    station            (10),
    stationGroup       (11),
    trunk              (12),
    trunkGroup         (13),
    other              (14),
    otherGroup         (15)}

ACDModels ::= BIT STRING
{
    visibleACDRelatedDevices ( 0 ),
    nonVisibleACDRelatedDevices ( 1 )}

AgentLogOnModels ::= BIT STRING
{
    logOnToACDDevice ( 0 ),
    logOnToACDGroupExplOneStep ( 1 ),
    logOnToACDGroupExplTwoSteps ( 2 ),
    logOnToACDImplOneStep ( 3 )}

AppearanceType ::= ENUMERATED
{
    selectedStandard ( 0 ),
    basicStandard ( 1 ),
    basicBridged ( 2 ),
    exclusiveBridged ( 3 ),
    independentSharedBridged ( 4 ),
    interdependentSharedBridged ( 5 )}

TransAndConfSetup ::= BIT STRING
{
    consultationCall ( 0 ),
    holdCallMakeCall ( 1 ),
    alternateCall ( 2 ),
    twoCallsInHold ( 3 ),

```

```
twoCallsInConnected ( 4)}
```

```
END -- of CSTA-capability-exchange
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.11 Call Detail Record

```

CSTA-call-detail-record
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) call-detail-record(132) }

DEFINITIONS ::=
BEGIN

EXPORTS
CDRCrossRefID, CDRInfo, CDRReason, CDRTermReason, CDRTIMEPERIOD,
CDRTransferMode;

IMPORTS
DeviceID, CalledDeviceID, CallingDeviceID, AssociatedCalledDeviceID,
AssociatedCallingDeviceID, NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) device-identifiers(123) }
ConnectionID FROM CSTA-call-connection-identifiers
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) call-connection-identifiers(124) }
AccountInfo, AuthCode FROM CSTA-device-feature-types
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) device-feature-types(127) }
TimeInfo FROM CSTA-security
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) security(128) }
CallCharacteristics FROM CSTA-call-control
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) call-control(130) }
DeviceCategory, NamedDeviceTypes FROM CSTA-capability-exchange
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) capability-exchange(131) }
ChargingInfo FROM CSTA-charge-info
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) charge-info(133) }
MediaCallCharacteristics FROM CSTA-media-services
{ iso(1) identified-organization(3) icd-ecma(12)
  standard(0) csta3(285) media-services(136) }
;

CDRCrossRefID ::= OCTET STRING (SIZE(0..4))

CDRReason ::= ENUMERATED
{
  timeout                (0),
  thresholdReached      (1),
  other                  (2)}

CDRInfo ::= SEQUENCE OF CDRInformationItem

CDRInformationItem ::= SEQUENCE
{
  recordNumber           INTEGER (1..128)           OPTIONAL,
  recordCreationTime     TimeInfo,
  callingDevice          CallingDeviceID           OPTIONAL,
  calledDevice          CalledDeviceID            OPTIONAL,
  associatedCallingDevice AssociatedCallingDeviceID OPTIONAL,
  associatedCalledDevice AssociatedCalledDeviceID  OPTIONAL,
  networkCallingDevice  NetworkCallingDeviceID    OPTIONAL,
  networkCalledDevice   NetworkCalledDeviceID     OPTIONAL,
  callCharacteristics   [0] IMPLICIT CallCharacteristics OPTIONAL,
  mediaCallCharacteristics [1] IMPLICIT MediaCallCharacteristics OPTIONAL,
  chargedDevice        [2] ChargedDevice          OPTIONAL,
  recordedCall         ConnectionID               OPTIONAL,
  nodeNumber           [3] IMPLICIT NodeNumber    OPTIONAL,
  tariffTable          [4] IMPLICIT INTEGER       OPTIONAL,
  connectionStart      [5] IMPLICIT TimeInfo     OPTIONAL,
  connectionEnd        [6] IMPLICIT TimeInfo     OPTIONAL,
}

```

```

connectionDuration      [7] IMPLICIT INTEGER          OPTIONAL,
accessCode              [8] IMPLICIT OCTET STRING  OPTIONAL,
carrier                 [9] IMPLICIT INTEGER       OPTIONAL,
selectedRoute           [10] IMPLICIT INTEGER      OPTIONAL,
billingID               [11] IMPLICIT BillingID     OPTIONAL,
chargingInfo            [12] IMPLICIT ChargingInfo  OPTIONAL,
supplServiceInfo       [13] IMPLICIT SupplServiceInfo  OPTIONAL,
reasonForTerm          [14] IMPLICIT ReasonForTerm  OPTIONAL,
authCode                [15] IMPLICIT AuthCode     OPTIONAL,
accountInfo             [16] IMPLICIT AccountInfo   OPTIONAL,
deviceCategory         [17] IMPLICIT DeviceCategory  OPTIONAL,
namedDeviceTypes       [18] IMPLICIT NamedDeviceTypes  OPTIONAL,
operatorDevice          [19] DeviceID              OPTIONAL}

ChargedDevice ::= CHOICE
{
    operator          [0] IMPLICIT DeviceID,
    nonOperator       [1] IMPLICIT DeviceID}

NodeNumber ::= SEQUENCE
{
    area0  [0] IMPLICIT INTEGER OPTIONAL,
    area1  [1] IMPLICIT INTEGER OPTIONAL,
    area2  [2] IMPLICIT INTEGER OPTIONAL}

BillingID ::= ENUMERATED
{
    normalCharging          (0),
    reverseCharging         (1),
    creditCardCharging      (2),
    callForwarding          (3),
    callDeflection          (4),
    callTransfer            (5),
    other                   (6)}

SupplServiceInfo ::= BIT STRING
{
    normalCall              (0),
    consultationCall        (1),
    transferCall            (2),
    callCompletion          (3),
    callForwarding          (4),
    callDiversion           (5),
    conferencing            (6),
    intrusion               (7),
    userUserInfo            (8),
    other                   (9)}

ReasonForTerm ::= ENUMERATED
{
    normalClearing          (0),
    unsuccessfulCallAttempt (1),
    abnormalTermination     (2),
    callTransferred         (3),
    other                   (4)}

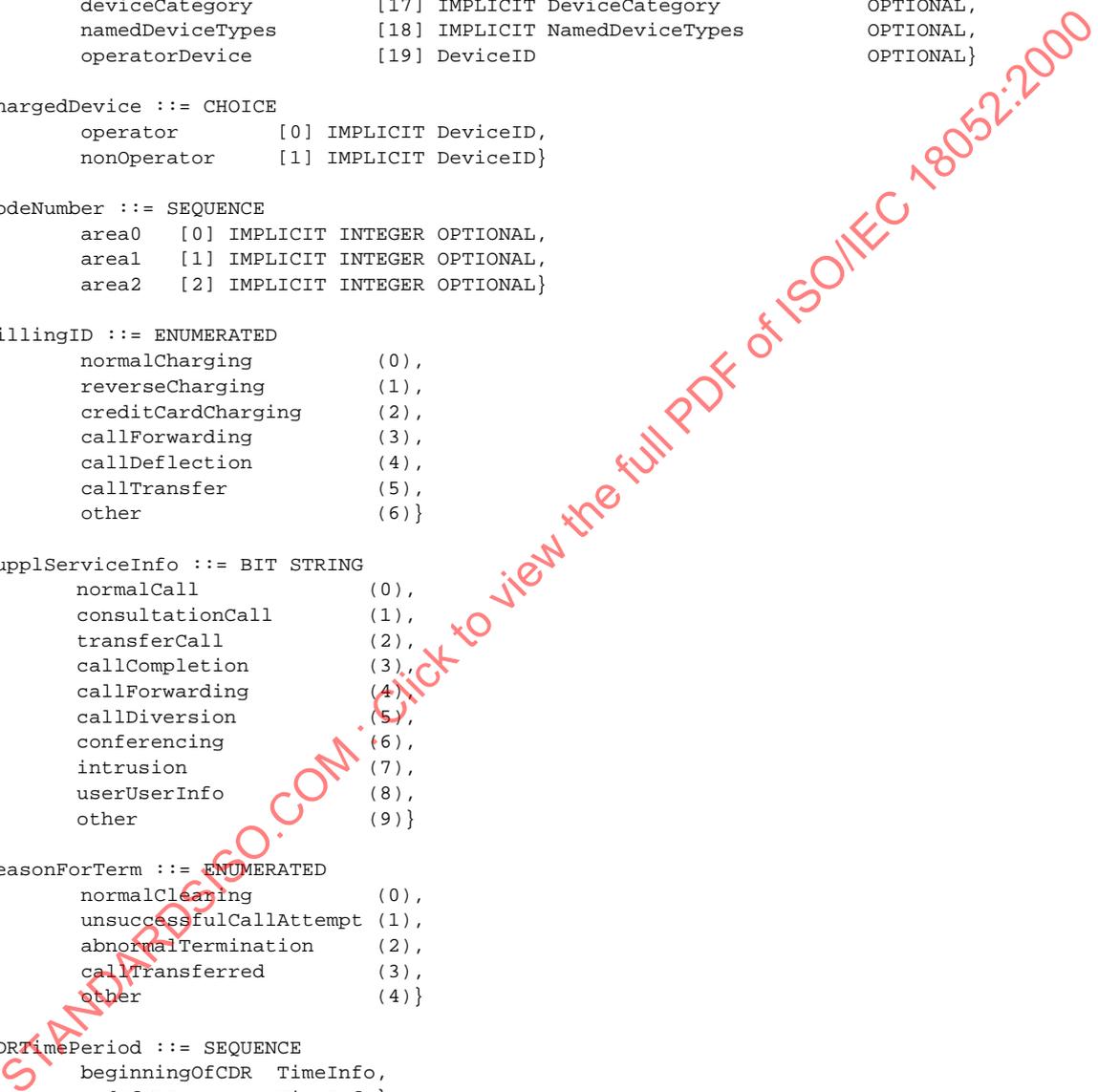
CDRTimePeriod ::= SEQUENCE
{
    beginningOfCDR  TimeInfo,
    endOfCDR        TimeInfo}

CDRTransferMode ::= ENUMERATED
{
    transferAtEndOfCall      (0),
    transferOnRequest        (1),
    transferOnThresholdReached (2)}

CDRTermReason ::= ENUMERATED
{
    endOfDataDetected      (0),
    errorDetected          (1),
    thresholdReached       (2),
    other                   (3)}

END -- of CSTA-call-detail-record

```



## 9.12 Charge information

```

CSTA-charge-info
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) charge-info( 133) }

DEFINITIONS ::=
BEGIN

EXPORTS
ChargingInfo;

ChargingInfo ::= SEQUENCE
  {
    numberUnits          NumberUnits,
    typeOfChargingInfo  ENUMERATED
    {
      subTotal          (0),
      total             (1)
    }
  }

NumberUnits ::= CHOICE
  {
    numberOfChargeUnits [0] IMPLICIT NumberOfChargingUnits,
    numberOfCurrencyUnits [1] IMPLICIT NumberOfCurrencyUnits
  }

NumberOfChargingUnits ::= SEQUENCE OF SEQUENCE
  {
    chargingUnits  INTEGER,
    typeOfUnits   OCTET STRING OPTIONAL }

NumberOfCurrencyUnits ::= SEQUENCE
  {
    currencyType      OCTET STRING,          -- size 0 indicates default currency
    currencyAmount    INTEGER,
    currencyMultiplier  ENUMERATED
    {
      oneThousandth   (0),
      oneHundredth    (1),
      oneTenth        (2),
      one              (3),
      ten              (4),
      hundred          (5),
      thousand        (6)
    }
  }
}

END -- of CSTA-charge-info

```

### 9.13 Data call types

```
CSTA-data-call-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-call-types( 134) }

DEFINITIONS ::=
BEGIN

EXPORTS
NumberOfChannels, MaxChannelBind, ConnectionRateList, DelayToleranceList;

NumberOfChannels      ::= INTEGER

MaxChannelBind        ::= INTEGER

ConnectionRateList ::= SEQUENCE OF INTEGER

DelayToleranceList ::= SEQUENCE OF INTEGER

END      -- of CSTA-data-call-types
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.14 Escape types

```
CSTA-escape-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) escape-types( 135) }

DEFINITIONS ::=
BEGIN
EXPORTS
EscapeRegisterID;

EscapeRegisterID ::= [0] OCTET STRING (SIZE(0..4))

END -- of CSTA-escape-types
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.15 Media services

```

CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) }

DEFINITIONS ::=
BEGIN

EXPORTS
ConnectionInformation, ConnectionMode, ConnectionModeBMap, MediaCallCharacteristics,
MediaServiceInstanceID, MediaServiceType, MediaStreamID, MediaClass;

MediaServiceType ::= ENUMERATED
{
    cstaVoiceUnit                ( 0),
    dataModem                    ( 1),
    digitalDataIsochronousIeee1394 ( 2),
    digitalDataIsochronousGeoport ( 3),
    digitalDataIsochronousIeeeAtm  ( 4),
    digitalDataIsochronousIeeeIsdn ( 5),
    digitalDataApi                ( 6),
    ectfS100MediaServicesDefault   ( 7),
    ectfS100MediaServicesAppServices ( 8),
    cstaIVRScript1                ( 9),
    cstaIVRScript2                (10),
    cstaIVRScript3                (11),
    cstaIVRScript4                (12),
    cstaIVRScript5                (13),
    cstaIVRScript6                (14),
    cstaIVRScript7                (15),
    cstaIVRScript8                (16),
    cstaIVRScript9                (17),
    cstaIVRScript10               (18),
    liveSoundCaptureAnalog        (19),
    liveSoundTransmitAnalog        (20),
    liveSoundCaptureIeee1394      (21),
    liveSoundTransmitIeee1394     (22),
    liveSoundCaptureTransmitGeoport (23),
    liveSoundCaptureTransmitAtm    (24),
    liveSoundCaptureTransmitISDN   (25),
    soundCaptureTransmitADPCM      (26),
    soundCaptureTransmitApi        (27),
    usb                            (28),
    sfSpecific1                   (29),
    sfSpecific2                   (30),
    sfSpecific3                   (31),
    sfSpecific4                   (32),
    sfSpecific5                   (33),
    sfSpecific6                   (34),
    sfSpecific7                   (35),
    sfSpecific8                   (36),
    sfSpecific9                   (37),
    sfSpecific10                  (38)}

MediaStreamID ::= OCTET STRING
MediaServiceInstanceID ::= OCTET STRING (SIZE(0..64))

ConnectionInformation ::= SEQUENCE
{
    flowDirection                ENUMERATED
    {
        transmit                (0),
        receive                  (1),
        transmitAndReceive      (2)
    } OPTIONAL,
    numberOfChannels             INTEGER    DEFAULT 1 }

ConnectionMode ::= ENUMERATED
{
    consultationConference        (0),
    consultationConferenceHold    (1),

```

```

deflect (2),
directedPickup (3),
join (4),
singleStepConference (5),
singleStepConferenceHold (6),
singleStepTransfer (7),
transfer (8),
direct (9)}

ConnectionModeBMap ::= BIT STRING
{
    consultationConference (0),
    consultationConferenceHold (1),
    deflect (2),
    directedPickup (3),
    join (4),
    singleStepConference (5),
    singleStepConferenceHold (6),
    singleStepTransfer (7),
    transfer (8),
    direct (9)}

MediaCallCharacteristics ::= SEQUENCE
{
    mediaClass MediaClass,
    connectionRate [0] IMPLICIT INTEGER OPTIONAL,
    -- value 0 indicates that
    -- the connection rate is
    -- unknown
    bitRate [1] IMPLICIT ENUMERATED
        {
            constant (0),
            variable (1)} DEFAULT constant,
    delayTolerance [2] IMPLICIT INTEGER OPTIONAL,
    switchingSubDomainCCIEType [3] IMPLICIT ENUMERATED {
        isdn (0),
        atm (1),
        isoEthernet (2),
        rsvp (3),
        other (4)
    } OPTIONAL,
    switchingSubDomainInformationElements OCTET STRING OPTIONAL
    -- is mandatory, if the switchingSubDomainCCIEType is present,
    -- should be ignored otherwise
}

MediaClass ::= BIT STRING
{
    voice (0),
    data (1),
    image (2),
    audio (4),
    other (3),
    notKnown (5)}

END -- of CSTA-media-services

```

## 9.16 Physical device features

```

CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) }

DEFINITIONS ::=
BEGIN

EXPORTS
    AuditoryApparatusID, AuditoryApparatusList, ButtonID, CharacterSet, DisplayID, LampBrightness,
    LampColor, LampID, LampMode, MicGainAbs, MicrophoneGain, RingerID, RingMode,
    VolAbs, Volume, HookswitchID;

AuditoryApparatusList ::= SEQUENCE OF
    SEQUENCE {
        auditoryApparatus AuditoryApparatusID,
        auditoryApparatusType ENUMERATED {
            speakerphone (0),
            handset (1),
            headset (2),
            speakerOnlyPhone (3),
            other (4)
        },
        speaker BIT STRING {
            present (0),
            volumeSettable (1),
            volumeReadable (2),
            muteSettable (3),
            muteReadable (4)
        },
        microphone BIT STRING {
            present (0),
            gainSettable (1),
            gainReadable (2),
            muteSettable (3),
            muteReadable (4)
        },
        hookswitch BIT STRING {
            hookswitchSettable (0),
            hookswitchOnHook (1)
        },
        hookswitchID HookswitchID
    }

AuditoryApparatusID ::= OCTET STRING (SIZE(0..4))

ButtonID ::= OCTET STRING (SIZE(0..4))

CharacterSet ::= ENUMERATED
    {
        ascii (0),
        unicode (1),
        proprietary (2)
    }

DisplayID ::= OCTET STRING (SIZE(0..4))

HookswitchID ::= OCTET STRING (SIZE(0..4))

LampBrightness ::= ENUMERATED
    {
        unspecified (0),
        dim (1),
        bright (2)
    }

```

```

LampColor ::= INTEGER
{
    noColor (0),
    red      (1),
    yellow   (2),
    green    (3),
    blue     (4),
    unknown  (5)} (0..100)

LampID ::= OCTET STRING (SIZE(0..4))

LampMode ::= INTEGER
{
    brokenFlutter (0),
    flutter       (1),
    off           (2),
    steady       (3),
    wink         (4),
    unknown      (5)} (0..100)

MicrophoneGain ::= CHOICE
{
    micGainAbs    MicGainAbs,
    micGainInc    MicGainInc}

MicGainInc ::= ENUMERATED
{
    increment (0),
    decrement (1)}

MicGainAbs ::= INTEGER (0..100)

RingerID ::= OCTET STRING (SIZE(0..4))

RingMode ::= ENUMERATED
{
    ringing      (0),
    notRinging   (1)}

Volume ::= CHOICE
{
    volAbs  VolAbs,
    volInc  VolInc}

VolInc ::= ENUMERATED
{
    increment (0),
    decrement (1)}

VolAbs ::= INTEGER (0..100)

END -- of CSTA-physical-device-feature

```

STANDARDS150.COM: Click to view the full PDF of ISO/IEC 18052:2000

## 9.17 Data Collection

```
CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) }

DEFINITIONS ::=
BEGIN

EXPORTS
DcollCrossRefID;

DcollCrossRefID ::= OCTET STRING (SIZE(0..4))

END -- of CSTA-data-collection
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000



```

networkSignal                (46),
newCall                      (22),
nextMessage                  (47),
noAvailableAgents           (23),
normal                       (78),
normalClearing              (48),
noSpeechDetected            (49),
notAvaliableBearerService   (79),
notSupportedBearerService   (80),
numberChanged               (50),
numberUnallocated           (81),
overflow                    (26),
override                    (24),
park                        (25),
queueCleared                (82),
recall                      (27),
redirected                  (28),
remainsInQueue              (83),
reorderTone                 (29),
reserved                    (84),
resourcesNotAvailable       (30),
selectedTrunkBusy           (85),
silentParticipation         (31),
singleStepConference        (51),
singleStepTransfer          (52),
speechDetected              (53),
suspend                     (86),
switchingFunctionTerminated (54),
terminationCharacterReceived (55),
timeout                     (56),
transfer                    (32),
trunksBusy                  (33),
unauthorisedBearerService   (87)}
-- voiceUnitInitiator       (34) }
-- *** note that the voiceUnitOriginator (34) is no longer used (commented out) ***

END -- of event-cause-definitions.

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 9.19 Error Value

```

CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DEFINITIONS ::=
BEGIN

EXPORTS
UniversalFailure, universalFailure, ErrorValue;

IMPORTS

ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) };

universalFailure ERROR ::=
{
    PARAMETER      UniversalFailure
    CODE           local:1
}

ErrorValue ::= UniversalFailure

UniversalFailure ::= CHOICE
{
    operation                [0] OperationErrors,
    security                 [1] SecurityErrors,
    stateIncompatibility     [2] StateIncompatibilityErrors,
    systemResourceAvailability [3] SystemResourceAvailabilityErrors,
    subscribedResourceAvailability [4] SubscribedResourceAvailabilityErrors,
    performanceManagement    [5] PerformanceManagementErrors,
    privateData              [6] PrivateDataInfoErrors,
    unspecified              [7] UnspecifiedErrors}

OperationErrors ::= ENUMERATED -- in CSTA2 added in CSTA3
{
    generic ( 1),
    atLeastOneConditionalParameterNotProvided (29),
    featureAlreadySet (30),
    invalidMsgID (25),
    invalidParameterValue (31),
    invalidAccountCode (21),
    invalidAgentGroup (32),
    invalidAgentIdentifier (33),
    invalidAgentPassword (34),
    invalidAgentState (35),
    invalidAlertTime (36),
    invalidAllocationState (16),
    invalidAuthCode (22),
    invalidAutoAnswer (37),
    invalidBitRate (38),
    invalidButtonIdentifier (39),
    invalidCallType (42),
    invalidConnectionRate (43),
    invalidConsultPurpose (44),
    invalidCorrelatorData (20),
    invalidCrossRefID (17),
    invalidDelayTolerance (45),
    invalidDestination (14),
    invalidDestinationDetect (46),
    invalidDoNotDisturb (47),
    invalidEscapeCrossRefID (48),
    invalidFeature (15),
    invalidFile (28),
    invalidFlowDirection (49),
    invalidForwardingDestination ( 7),
    invalidForwardingFlag (51),
    invalidForwardingType (52),

```

invalidHookswitchType	(53),
invalidHookswitchComponent	(54),
invalidLampMode	(55),
invalidLampID	(56),
invalidMessageWaitingSetting	(57),
invalidMicrophoneGain	(58),
invalidMicrophoneMute	(59),
invalidMonitorCrossRefID	(60),
invalidMonitorFilter	(61),
invalidMonitorObject	(62),
invalidMonitorType	(63),
invalidNumberOFChannels	(64),
invalidParticipationType	(65),
invalidRemainRetry	(66),
invalidRingCount	(67),
invalidRingPattern	(68),
invalidRingVolume	(69),
invalidRouteingAlgorithm	(70),
invalidRouteingCrossRefID	(71),
invalidRouteRegistrationCrossRefID	(72),
invalidSpeakerVolume	(73),
invalidSpeakerMute	(74),
invalidSwitchingSubdomainCharsType	(75),
invalidObjectType	(18),
invalidActiveCallObject	(76),
invalidCalledDeviceObjectType	(77),
invalidCallingDeviceObjectType	(78),
invalidCallToBePickedUpObjectType	(79),
invalidCallToDivertObjectType	(80),
invalidCallToParkObjectType	(81),
invalidDestinationDeviceObject	(195),
invalidHeldCallObject	(82),
invalidMonitorObjectType	(83),
invalidParkToObjectType	(84),
messageIDRequired	(26),
notDifferentDevices	(85),
notSameDevice	(86),
objectNotKnown	( 4),
invalidCallID	(11),
invalidActiveCallID	(87),
invalidHeldCallID	(88),
invalidConnectionID	(13),
invalidActiveConnectionID	(89),
invalidHeldConnectionID	(90),
invalidDeviceID	(12),
invalidActiveDeviceID	(91),
invalidCalledDeviceID	( 6),
invalidCallingDeviceID	( 5),
invalidCallToParkDeviceID	(92),
invalidDestinationDeviceID	(93),
invalidDivertingDeviceID	(94),
invalidHeldDeviceID	(95),
invalidParkToDeviceID	(96),
invalidPickUpDeviceID	(97),
parameterNotSupported	(98),
accountCodeNotSupported	(99),
agentGroupNotSupported	(100),
agentPasswordNotSupported	(101),
agentStateNotSupported	(102),
alertTimeNotSupported	(103),
allocationNotSupported	(104),
authorisationCodeNotSupported	(105),
autoAnswerNotSupported	(106),
bitRateNotSupported	(107),
buttonNotSupported	(108),
callTypeNotSupported	(109),
charactersToSendNotSupported	(110),
connectionRateNotSupported	(111),

STANDARD PDF.COM : Click to view the full PDF of ISO/IEC 18052:2000

connectionReservationNotSupported	(112),
consultPurposeNotSupported	(113),
correlatorDataNotSupported	(114),
delayToleranceNotSupported	(115),
destinationDetectNotSupported	(116),
digitModeNotSupported	(117),
errorValueNotSupported	(118),
flowDirectionNotSupported	(119),
forwardingDestinationNotSupported	(120),
lampNotSupported	(121),
monitorTypeNotSupported	(122),
numberOfChannelsNotSupported	(123),
parameterTypeNotSupported	(124),
priorityNotSupported	(125),
privateDataNotSupported	(126),
pulseDurationNotSupported	(127),
pulseRateNotSupported	(128),
remainRetryNotSupported	(129),
ringCountNotSupported	(130),
routeUsedNotSupported	(131),
securityNotSupported	(132),
swSubdomainCCIETypeNotSupported	(133),
toneDurationNotSupported	(134),
sysStatRegIDNotSupported	(135),
userDataNotSupported	(136),
privilegeViolationSpecifiedDevice( 8),	
privilegeViolationActiveDevice	(137),
privilegeViolationCalledDevice( 9),	
privilegeViolationCallingDevice(10),	
privilegeViolationCallToParkDevice	(138),
privilegeViolationDestinationDevice	(139),
privilegeViolationOnDivertingDevice	(140),
privilegeViolationHeldDevice	(141),
privilegeViolationOnParkToDevice	(142),
privilegeViolationPickUpDevice	(143),
routeingTimerExpired	(144),
requestIncompatibleWithObject	( 2),
requestIncompatibleWithConnection	(145),
requestIncompatibleWithActiveConnection	(146),
requestIncompatibleWithHeldConnection	(147),
requestIncompatibleWithDevice	(148),
requestIncompatibleWithCalledDevice	(24),
requestIncompatibleWithCallingDevice	(23),
requestIncompatibleWithSubjectDevice	(149),
requestIncompatibleWithActiveDevice	(150),
requestIncompatibleWithCallToParkDevice	(151),
requestIncompatibleWithDestinationDevice	(152),
requestIncompatibleWithDivertingDevice	(153),
requestIncompatibleWithHeldDevice	(154),
requestIncompatibleWithMedia	(27),
requestIncompatibleWithParkToDevice	(155),
requestIncompatibleWithPickupDevice	(156),
serviceNotSupported	(50),
securityViolation	(19),
valueOutOfRange	( 3),
agentStateOutOfRange	(157),
alertTimeOutOfRange	(158),
allocationOutOfRange	(159),
autoAnswerOutOfRange	(160),
bitRateOutOfRange	(161),
callTypeOutOfRange	(162),
connectionRateOutOfRange	(163),
connectionReservoationOutOfRange	(164),
consultPurposeOutOfRange	(165),
correlatorDataOutOfRange	(166),
delayToleranceOutOfRange	(167),
destinationDetectOutOfRange	(168),
digitModeOutOfRange	(169),

doNotDisturbOutOfRange	(170),
flowDirectionOutOfRange	(171),
forwardingFlagOutOfRange	(172),
forwardingTypeOutOfRange	(173),
hookswitchComponentOutOfRange	(174),
hookswitchTypeOutOfRange	(175),
lampModeOutOfRange	(176),
messageWaitingSettingOutOfRange	(177),
micGainOutOfRange	(178),
micMuteOutOfRange	(179),
monitorTypeOutOfRange	(180),
numberOfChannelsOutOfRange	(181),
participationTypeOutOfRange	(182),
pulseDurationOutOfRange	(183),
pulseRateOutOfRange	(184),
ringCountOutOfRange	(185),
ringPatternOutOfRange	(186),
ringVolumnOutOfRange	(187),
routeingAlgorithmOutOfRange	(188),
speakerMuteOutOfRange	(189),
speakerVolumeOutOfRange	(190),
switchingCcittTypeOutOfRange	(191),
systemStatusOutOfRange	(192),
toneCharacterOutOfRange	(193),
toneDurationOutOfRange	(194)}
SecurityErrors ::= ENUMERATED	
{	
generic	( 0),
sequenceNumberViolated	( 1),
timeStampViolated	( 2),
securityInfoViolated	( 4)}
StateIncompatibilityErrors ::= ENUMERATED	
{	
generic	( 1),
invalidObjectState	( 2),
invalidDeviceState	(15),
connectedCallExists	(16),
invalidActiveDeviceState	(17),
invalidCalledDeviceState	(18),
invalidCallingDeviceState	(19),
invalidCallToParkDeviceState	(20),
invalidDestinationDeviceState	(21),
invalidDivertingDeviceState	(22),
invalidHeldDeviceState	(23),
invalidParkToDeviceState	(24),
invalidConnectionState	(25),
invalidActiveConnectionState	(26),
invalidConnectionIDForActiveCall	( 3),
invalidHeldConnectionState	(27),
noActiveCall	( 4),
noCallToAnswer	( 8),
noCallToClear	( 6),
noCallToComplete	( 9),
noConnectionToClear	( 7),
noHeldCall	( 5),
incorrectMsgState	(28),
beginningOfMsg	(13),
endOfMsg	(12),
msgSuspended	(14),
notAbleToPlay	(10),
notAbleToResume	(11)}
SystemResourceAvailabilityErrors ::= ENUMERATED	
{	
generic	( 1),
resourceBusy	( 2),
internalResourceBusy	( 3),
classifierBusy	( 9),
noMediaChannelsAvailable	(10),

STANDARD PDF COPY Click to view the full PDF of ISO/IEC 18052:2000

```

channelsInUseForBridgedDevices          (11),
channelsInUseForData                    (12),
toneDetectorBusy                        (13),
toneGeneratorBusy                       (14),
networkBusy                             ( 5),
resourceOutOfService                    ( 4),
deviceOutOfService                      (15),
activeDeviceOutOfService                (16),
calledDeviceOutOfService                (17),
callingDeviceOutOfService               (18),
callToParkDeviceOutOfService           (19),
destinationDeviceOutOfService           (20),
divertingDeviceOutOfService             (21),
heldDeviceOutOfService                  (22),
parkToDeviceOutOfService                (23),
pickupDeviceOutOfService                (24),
networkOutOfService                     ( 6),
otherResourceOutOfService                (25),
resourceLimitExceeded                   (26),
overallMonitorLimitExceeded             ( 7),
conferenceMemberLimitExceeded           ( 8),
registrationLimitExceeded                (27)}

SubscribedResourceAvailabilityErrors ::= ENUMERATED
{
    generic                               ( 1),
    objectMonitorLimitExceeded            ( 2),
    trunkLimitExceeded                    ( 3),
    outstandingReqLimitExceeded           ( 4),
    objectRegistrationLimitExceeded        ( 5)}

PerformanceManagementErrors ::= ENUMERATED
{
    generic                               (1),
    performanceLimitExceeded              (2)}

PrivateDataInfoErrors ::= ENUMERATED
{
    CSTAPrivateDataInfoError(1)}

UnspecifiedErrors ::= NULL

END -- of CSTA-error-definition

```

## 10 Event Report Service

```

CSTA-event-report-definitions
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-report-definitions( 21) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
CallClearedEvent FROM CSTA-call-cleared-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-cleared-event( 22) }
ConferencedEvent FROM CSTA-conferenced-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) conferenced-event( 23) }
ConnectionClearedEvent FROM CSTA-connection-cleared-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-cleared-event( 24) }
DeliveredEvent FROM CSTA-delivered-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) delivered-event( 25) }
DivertedEvent FROM CSTA-diverted-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) diverted-event( 26) }
EstablishedEvent FROM CSTA-established-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) established-event( 27) }
FailedEvent FROM CSTA-failed-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) failed-event( 28) }
HeldEvent FROM CSTA-held-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) held-event( 29) }
NetworkReachedEvent FROM CSTA-network-reached-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) network-reached-event( 30) }
OriginatedEvent FROM CSTA-originated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) originated-event( 31) }
QueuedEvent FROM CSTA-queued-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) queued-event( 32) }
RetrievedEvent FROM CSTA-retrieved-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) retrieved-event( 33) }
ServiceInitiatedEvent FROM CSTA-service-initiated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) service-initiated-event( 34) }
TransferredEvent FROM CSTA-transferred-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) transferred-event( 35) }
AutoAnswerEvent FROM CSTA-auto-answer-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) auto-answer-event( 40) }
CallInformationEvent FROM CSTA-call-information-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-information-event( 41) }
DoNotDisturbEvent FROM CSTA-do-not-disturb-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) do-not-disturb-event( 42) }
ForwardingEvent FROM CSTA-forwarding-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) forwarding-event( 43) }
MessageWaitingEvent FROM CSTA-message-waiting-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) message-waiting-event( 44) }

```

MicrophoneMuteEvent FROM CSTA-microphone-mute-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) microphone-mute-event( 45) }

SpeakerMuteEvent FROM CSTA-speaker-mute-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) speaker-mute-event( 46) }

SpeakerVolumeEvent FROM CSTA-speaker-volume-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) speaker-volume-event( 47) }

AgentBusyEvent FROM CSTA-agent-busy-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) agent-busy-event( 319) }

AgentLoggedInEvent FROM CSTA-agent-logged-on-event  
 { iso( 1) identified-organization( 3) icd-ecma( 321)  
   standard( 0) csta3( 285) agent-logged-on-event( 321) }

AgentLoggedOffEvent FROM CSTA-agent-logged-off-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) agent-logged-off-event( 320) }

AgentNotReadyEvent FROM CSTA-agent-not-ready-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) agent-not-ready-event( 322) }

AgentReadyEvent FROM CSTA-agent-ready-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) agent-ready-event( 323) }

AgentWorkingAfterCallEvent FROM CSTA-agent-working-after-call-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) agent-working-after-call-event( 324) }

BackInServiceEvent FROM CSTA-back-in-service-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) back-in-service-event( 333) }

OutOfServiceEvent FROM CSTA-out-of-service-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) out-of-service-event( 335) }

PrivateEvent FROM CSTA-private-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) private-event( 71) }

PlayEvent FROM CSTA-play-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) play( 75) }

RecordEvent FROM CSTA-record-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) record( 76) }

ReviewEvent FROM CSTA-review-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) review( 77) }

StopEvent FROM CSTA-stop-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) stop( 78) }

SuspendPlayEvent FROM CSTA-suspend-play-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) suspend-play( 79) }

SuspendRecordEvent FROM CSTA-suspend-record-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) suspend-record( 80) }

VoiceAttributesChangeEvent FROM CSTA-voice-attributes-change-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) voice-attributes-change-event( 74) }

MonitorCrossRefID FROM CSTA-status-reporting  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) status-reporting( 126) }

BridgedEvent FROM CSTA-bridged-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) bridged-event( 224) }

DigitsDialedEvent FROM CSTA-digits-dialed-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)  
   standard( 0) csta3( 285) digits-dialed-event( 225) }

NetworkCapabilitiesChangedEvent FROM CSTA-network-capabilities-changed-event  
 { iso( 1) identified-organization( 3) icd-ecma( 12)

```

        standard( 0) csta3( 285) network-capabilities-changed-event( 226) }
OfferedEvent FROM CSTA-offered-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) offered-event( 227) }
ChargingEvent FROM CSTA-charging-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) charging-event( 240) }
DigitsGeneratedEvent FROM CSTA-digits-generated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) digits-generated-event( 241) }
TelephonyTonesGeneratedEvent FROM CSTA-telephony-tones-generated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) telephony-tones-generated-event( 242) }
ServiceCompletionFailureEvent FROM CSTA-service-completion-failure-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) service-completion-failure-event( 243) }
MediaAttachedEvent FROM CSTA-media-attached-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-attached-event( 246) }
MediaDetachedEvent FROM CSTA-media-detached-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-detached-event( 247) }
ButtonInformationEvent FROM CSTA-button-information-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) button-information-event( 283) }
ButtonPressEvent FROM CSTA-button-press-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) button-press-event( 284) }
DisplayUpdatedEvent FROM CSTA-display-updated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) display-updated-event( 285) }
HookswitchEvent FROM CSTA-hookswitch-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) hookswitch-event( 286) }
LampModeEvent FROM CSTA-lamp-mode-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) lamp-mode-event( 287) }
MicrophoneGainEvent FROM CSTA-microphone-gain-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) microphone-gain-event( 288) }
RingerStatusEvent FROM CSTA-ringer-status-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) ringer-status-event( 289) }
AutoWorkModeEvent FROM CSTA-auto-work-mode-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) auto-work-mode-event( 326) }
CallBackEvent FROM CSTA-call-back-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-back-event( 327) }
CallBackMessageEvent FROM CSTA-call-back-message-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-back-message-event( 328) }
CallerIDStatusEvent FROM CSTA-caller-id-status-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) caller-id-status-event( 329) }
RouteingModeEvent FROM CSTA-routeing-mode-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) routeing-mode-event( 332) }
DeviceCapsChangedEvent FROM CSTA-device-capabilities-changed-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-capabilities-changed-event( 334) }
;

cSTAEventReport OPERATION ::=
{
    ARGUMENT CSTAEventReportArgument
    ALWAYS RESPONDS FALSE
    CODE local:21
}

```

CSTAEventReportArgument ::= SEQUENCE  
 {
     crossRefIdentifier     MonitorCrossRefID,  
     eventSpecificInfo     EventSpecificInfo}

EventSpecificInfo ::= CHOICE  
 {
     callControlEvents         [0] CallControlEvents,  
     callAssociatedEvents     [1] CallAssociatedEvents,  
     mediaAttachmentEvents    [2] MediaAttachmentEvents,  
     physicalDeviceFeatureEvents [3] PhysicalDeviceFeatureEvents,  
     logicalDeviceFeatureEvents [4] LogicalDeviceFeatureEvents,  
     deviceMaintenanceEvents [5] DeviceMaintenanceEvents,  
     voiceUnitEvents         [6] VoiceUnitEvents,  
     vendorSpecEvents         [7] VendorSpecEvents}

CallControlEvents ::= CHOICE  
 {
     bridged                   [ 0] IMPLICIT BridgedEvent,  
     callCleared              [ 1] IMPLICIT CallClearedEvent,  
     conferenced              [ 2] IMPLICIT ConferencedEvent,  
     connectionCleared        [ 3] IMPLICIT ConnectionClearedEvent,  
     delivered                 [ 4] IMPLICIT DeliveredEvent,  
     digitsDialed             [ 5] IMPLICIT DigitsDialedEvent,  
     diverted                 [ 6] IMPLICIT DivertedEvent,  
     established              [ 7] IMPLICIT EstablishedEvent,  
     failed                    [ 8] IMPLICIT FailedEvent,  
     held                      [ 9] IMPLICIT HeldEvent,  
     networkCapabilitiesChanged [10] IMPLICIT NetworkCapabilitiesChangedEvent,  
     networkReached           [11] IMPLICIT NetworkReachedEvent,  
     offered                  [12] IMPLICIT OfferedEvent,  
     originated               [13] IMPLICIT OriginatedEvent,  
     queued                   [14] IMPLICIT QueuedEvent,  
     retrieved                [15] IMPLICIT RetrievedEvent,  
     serviceInitiated         [16] IMPLICIT ServiceInitiatedEvent,  
     transferred              [17] IMPLICIT TransferredEvent}

CallAssociatedEvents ::= CHOICE  
 {
     callInformation           [ 0] IMPLICIT CallInformationEvent,  
     charging                  [ 1] IMPLICIT ChargingEvent,  
     digitsGeneratedEvent      [ 2] IMPLICIT DigitsGeneratedEvent,  
     telephonyTonesGeneratedEvent [ 3] IMPLICIT TelephonyTonesGeneratedEvent,  
     serviceCompletionFailure   [ 4] IMPLICIT ServiceCompletionFailureEvent}

MediaAttachmentEvents ::= CHOICE  
 {
     mediaAttached             [ 0] IMPLICIT MediaAttachedEvent,  
     mediaDetached             [ 1] IMPLICIT MediaDetachedEvent}

PhysicalDeviceFeatureEvents ::= CHOICE  
 {
     buttonInformation         [ 0] IMPLICIT ButtonInformationEvent,  
     buttonPress               [ 1] IMPLICIT ButtonPressEvent,  
     displayUpdated            [ 2] IMPLICIT DisplayUpdatedEvent,  
     hookswitch                [ 3] IMPLICIT HookswitchEvent,  
     lampMode                  [ 4] IMPLICIT LampModeEvent,  
     messageWaiting            [ 5] IMPLICIT MessageWaitingEvent,  
     microphoneGain            [ 6] IMPLICIT MicrophoneGainEvent,  
     microphoneMute            [ 7] IMPLICIT MicrophoneMuteEvent,  
     ringerStatus              [ 8] IMPLICIT RingerStatusEvent,  
     speakerMute                [ 9] IMPLICIT SpeakerMuteEvent,  
     speakerVolume             [10] IMPLICIT SpeakerVolumeEvent}

LogicalDeviceFeatureEvents ::= CHOICE  
 {
     agentBusy                 [ 0] IMPLICIT AgentBusyEvent,  
     agentLoggedOn             [ 1] IMPLICIT AgentLoggedOnEvent,  
     agentLoggedOff            [ 2] IMPLICIT AgentLoggedOffEvent,  
     agentNotReady             [ 3] IMPLICIT AgentNotReadyEvent,  
     agentReady                [ 4] IMPLICIT AgentReadyEvent,  
     agentWorkingAfterCall     [ 5] IMPLICIT AgentWorkingAfterCallEvent,  
     autoAnswer                [ 6] IMPLICIT AutoAnswerEvent,

```

autoWorkMode          [ 7] IMPLICIT AutoWorkModeEvent,
callBack              [ 8] IMPLICIT CallBackEvent,
callBackMessage       [ 9] IMPLICIT CallBackMessageEvent,
callerIDStatus        [10] IMPLICIT CallerIDStatusEvent,
doNotDisturb          [11] IMPLICIT DoNotDisturbEvent,
forwarding            [12] IMPLICIT ForwardingEvent,
routeingMode          [13] IMPLICIT RouteingModeEvent}

DeviceMaintenanceEvents ::= CHOICE
{
    backInService      [ 0] IMPLICIT BackInServiceEvent,
    deviceCapabilityChanged [ 1] IMPLICIT DeviceCapsChangedEvent,
    outOfService       [ 2] IMPLICIT OutOfServiceEvent}

VoiceUnitEvents ::= CHOICE
{
    play               [ 0] IMPLICIT PlayEvent,
    record             [ 1] IMPLICIT RecordEvent,
    review             [ 2] IMPLICIT ReviewEvent,
    stop              [ 3] IMPLICIT StopEvent,
    suspendPlay       [ 4] IMPLICIT SuspendPlayEvent,
    suspendRecord     [ 5] IMPLICIT SuspendRecordEvent,
    voiceAttributesChange [ 6] IMPLICIT VoiceAttributesChangeEvent}

VendorSpecEvents ::= CHOICE
{
    privateEvent      [ 0] IMPLICIT PrivateEvent}

END -- of CSTA-event-report-definitions

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 11 Capability exchange services

### 11.1 Services

#### 11.1.1 Get logical device information

```

CSTA-get-logical-device-information
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-logical-device-information( 201) }

DEFINITIONS ::=
BEGIN

IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)};
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
MonitorFilter, MonitorMediaClass FROM CSTA-status-reporting
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DeviceCategory, GroupDeviceAttributes, NamedDeviceTypes, ACDModels, AgentLogOnModels,
  AppearanceType, MiscMonitorCaps, TransAndConfSetup, MediaServiceCapsList,
  RouteingServList, LogDevServList, LogDevEvtsList
FROM CSTA-capability-exchange
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) capability-exchange( 131) }
NumberOfChannels, MaxChannelBind, ConnectionRateList, DelayToleranceList
FROM CSTA-data-call-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-call-types( 134) };

getLogicalDeviceInformation OPERATION ::=
{
  ARGUMENT    GetLogicalDeviceInformationArgument
  RESULT      GetLogicalDeviceInformationResult
  ERRORS      {universalFailure}
  CODE        local:201
}

GetLogicalDeviceInformationArgument ::= SEQUENCE
{
  device          DeviceID,
  extensions      CSTACommonArgumentsOPTIONAL}

GetLogicalDeviceInformationResult ::= SEQUENCE
{
  deviceCategory          [ 0] IMPLICIT DeviceCategory          DEFAULT station,
  groupDeviceAttributes   [ 1] IMPLICIT GroupDeviceAttributes OPTIONAL,
  namedDeviceTypes        [ 2] IMPLICIT NamedDeviceTypes        OPTIONAL,
  shortFormDeviceID       [ 3] DeviceID                          OPTIONAL,
  hasPhysicalElement       BOOLEAN,
  acdModels                ACDModels,
  agentLogOnModels         [ 4] IMPLICIT AgentLogOnModels        OPTIONAL,
  appearanceAddressable    BOOLEAN,
  appearanceType           AppearanceType,
  appearanceList           [ 5] IMPLICIT SEQUENCE OF IA5String OPTIONAL,
  otherPhysicalDeviceList  [ 6] IMPLICIT SEQUENCE OF DeviceID  OPTIONAL,
  miscMonitorCaps          [ 7] IMPLICIT MiscMonitorCaps        OPTIONAL,
  associatedGroupList      [ 8] IMPLICIT SEQUENCE OF DeviceID  OPTIONAL,
  maxCallbacks             [ 9] IMPLICIT INTEGER                 OPTIONAL,
  maxAutoAnswerRings      [10] IMPLICIT INTEGER                 OPTIONAL,
  maxActiveCalls           [11] IMPLICIT INTEGER                 OPTIONAL,
}

```

maxHeldCalls	[12]	IMPLICIT	INTEGER	OPTIONAL,
maxFwdSettings	[13]	IMPLICIT	INTEGER	OPTIONAL,
maxDevicesInConf	[14]	IMPLICIT	INTEGER	OPTIONAL,
transAndConfSetup	[15]	IMPLICIT	TransAndConfSetup	OPTIONAL,
deviceOnDeviceMonitorFilter	[16]	IMPLICIT	MonitorFilter	OPTIONAL,
deviceOnConnectionMonitorFilter	[17]	IMPLICIT	MonitorFilter	OPTIONAL,
callOnDeviceMonitorFilter	[18]	IMPLICIT	MonitorFilter	OPTIONAL,
callOnConnectionMonitorFilter	[19]	IMPLICIT	MonitorFilter	OPTIONAL,
mediaClassSupport	[20]	IMPLICIT	MonitorMediaClass	OPTIONAL,
mediaServiceCapsList	[21]	IMPLICIT	MediaServiceCapsList	OPTIONAL,
connectionRateList	[22]	IMPLICIT	ConnectionRateList	OPTIONAL,
delayToleranceList	[23]	IMPLICIT	DelayToleranceList	OPTIONAL,
numberOfChannels	[24]	IMPLICIT	NumberOfChannels	OPTIONAL,
maxChannelBind	[25]	IMPLICIT	MaxChannelBind	OPTIONAL,
routeingServList	[26]	IMPLICIT	RouteingServList	OPTIONAL,
logDevServList	[27]	IMPLICIT	LogDevServList	OPTIONAL,
logDevEvtsList	[28]	IMPLICIT	LogDevEvtsList	OPTIONAL,
extensions			CSTACommonArguments	OPTIONAL}

END -- of CSTA-get-logical-device-information

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 11.1.2 Get physical device information

```

CSTA-get-physical-device-information
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-physical-device-information( 202) }

DEFINITIONS ::=
BEGIN

IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
MonitorFilter FROM CSTA-status-reporting
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DeviceCategory, GroupDeviceAttributes, NamedDeviceTypes, PhysDevServList, PhysDevEvtsList
FROM CSTA-capability-exchange
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) capability-exchange( 131) };

getPhysicalDeviceInformation OPERATION ::=
{
  ARGUMENT    GetPhysicalDeviceInformationArgument
  RESULT      GetPhysicalDeviceInformationResult
  ERRORS      {universalFailure}
  CODE        local:202
}

GetPhysicalDeviceInformationArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments    OPTIONAL}

GetPhysicalDeviceInformationResult ::= SEQUENCE
{
  deviceCategory [ 0] IMPLICIT DeviceCategory          DEFAULT sta-
tion,
  groupDeviceAttributes [ 1] IMPLICIT GroupDeviceAttributes  OPTIONAL,
  namedDeviceTypes [ 2] IMPLICIT NamedDeviceTypes          OPTIONAL,
  hasLogicalElement BOOLEAN,
  otherLogicalDeviceList [ 3] IMPLICIT SEQUENCE OF DeviceID  OPTIONAL,
  deviceModelName [ 4] IMPLICIT IA5String (SIZE(0..64))  OPTIONAL,
  deviceOnDeviceMonitorFilter [ 5] IMPLICIT MonitorFilter    OPTIONAL,
  deviceOnConnectionMonitorFilter [ 6] IMPLICIT MonitorFilter  OPTIONAL,
  callOnDeviceMonitorFilter [ 7] IMPLICIT MonitorFilter    OPTIONAL,
  callOnConnectionMonitorFilter [ 8] IMPLICIT MonitorFilter    OPTIONAL,
  maxDisplays [ 9] IMPLICIT INTEGER                        OPTIONAL,
  maxButtons [10] IMPLICIT INTEGER                         OPTIONAL,
  maxLamps [11] IMPLICIT INTEGER                          OPTIONAL,
  maxRingPatterns [12] IMPLICIT INTEGER                   OPTIONAL,
  physDevServList [13] IMPLICIT PhysDevServList            OPTIONAL,
  physDevEvtsList [14] IMPLICIT PhysDevEvtsList           OPTIONAL,
  extensions CSTACommonArguments                          OPTIONAL}

END -- of CSTA-get-physical-device-information

```

## 11.1.3 Get switching function capabilities

```

CSTA-get-switching-function-capabilities
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-switching-function-capabilities( 203) }

DEFINITIONS ::=
BEGIN

IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
MonitorMediaClass, MonitorFilter FROM CSTA-status-reporting
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) status-reporting( 126) }
TimeInfo FROM CSTA-security
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) security( 128) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) }
ACDModels, CapExchangeServList, DeviceIDFormat, FilterThreshold,
IgnoreUnsupportedParameters, CallControlServList, CallControlEvtsList,
CallAssociatedServList, CallAssociatedEvtsList, MediaServList, MediaEvtsList,
RouteingServList, PhysDevEvtsList, PhysDevServList, LogicalEvtsList, LogicalServList,
DeviceMaintEvtsList, IOServicesServList, DataCollectionServList, VoiceUnitServList,
VoiceUnitEvtsList, CDRServList, VendorSpecificServList, VendorSpecificEvtsList,
MaxLengthParameters, TransAndConfSetup, MediaServiceCapsList, MiscMonitorCaps,
MonitoringServList, PauseTime, SnapshotServList, SwAppearanceAddressability,
SwAppearanceTypes, SwDomainFeatures, SystemStatusServList, TimeStampMode
FROM CSTA-capability-exchange
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) capability-exchange( 131) }
NumberOfChannels, MaxChannelBind, ConnectionRateList, DelayToleranceList
FROM CSTA-data-call-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-call-types( 134) };

getSwitchingFunctionCapabilities OPERATION ::=
{
  ARGUMENT  GetSwitchingFunctionCapsArgument
  RESULT    GetSwitchingFunctionCapsResult
  ERRORS    {universalFailure}
  CODE      local:203
}

GetSwitchingFunctionCapsArgument ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData      NULL}

GetSwitchingFunctionCapsResult ::= SEQUENCE
{
  switchingSubDomainName      IA5String (SIZE(0..64)),
  manufacturerName           IA5String (SIZE(0..64)),
  profiles                    Profiles,
  deviceIDFormat              DeviceIDFormat,
  swDomainFeatures            SwDomainFeatures,
  swAppearanceAddressability  SwAppearanceAddressability,
  swAppearanceTypes          SwAppearanceTypes,
  ignoreUnsupportedParameters IgnoreUnsupportedParameters,
  callCharacteristicsSupported [0] IMPLICIT CallCharacteristics      OPTIONAL,
  mediaClassSupport           [1] IMPLICIT MonitorMediaClass        OPTIONAL,
  numberOfChannels            [2] IMPLICIT NumberOfChannels         OPTIONAL,
}

```

maxChannelBind	[3] IMPLICIT MaxChannelBind	OPTIONAL,
miscMediaCallCharacteristics	[4] IMPLICIT MiscMediaCallCharacteristics	OPTIONAL,
connectionRateList	[5] IMPLICIT ConnectionRateList	OPTIONAL,
delayToleranceList	[6] IMPLICIT DelayToleranceList	OPTIONAL,
pauseTime	[7] IMPLICIT PauseTime	OPTIONAL,
currentTime	[8] IMPLICIT TimeInfo	OPTIONAL,
messageSeqNumbers	[9] IMPLICIT MessageSeqNumbers	OPTIONAL,
timeStampMode	[10] IMPLICIT TimeStampMode	OPTIONAL,
securityMode	[11] IMPLICIT SecurityMode	OPTIONAL,
securityFormat	[12] IMPLICIT SecurityFormat	OPTIONAL,
privateDataFormat	[13] IMPLICIT SecurityFormat	OPTIONAL,
transAndConfSetup	[14] IMPLICIT TransAndConfSetup	OPTIONAL,
monitorFilterItems	[15] IMPLICIT MonitorFilterItems	OPTIONAL,
miscMonitorCaps	[16] IMPLICIT MiscMonitorCaps	OPTIONAL,
correlatorDataSupported	[17] IMPLICIT BOOLEAN	OPTIONAL,
dynamicFeatureSupported	[18] IMPLICIT DynamicFeatureSupported	OPTIONAL,
callLinkageOptions	[28] IMPLICIT CallLinkageOptions	OPTIONAL,
aCDModels	[19] IMPLICIT ACDModels	OPTIONAL,
agentLogOnModels	[20] IMPLICIT AgentLogOnModels	OPTIONAL,
agentStateModels	[21] IMPLICIT AgentStateModels	OPTIONAL,
connectionView	ConnectionView,	
maxLengthParameters	MaxLengthParameters,	
servEvtsList	[22] IMPLICIT ServevtsList	OPTIONAL,
privateDataVersionList	[23] IMPLICIT PrivateDataVersionList	OPTIONAL,
systemStatusTimer	[24] IMPLICIT INTEGER (0..180)	OPTIONAL,
simpleThreshold	[25] IMPLICIT INTEGER	OPTIONAL,
filterThreshold	[26] IMPLICIT FilterThreshold	OPTIONAL,
mediaServiceCapsList	[27] IMPLICIT MediaServiceCapsList	OPTIONAL,
extensions	CSTACommonArguments	OPTIONAL}

MonitorFilterItems ::= SEQUENCE  
{
deviceOnDeviceMonitorFilter [0] IMPLICIT MonitorFilter OPTIONAL,  
deviceOnConnectionMonitorFilter [1] IMPLICIT MonitorFilter OPTIONAL,  
callOnDeviceMonitorFilter [2] IMPLICIT MonitorFilter OPTIONAL,  
callOnConnectionMonitorFilter [3] IMPLICIT MonitorFilter OPTIONAL}

MessageSeqNumbers ::= BIT STRING  
{
allEvents (0),  
allAcks (1),  
allServReqs (2)}

SecurityMode ::= ENUMERATED  
{
allEvents (0),  
allAcks (1),  
allServReqs (2)}

SecurityFormat ::= BIT STRING  
{
octetStringFromSF (0),  
otherTypeFromSF (1),  
octetStringToSF (2),  
otherTypeToSF (3)}

Profiles ::= BIT STRING  
{
basicTelephonyProfile (0),  
routeingProfile (1)}

DynamicFeatureSupported ::= ENUMERATED  
{
none (0),  
all (1),  
some (2)}

CallLinkageOptions ::= BIT STRING  
{
callLinkageFeatureSupported (0),  
threadLinkageFeatureSupported (1) }

AgentLogOnModels ::= BIT STRING  
{
logOnACDDDevice (0),

```

logOnACDGroupExplOneStep      (1),
logOnACDGroupExplTwoSteps    (2),
logOnACDGroupImplOneStep     (3)}

AgentStateModels ::= BIT STRING
{
    multiState                (0),
    multiStateSemiIndependentLinked (1),
    agentOriented             (2)}

ConnectionView ::= ENUMERATED
{
    fixed      (0),
    local      (1)}

ServEvtsList ::= SEQUENCE
{
    capExchangeServList      [ 0] IMPLICIT CapExchangeServList      OPTIONAL,
    systemServList           [ 1] IMPLICIT SystemStatusServList     OPTIONAL,
    monitoringServList       [ 2] IMPLICIT MonitoringServList       OPTIONAL,
    snapshotServList         [ 3] IMPLICIT SnapshotServList         OPTIONAL,
    callControlServList      [ 4] IMPLICIT CallControlServList      OPTIONAL,
    callControlEvtsList      [ 5] IMPLICIT CallControlEvtsList      OPTIONAL,
    callAssociatedServList   [ 6] IMPLICIT CallAssociatedServList   OPTIONAL,
    callAssociatedEvtsList   [ 7] IMPLICIT CallAssociatedEvtsList   OPTIONAL,
    mediaServList            [ 8] IMPLICIT MediaServList            OPTIONAL,
    mediaEvtsList            [ 9] IMPLICIT MediaEvtsList            OPTIONAL,
    routeingServList         [10] IMPLICIT RouteingServList         OPTIONAL,
    physDevServList          [11] IMPLICIT PhysDevServList          OPTIONAL,
    physDevEvtsList          [12] IMPLICIT PhysDevEvtsList          OPTIONAL,
    logicalServList          [13] IMPLICIT LogicalServList          OPTIONAL,
    logicalEvtsList          [14] IMPLICIT LogicalEvtsList          OPTIONAL,
    deviceMaintEvtsList      [15] IMPLICIT DeviceMaintEvtsList      OPTIONAL,
    iOServicesServList       [16] IMPLICIT IOServicesServList       OPTIONAL,
    dataCollectionServList   [17] IMPLICIT DataCollectionServList   OPTIONAL,
    voiceUnitServList        [18] IMPLICIT VoiceUnitServList        OPTIONAL,
    voiceUnitEvtsList        [19] IMPLICIT VoiceUnitEvtsList        OPTIONAL,
    cdrServList              [20] IMPLICIT CDRServList              OPTIONAL,
    vendorSpecificServList   [21] IMPLICIT VendorSpecificServList   OPTIONAL,
    vendorSpecificEvtsList   [22] IMPLICIT VendorSpecificEvtsList   OPTIONAL,
    statusReportingServList [23] IMPLICIT StatusReportingServList   OPTIONAL}

StatusReportingServList ::= SEQUENCE
{
    monitoringServices      [ 0] IMPLICIT MonitoringServList      OPTIONAL,
    snapshotServices        [ 1] IMPLICIT SnapshotServList        OPTIONAL,
    systemServices          [ 2] IMPLICIT SystemStatusServList    OPTIONAL}

MiscMediaCallCharacteristics ::= BIT STRING
{
    supportAdjustment      (0) }

PrivateDataVersionList ::= SEQUENCE OF INTEGER

END -- of CSTA-get-switching-function-capabilities

```



## 11.1.4 Get switching function devices

```

CSTA-get-switching-function-devices
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-switching-function-devices( 204) }

DEFINITIONS ::=
BEGIN

IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ServiceCrossRefID FROM CSTA-capability-exchange
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) capability-exchange( 131) };

getSwitchingFunctionDevices OPERATION ::=
{
  ARGUMENT    GetSwitchingFunctionDevicesArgument
  RESULT      GetSwitchingFunctionDevicesResult
  ERRORS      {universalFailure}
  CODE        local:204
}

GetSwitchingFunctionDevicesArgument ::= SEQUENCE
{
  requestedDeviceID          DeviceID          OPTIONAL,
  requestedDeviceCategory    ReqDeviceCategory OPTIONAL,
  extensions                  CSTACommonArguments OPTIONAL}

GetSwitchingFunctionDevicesResult ::= SEQUENCE
{
  serviceCrossRefID          ServiceCrossRefID,
  extensions                  CSTACommonArguments OPTIONAL}

ReqDeviceCategory ::= ENUMERATED
{
  acd                        ( 0),
  groupACD                   ( 1),
  groupHunt                   ( 2),
  groupPick                   ( 3),
  groupOther                  ( 4),
  networkInterface           ( 5),
  park                        ( 6),
  routeingDevice             ( 7),
  station                     ( 8),
  voiceUnit                   ( 9),
  other                       (10)}

END -- of CSTA-get-switching-function-devices

```

11.1.5 Switching function devices

```

CSTA-switching-function-devices
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) switching-function-devices( 205) }

DEFINITIONS ::=
BEGIN

IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServiceCrossRefID, DeviceCategory, NamedDeviceTypes FROM CSTA-capability-exchange
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) capability-exchange( 131) };

switchingFunctionDevices OPERATION ::=
{
    ARGUMENT      SwitchingFunctionDevicesArgument
    ERRORS        {universalFailure}
    CODE          local:205
}

SwitchingFunctionDevicesArgument ::= SEQUENCE
{
    serviceCrossRefID      ServiceCrossRefID,
    segmentID              INTEGER                OPTIONAL,
    lastSegment            BOOLEAN,
    deviceList              DeviceList,
    extensions              CSTACommonArguments  OPTIONAL}

DeviceList ::= SEQUENCE OF SEQUENCE
{
    deviceID              DeviceID,
    deviceCategory        [0] IMPLICIT DeviceCategory    DEFAULT station,
    namedDeviceTypes      [1] IMPLICIT NamedDeviceTypes  OPTIONAL,
    deviceAttributes      DeviceAttributes                OPTIONAL,
    deviceModelName       IA5String (SIZE(0..64))         OPTIONAL}

DeviceAttributes ::= BIT STRING
{
    mediaAccessDevice     ( 0),
    routingDevice         ( 1),
    groupACD              ( 2),
    groupHunt             ( 3),
    groupPick             ( 4)}

END -- of CSTA-switching-function-devices

```

## 12 System services

### 12.1 Registration services

#### 12.1.1 Change system status filter

```

CSTA-change-system-status-filter
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) change-system-status-filter( 206) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID, SystemStatus FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

changeSysStatFilter OPERATION ::=
{
    ARGUMENT    ChangeSysStatFilterArg
    RESULT      ChangeSysStatFilterRes
    ERRORS      {universalFailure}
    CODE        local: 206
}

ChangeSysStatFilterArg ::= SEQUENCE
{
    sysStatRegisterID    SysStatRegisterID,
    requestedStatusFilter SystemStatus,
    extensions            CSTACommonArguments    OPTIONAL}

ChangeSysStatFilterRes ::= SEQUENCE
{
    actualStatusFilter    SystemStatus,
    extensions            CSTACommonArguments    OPTIONAL}

END -- of CSTA-change-system-status-filter

```

## 12.1.2 System register

```

CSTA-system-register
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) system-register( 207) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

systemRegister OPERATION ::=
{
  ARGUMENT      SystemRegisterArgument
  RESULT        SystemRegisterResult
  ERRORS        {universalFailure}
  CODE          local: 207
}

SystemRegisterArgument ::= SEQUENCE
{
  requestTypes      RequestTypes,
  requestedStatusFilter  StatusFilter          OPTIONAL,
  extensions        CSTACommonArguments      OPTIONAL}

SystemRegisterResult ::= SEQUENCE
{
  sysStatRegisterID  SysStatRegisterID,
  actualStatusFilter  StatusFilter          OPTIONAL,
  extensions          CSTACommonArguments  OPTIONAL}

StatusFilter ::= BIT STRING
{
  initializing      ( 0),
  enabled           ( 1),
  normal            ( 2),
  messageLost       ( 3),
  disabled          ( 4),
  partiallyDisabled ( 5),
  overloadImminent  ( 6),
  overloadReached   ( 7),
  overloadRelieved  ( 8)}

RequestTypes ::= BIT STRING
{
  systemStatus      (0),
  requestSystemStatus (1),
  switchingFunctionCapsChanged (2),
  switchingFunctionDevicesChanged (3)}

END -- of CSTA-system-register

```

## 12.1.3 System register abort

```

CSTA-system-register-abort
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) system-register-abort( 208) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

systemRegisterAbort OPERATION ::=
{
  ARGUMENT          SystemRegisterAbortArgument
  ERRORS            {universalFailure}
  ALWAYS RESPONDS  FALSE
  CODE              local: 208
}

SystemRegisterAbortArgument ::= SEQUENCE
{
  sysStatRegisterID  SysStatRegisterID,
  extensions         CSTACommonArguments  OPTIONAL}

END -- of CSTA-system-register-abort

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

12.1.4 System register cancel

```

CSTA-system-register-cancel
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) system-register-cancel( 209) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
;

systemRegisterCancel OPERATION ::=
{
    ARGUMENT    SystemRegisterCancelArgument
    RESULT      SystemRegisterCancelResult
    ERRORS      {universalFailure}
    CODE        local: 209
}

SystemRegisterCancelArgument ::= SEQUENCE
{
    sysStatRegisterID    SysStatRegisterID,
    extensions           CSTACommonArguments    OPTIONAL}

SystemRegisterCancelResult ::= CHOICE
{
    extensions           CSTACommonArguments,
    noData               NULL}

END -- of CSTA-system-register-cancel

```

STANDARDSISO.COM: Click to view the full PDF of ISO/IEC 18052:2000

## 12.2 Services

### 12.2.1 Request system status

```

CSTA-request-system-status
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) request-system-status( 210) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID, SystemStatus FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

requestSysStat OPERATION ::=
{
    ARGUMENT    RequestSysStatArg
    RESULT      RequestSysStatRes
    ERRORS      {universalFailure}
    CODE        local: 210
}

RequestSysStatArg ::= SEQUENCE
{
    sysStatRegisterID    SysStatRegisterID    OPTIONAL,
    extensions            CSTACommonArguments OPTIONAL}

RequestSysStatRes ::= SEQUENCE
{
    systemStatus    SystemStatus,
    extensions      CSTACommonArguments    OPTIONAL}

END -- of CSTA-request-system-status

```

12.2.2 System status

```

CSTA-system-status
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) system-status( 211) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID, SystemStatus FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

systemStatus OPERATION ::=
{
    ARGUMENT    SystemStatusArg
    RESULT      SystemStatusRes
    ERRORS      {universalFailure}
    CODE        local: 211
}

SystemStatusArg ::= SEQUENCE
{
    sysStatRegisterID    SysStatRegisterID    OPTIONAL,
    systemStatus          SystemStatus,
    extensions            CSTACommonArguments  OPTIONAL}

SystemStatusRes ::= CHOICE
{
    extensions    CSTACommonArguments,
    noData        NULL}

END -- of CSTA-system-status

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

### 12.2.3 Switching function capabilities changed

```

CSTA-switching-function-capabilities-changed
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) switching-function-capabilities-changed( 212) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

swFunctionCapsChanged OPERATION ::=
{
  ARGUMENT    SwFunctionCapsChangedArg
  RESULT      SwFunctionCapsChangedRes
  ERRORS      {universalFailure}
  CODE        local: 212
}

SwFunctionCapsChangedArg ::= SEQUENCE
{
  sysStatRegisterID    SysStatRegisterID    OPTIONAL,
  extensions           CSTACommonArguments  OPTIONAL}

SwFunctionCapsChangedRes ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-switching-function-capabilities-changed

```

## 12.2.4 Switching function devices changed

```

CSTA-switching-function-devices-changed
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) switching-function-devices-changed( 213) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
SysStatRegisterID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

swFunctionDevicesChanged OPERATION ::=
{
  ARGUMENT    SwFunctionDevicesChangedArg
  RESULT      SwFunctionDevicesChangedRes
  ERRORS      {universalFailure}
  CODE        local: 213
}

SwFunctionDevicesChangedArg ::= SEQUENCE
{
  sysStatRegisterID    SysStatRegisterID    OPTIONAL,
  extensions            CSTACommonArguments  OPTIONAL}

SwFunctionDevicesChangedRes ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData     NULL}

END -- of CSTA-switching-function-devices-changed

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 13 Monitoring services

### 13.1 Services

#### 13.1.1 Change monitor filter

```

CSTA-change-monitor-filter
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) change-monitor-filter( 102) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
MonitorFilter, MonitorCrossRefID FROM CSTA-status-reporting
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

changeMonitorFilter      OPERATION ::=
{
    ARGUMENT      ChangeMonitorFilterArgument
    RESULT        ChangeMonitorFilterResult
    ERRORS        {universalFailure}
    CODE          local: 72
}

ChangeMonitorFilterArgument ::=
SEQUENCE
{crossRefIdentifier      MonitorCrossRefID,
 requestedFilterList     MonitorFilter,
 extensions              CSTACommonArguments      OPTIONAL}

ChangeMonitorFilterResult ::=
SEQUENCE
{actualFilterList       MonitorFilter      OPTIONAL,
 extensions            CSTACommonArguments  OPTIONAL}

END -- of CSTA-change-monitor-filter

```

13.1.2 Monitor start

```

CSTA-monitor-start
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) monitor-start( 101) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
MonitorObject, MonitorFilter, MonitorType, MonitorMediaClass,
MonitorCrossRefID FROM CSTA-status-reporting
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

monitorStart OPERATION ::=
{
    ARGUMENT    MonitorStartArgument
    RESULT      MonitorStartResult
    ERRORS      {universalFailure}
    CODE        local: 71
}

MonitorStartArgument ::= SEQUENCE
{
    monitorObject          MonitorObject,
    requestedMonitorFilter [0] IMPLICIT MonitorFilter      OPTIONAL,
    monitorType            MonitorType                      OPTIONAL,
    requestedMonitorMediaClass [1] IMPLICIT MonitorMediaClass OPTIONAL,
    extensions              CSTACommonArguments            OPTIONAL}

MonitorStartResult ::= SEQUENCE
{
    crossRefIdentifier      MonitorCrossRefID,
    actualMonitorFilter     [0] IMPLICIT MonitorFilter      OPTIONAL,
    actualMonitorMediaClass [1] IMPLICIT MonitorMediaClass OPTIONAL,
    monitorExistingCalls    BOOLEAN                       OPTIONAL,
    extensions              CSTACommonArguments            OPTIONAL}

END -- of CSTA-monitor-start

```

STANDARDSISO.COM · Click to view the full PDF of ISO/IEC 18052:2000

## 13.1.3 Monitor stop

```

CSTA-monitor-stop
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) monitor-stop( 103) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
MonitorCrossRefID FROM CSTA-status-reporting
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

monitorStop      OPERATION ::=
{
    ARGUMENT      MonitorStopArgument
    RESULT        MonitorStopResult
    ERRORS        {universalFailure}
    CODE          local: 73
}

MonitorStopArgument ::=
SEQUENCE
{crossRefIdentifier      MonitorCrossRefID,
 extensions              CSTACommonArguments      OPTIONAL}

MonitorStopResult ::=
CHOICE
{extensions              CSTACommonArguments,
 noData                  NULL}

END -- of CSTA-monitor-stop

```

## 14 Snapshot services

### 14.1 Services

#### 14.1.1 Snapshot call

```

CSTA-snapshot-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) snapshot-call( 105) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CallingDeviceID, CalledDeviceID, AssociatedCallingDeviceID,
AssociatedCalledDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
SnapshotCallData FROM CSTA-status-reporting
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) status-reporting( 126) }
CorrelatorData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ServiceCrossRefID FROM CSTA-capability-exchange
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) capability-exchange( 131) }
MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

snapshotCall      OPERATION ::=
{
    ARGUMENT      SnapshotCallArgument
    RESULT        SnapshotCallResult
    ERRORS        {universalFailure}
    CODE          local: 75
}

SnapshotCallArgument ::=
SEQUENCE
{snapshotObject      ConnectionID,
 extensions          CSTACommonArguments    OPTIONAL}

SnapshotCallResult ::=
SEQUENCE
{crossRefIDorSnapshotData
 CHOICE
    {serviceCrossRefID [0] IMPLICIT ServiceCrossRefID,
 snapshotData          SnapshotCallData},
 mediaCallCharacteristics MediaCallCharacteristics    OPTIONAL,
 callCharacteristics    CallCharacteristics            OPTIONAL,
 callingDevice          CallingDeviceID                OPTIONAL,
 calledDevice           CalledDeviceID                 OPTIONAL,
 associatedCallingDeviceID AssociatedCallingDeviceID    OPTIONAL,
 associatedCalledDeviceID AssociatedCalledDeviceID     OPTIONAL,
 correlatorData        [1] IMPLICIT CorrelatorData    OPTIONAL,

```

callLinkageData  
extensions

[2] IMPLICIT CallLinkageData  
CSTACCommonArguments

OPTIONAL,  
OPTIONAL}

END -- of CSTA-snapshot-call

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 14.1.2 Snapshot device

```

CSTA-snapshot-device
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) snapshot-device( 104) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
SnapshotDeviceData FROM CSTA-status-reporting
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ServiceCrossRefID FROM CSTA-capability-exchange
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) capability-exchange( 131) };

snapshotDevice OPERATION ::=
{
  ARGUMENT      SnapshotDeviceArgument
  RESULT        SnapshotDeviceResult
  ERRORS        {universalFailure}
  CODE          local: 74
}

SnapshotDeviceArgument ::=
SEQUENCE
{snapshotObject DeviceID,
 extensions      CSTACommonArguments      OPTIONAL}

SnapshotDeviceResult ::=
SEQUENCE
{crossRefIDorSnapshotData
  CHOICE
  {serviceCrossRefID      ServiceCrossRefID,
   snapshotData           SnapshotDeviceData},
 extensions CSTACommonArguments      OPTIONAL}

END -- of CSTA-snapshot-device

```

## 14.1.3 Snapshot calldata

```

CSTA-snapshot-call-data
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) snapshot-call-data(106) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SnapshotCallData FROM CSTA-status-reporting
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServiceCrossRefID FROM CSTA-capability-exchange
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) capability-exchange( 131) };

snapshotCallData      OPERATION ::=
{
    ARGUMENT          SnapshotCallDataArgument
    ERRORS             {universalFailure}
    ALWAYS RESPONDS   FALSE
    CODE               local: 76
}

SnapshotCallDataArgument ::=
SEQUENCE
{serviceCrossRefID    ServiceCrossRefID,
 segmentID            INTEGER,                OPTIONAL,
 lastSegment          BOOLEAN,
 snapshotData         SnapshotCallData,
 extensions           CSTACommonArguments    OPTIONAL}

END -- of CSTA-snapshot-call-data

```

14.1.4 Snapshot devicedata

```

CSTA-snapshot-device-data
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) snapshot-device-data(107) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SnapshotDeviceData FROM CSTA-status-reporting
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) status-reporting( 126) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServiceCrossRefID FROM CSTA-capability-exchange
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) capability-exchange( 131) };

snapshotDeviceData      OPERATION ::=
{
    ARGUMENT          SnapshotDeviceDataArgument
    ERRORS             {universalFailure}
    ALWAYS RESPONDS   FALSE
    CODE               local: 77
}

SnapshotDeviceDataArgument ::=
SEQUENCE
{serviceCrossRefID     ServiceCrossRefID,
 segmentID             INTEGER,                OPTIONAL,
 lastSegment           BOOLEAN,
 snapshotData          SnapshotDeviceData,
 extensions            CSTACommonArguments     OPTIONAL}

END -- of CSTA-snapshot-device-data

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 15 Call control services and events

### 15.1 Services

#### 15.1.1 Accept call

```

CSTA-accept-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) accept-call( 214) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

acceptCall OPERATION ::=
{
    ARGUMENT    AcceptCallArgument
    RESULT      AcceptCallResult
    ERRORS      {universalFailure}
    CODE        local: 214
}

AcceptCallArgument ::= SEQUENCE
{
    callToBeAccepted          ConnectionID,
    correlatorData            CorrelatorData OPTIONAL,
    userData                   UserData          OPTIONAL,
    extensions                 CSTACommonArguments OPTIONAL}

AcceptCallResult ::= CHOICE
{
    extensions                 CSTACommonArguments,
    noData                     NULL}

END -- of CSTA-accept-call

```

15.1.2 Alternate call

```

CSTA-alternate-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) alternate-call( 1) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConsultOptions FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
;

alternateCall      OPERATION ::=
{
    ARGUMENT      AlternateCallArgument
    RESULT        AlternateCallResult
    ERRORS        {universalFailure}
    CODE          local: 1
}

AlternateCallArgument ::= SEQUENCE
{
    heldCall          ConnectionID,
    activeCall        ConnectionID,
    connectionReservation  BOOLEAN,
    consultOptions    ConsultOptions,
    extensions        CSTACommonArguments
}

AlternateCallResult ::=CHOICE
{
    extensions CSTACommonArguments,
    noData     NULL}

END -- of CSTA-alternate-call

```

STANDARDSISO.COM: Click to view the full PDF of ISO/IEC 18052:2000

## 15.1.3 Answer call

```

CSTA-answer-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) answer-call( 2) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

answerCall OPERATION ::=
{
    ARGUMENT    AnswerCallArgument
    RESULT      AnswerCallResult
    ERRORS      {universalFailure}
    CODE        local: 2
}

AnswerCallArgument ::=
SEQUENCE
{
    callToBeAnswered      ConnectionID,
    correlatorData        CorrelatorData           OPTIONAL,
    userData               UserData                OPTIONAL,
    extensions             CSTACommonArguments     OPTIONAL}

AnswerCallResult ::=
CHOICE
{extensions      CSTACommonArguments,
 noData         NULL}

END -- of CSTA-answer-call

```

## 15.1.4 Call back call-related

```

CSTA-call-back-call-related
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-back-call-related( 215) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) };

callBack OPERATION ::=
{
  ARGUMENT    CallBackArgument
  RESULT      CallBackResult
  ERRORS      {universalFailure}
  CODE        local: 215
}

CallBackArgument ::=
  SEQUENCE
  {
    callbackConnection    ConnectionID,
    callCharacteristics    CallCharacteristics    OPTIONAL,
    extensions             CSTACommonArguments    OPTIONAL}

CallBackResult ::=
  SEQUENCE
  {
    targetDevice    DeviceID    OPTIONAL,
    extensions      CSTACommonArguments    OPTIONAL}

END -- of CSTA-call-back-call-related

```

STANDARDS150.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 15.1.5 Call back message call-related

```

CSTA-call-back-message-call-related
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-back-message-call-related( 216) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

callBackMessage OPERATION ::=
{
    ARGUMENT    CallBackMessageArgument
    RESULT      CallBackMessageResult
    ERRORS      {universalFailure}
    CODE        local: 216
}

CallBackMessageArgument ::=
SEQUENCE
{
    callbackMessageConnection    ConnectionID,
    extensions                    CSTACommonArguments OPTIONAL}

CallBackMessageResult ::=
SEQUENCE
{
    targetDevice    DeviceID            OPTIONAL,
    extensions      CSTACommonArguments OPTIONAL}

END -- of CSTA-call-back-message-call-related

```

15.1.6 Camp on call

```

CSTA-camp-on-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) camp-on-call( 217) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

camponCall OPERATION ::=
{
    ARGUMENT    CamponCallArgument
    RESULT      CamponCallResult
    ERRORS      {universalFailure}
    CODE        local: 217
}

CamponCallArgument ::=
    SEQUENCE
    {
        camponConnection    ConnectionID,
        extensions           CSTACommonArguments    OPTIONAL}

CamponCallResult ::=
    CHOICE
    {
        extensions           CSTACommonArguments,
        noData               NULL    }

END -- of CSTA-camp-on-call

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 15.1.7 Clear call

```

CSTA-clear-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) clear-call( 4) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

clearCall          OPERATION ::=
{
    ARGUMENT      ClearCallArgument
    RESULT        ClearCallResult
    ERRORS        {universalFailure}
    CODE          local: 4
}

ClearCallArgument ::=
SEQUENCE
{
    callToBeCleared ConnectionID,
    userData          UserData          OPTIONAL,
    extensions        CSTACommonArguments OPTIONAL}

ClearCallResult ::=
CHOICE
{extensions        CSTACommonArguments,
 noData            NULL}

END -- of CSTA-clear-call

```

15.1.8 Clear Connection

```

CSTA-clear-connection
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) clear-connection( 5) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

clearConnection OPERATION ::=
{
    ARGUMENT    ClearConnectionArgument
    RESULT      ClearConnectionResult
    ERRORS      {universalFailure}
    CODE        local: 5
}

ClearConnectionArgument ::=
SEQUENCE
{connectionToBeCleared ConnectionID,
 correlatorData         CorrelatorData          OPTIONAL,
 userData               UserData                OPTIONAL,
 extensions             CSTACommonArguments     OPTIONAL}

ClearConnectionResult ::=
CHOICE
{extensions             CSTACommonArguments,
 noData                NULL}

END -- of CSTA-clear-connection

```

STANDARDSISO.COM: Click to view the full PDF of ISO/IEC 18052:2000

## 15.1.9 Conference call

```

CSTA-conference-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) conference-call( 6) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
ConnectionList FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

conferenceCall OPERATION ::=
{
    ARGUMENT    ConferenceCallArgument
    RESULT      ConferenceCallResult
    ERRORS      {universalFailure}
    CODE        local: 6
}

ConferenceCallArgument ::=
SEQUENCE
{
    heldCall      ConnectionID,
    activeCall    ConnectionID,
    extensions     CSTACommonArguments    OPTIONAL}

ConferenceCallResult ::=
SEQUENCE
{
    conferenceCall    ConnectionID,
    connections       [0] IMPLICIT ConnectionList    OPTIONAL,
    conferenceCallInfo [1] IMPLICIT ConnectionInformation OPTIONAL,
    extensions         CSTACommonArguments    OPTIONAL}

END -- of CSTA-conference-call

```

## 15.1.10 Consultation call

```

CSTA-consultation-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) consultation-call( 7) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
AccountInfo, AuthCode, CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, ConsultOptions FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

consultationCall OPERATION ::=
{
    ARGUMENT ConsultationCallArgument
    RESULT ConsultationCallResult
    ERRORS {universalFailure}
    CODE local: 7
}

ConsultationCallArgument ::= SEQUENCE
{
    existingCall [0] IMPLICIT ConnectionID,
    consultedDevice [1] DeviceID,
    connectionReservation [2] IMPLICIT BOOLEAN OPTIONAL,
    accountCode [3] IMPLICIT AccountInfo OPTIONAL,
    authCode [4] IMPLICIT AuthCode OPTIONAL,
    correlatorData [5] IMPLICIT CorrelatorData OPTIONAL,
    userData [6] IMPLICIT UserData OPTIONAL,
    callCharacteristics [7] IMPLICIT CallCharacteristics OPTIONAL,
    mediaCallCharacteristics [8] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callingConnectionInfo [9] IMPLICIT ConnectionInformation OPTIONAL,
    consultOptions [10] IMPLICIT ConsultOptions DEFAULT unrestricted,
    extensions CSTACommonArguments OPTIONAL}

ConsultationCallResult ::= SEQUENCE
{
    initiatedCall ConnectionID,
    mediaCallCharacteristics [0] IMPLICIT MediaCallCharacteristics OPTIONAL,
    initiatedCallInfo [1] IMPLICIT ConnectionInformation OPTIONAL,
    extensions CSTACommonArguments OPTIONAL}

END -- of CSTA-consultation-call

```

## 15.1.11 Deflect call

```

CSTA-deflect-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) deflect-call( 218) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

deflectCall OPERATION ::=
{
    ARGUMENT    DeflectCallArgument
    RESULT      DeflectCallResult
    ERRORS      {universalFailure}
    CODE        local: 218
}

DeflectCallArgument ::=
SEQUENCE
{
    callToBeDiverted      ConnectionID,
    newDestination        DeviceID,
    correlatorData        CorrelatorData          OPTIONAL,
    userData              UserData                OPTIONAL,
    extensions            CSTACommonArguments     OPTIONAL}

DeflectCallResult ::=
CHOICE
{extensions            CSTACommonArguments,
 noData                NULL}

END -- of CSTA-deflect-call

```

15.1.12 Dial digits

```

CSTA-dial-digits
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) dial-digits( 219) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

dialDigits OPERATION ::=
{
  ARGUMENT DialDigitsArgument
  RESULT DialDigitsResult
  ERRORS {universalFailure}
  CODE local: 219
}

DialDigitsArgument ::=
SEQUENCE
{
  diallingConnection ConnectionID,
  diallingSequence DeviceID,
  correlatorData CorrelatorData OPTIONAL,
  extensions CSTACommonArguments OPTIONAL}

DialDigitsResult ::=
CHOICE
{extensions CSTACommonArguments,
 noData NULL}

END -- of CSTA-dial-digits

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 15.1.13 Directed pickup call

```

CSTA-directed-pickup-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) directed-pickup-call( 220) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

directedPickupCall      OPERATION ::=
{
    ARGUMENT    DirectedPickupCallArgument
    RESULT      DirectedPickupCallResult
    ERRORS      {universalFailure}
    CODE        local: 220
}

DirectedPickupCallArgument ::=
SEQUENCE
{
    callToBePickedUp      ConnectionID,
    requestingDevice      DeviceID,
    correlatorData        CorrelatorData          OPTIONAL,
    userData               UserData              OPTIONAL,
    extensions             CSTACommonArguments   OPTIONAL}

DirectedPickupCallResult ::=
SEQUENCE
{
    pickedCall      ConnectionID          OPTIONAL,
    pickedCallInfo  ConnectionInformation  OPTIONAL,
    extensions      CSTACommonArguments   OPTIONAL}

END -- of CSTA-directed-pickup-call

```

## 15.1.14 Group pickup call

```

CSTA-group-pickup-call
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) group-pickup-call( 221) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, UserData FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) };

groupPickupCall OPERATION ::=
{
  ARGUMENT    GroupPickupCallArgument
  RESULT      GroupPickupCallResult
  ERRORS      {universalFailure}
  CODE        local: 221
}

GroupPickupCallArgument ::=
SEQUENCE
{
  newDestination DeviceID,
  pickGroup       DeviceID                OPTIONAL,
  correlatorData CorrelatorData           OPTIONAL,
  userData        UserData                 OPTIONAL,
  extensions      CSTACommonArguments     OPTIONAL}

GroupPickupCallResult ::=
SEQUENCE
{
  pickedCall      ConnectionID             OPTIONAL,
  pickedCallInfo ConnectionInformation     OPTIONAL,
  extensions      CSTACommonArguments     OPTIONAL}

END -- of CSTA-group-pickup-call

```

## 15.1.15 Hold call

```

CSTA-hold-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) hold-call( 9) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

holdCall      OPERATION ::=
{
    ARGUMENT          HoldCallArgument
    RESULT            HoldCallResult
    ERRORS            {universalFailure}
    CODE              local: 9
}

HoldCallArgument ::=
SEQUENCE
{callToBeHeld      ConnectionID,
 connectionReservation  BOOLEAN          OPTIONAL,
 extensions          CSTACommonArguments OPTIONAL}

HoldCallResult ::=
CHOICE
{extensions          CSTACommonArguments,
 noData             NULL}

END -- of CSTA-hold-call

```

## 15.1.16 Intrude call

```

CSTA-intrude-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) intrude-call( 222) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
ParticipationType, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

intrudeCall OPERATION ::=
{
    ARGUMENT    IntrudeCallArgument
    RESULT      IntrudeCallResult
    ERRORS      {universalFailure}
    CODE        local: 222
}

IntrudeCallArgument ::= SEQUENCE
{
    intrude          ConnectionID,
    participationType ParticipationType    DEFAULT active,
    userData         UserData              OPTIONAL,
    extensions       CSTACommonArguments  OPTIONAL}

IntrudeCallResult ::= SEQUENCE
{
    conferencedCall ConnectionID          OPTIONAL,
    conferencedCallInfo ConnectionInformation OPTIONAL,
    extensions       CSTACommonArguments  OPTIONAL}

END -- of CSTA-intrude-call

```

## 15.1.17 Join call

```

CSTA-join-call
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) join-call( 223) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
AccountInfo, AuthCode, CorrelatorData, ParticipationType, UserData
FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) }
AutoOriginate FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) };

joinCall OPERATION ::=
{
  ARGUMENT    JoinCallArgument
  RESULT      JoinCallResult
  ERRORS      {universalFailure}
  CODE        local: 223
}

JoinCallArgument ::= SEQUENCE
{
  activeCall          ConnectionID,
  joiningDevice       DeviceID,
  autoOriginate       [0] IMPLICIT AutoOriginate          DEFAULT prompt,
  participationType   [1] IMPLICIT ParticipationType       DEFAULT active,
  accountCode         [2] IMPLICIT AccountInfo             OPTIONAL,
  authCode            [3] IMPLICIT AuthCode                OPTIONAL,
  correlatorData      [4] IMPLICIT CorrelatorData          OPTIONAL,
  userData            UserData                             OPTIONAL,
  extensions          CSTACommonArguments                 OPTIONAL}

JoinCallResult ::= SEQUENCE
{
  conferencedCall     ConnectionID,
  conferencedCallInfo ConnectionInformation OPTIONAL,
  extensions          CSTACommonArguments OPTIONAL}

END -- of CSTA-join-call

```

## 15.1.18 Make call

```

CSTA-make-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) make-call( 10) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
AccountInfo, AuthCode, CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
AutoOriginate, CallCharacteristics FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) }
;

makeCall          OPERATION ::=
{
    ARGUMENT      MakeCallArgument
    RESULT        MakeCallResult
    ERRORS        {universalFailure}
    CODE          local: 10
}

MakeCallArgument ::= SEQUENCE
{
    callingDevice          DeviceID,
    calledDirectoryNumber DeviceID,
    accountCode            [0] IMPLICIT AccountInfo          OPTIONAL,
    authCode               [1] IMPLICIT AuthCode             OPTIONAL,
    autoOriginate          [3] IMPLICIT AutoOriginate        DEFAULT prompt,
    correlatorData        [2] IMPLICIT CorrelatorData        OPTIONAL,
    userData               UserData                          OPTIONAL,
    callCharacteristics    CallCharacteristics                OPTIONAL,
    mediaCallCharacteristics [4] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callingConnectionInfo [5] IMPLICIT ConnectionInformation OPTIONAL,
    extensions             CSTACommonArguments                OPTIONAL}

MakeCallResult ::= SEQUENCE
{
    callingDevice          ConnectionID,
    mediaCallCharacteristics [0] IMPLICIT MediaCallCharacteristics OPTIONAL,
    initiatedCallInfo      [1] IMPLICIT ConnectionInformation OPTIONAL,
    extensions             CSTACommonArguments                OPTIONAL}

END -- of CSTA-make-call

```

## 15.1.19 Make predictive call

```

CSTA-make-predictive-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) make-predictive-call( 11) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
AccountInfo, AuthCode, CorrelatorData, UserData FROM
    CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
AutoOriginate, CallCharacteristics FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

makePredictiveCall OPERATION ::=
{
    ARGUMENT    MakePredictiveCallArgument
    RESULT      MakePredictiveCallResult
    ERRORS      { universalFailure }
    CODE        local: 11
}

MakePredictiveCallArgument ::= SEQUENCE
{
    callingDevice          DeviceID,
    calledDirectoryNumber  DeviceID,
    signallingDetection    [0] IMPLICIT SignallingDetection    OPTIONAL,
    destinationDetection  [1] IMPLICIT DestinationDetection  OPTIONAL,
    defaultAction          [2] IMPLICIT DefaultAction          OPTIONAL,
    accountCode            [3] IMPLICIT AccountInfo            OPTIONAL,
    authCode               [4] IMPLICIT AuthCode               OPTIONAL,
    autoOriginate          [5] IMPLICIT AutoOriginate          DEFAULT prompt,
    alertTime              [6] IMPLICIT INTEGER                OPTIONAL,
    correlatorData         [7] IMPLICIT CorrelatorData         OPTIONAL,
    callCharacteristics    [8] IMPLICIT CallCharacteristics    OPTIONAL,
    userData               UserData                            OPTIONAL,
    extensions             CSTACCommonArguments                OPTIONAL}

MakePredictiveCallResult ::= SEQUENCE
{
    initiatedCall          ConnectionID,
    initiatedCallInfo      ConnectionInformation                OPTIONAL,
    extensions             CSTACCommonArguments                OPTIONAL}

SignallingDetection ::= SEQUENCE
{
    signallingCondition    SignallingCondition,
    signallingConditionsAction SignallingConditionsAction}

SignallingCondition ::= ENUMERATED
{
    callDelivered          ( 0),

```

```
        callEstablished          (1)}

SignallingConditionsAction ::= ENUMERATED
{   destinationDetection      (0),
    remainConnected          (1)}

DestinationDetection ::= SEQUENCE OF SEQUENCE
{   destinationCondition      DestinationCondition,
    detectionAction           DetectionAction}

DestinationCondition ::= ENUMERATED
{   humanVoice                (0),
    answeringMachine          (1),
    facsimileMachine          (2)}

DefaultAction ::= DetectionAction

DetectionAction ::= ENUMERATED
{   clearCalledConnection     (0),
    remainConnected           (1)}

END -- of CSTA-make-predictive-call
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 15.1.20 Park call

```

CSTA-park-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) park-call( 18) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

parkCall OPERATION ::=
{
    ARGUMENT    ParkCallArgument
    RESULT      ParkCallResult
    ERRORS      {universalFailure}
    CODE        local: 18
}

ParkCallArgument ::=
SEQUENCE
{
    parking                ConnectionID,
    parkTo                 DeviceID,
    correlatorData         CorrelatorData OPTIONAL,
    extensions              CSTACommonArguments OPTIONAL}

ParkCallResult ::=
SEQUENCE
{
    parkedTo               ConnectionID OPTIONAL,
    extensions              CSTACommonArguments OPTIONAL}

END -- of CSTA-park-call

```

15.1.21 Reconnect call

```

CSTA-reconnect-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) reconnect-call( 13) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

reconnectCall OPERATION ::=
{
    ARGUMENT ReconnectCallArgument
    RESULT ReconnectCallResult
    ERRORS {universalFailure}
    CODE local: 13
}

ReconnectCallArgument ::= SEQUENCE
{
    activeCall ConnectionID,
    heldCall ConnectionID,
    extensions CSTACommonArguments OPTIONAL}

ReconnectCallResult ::=
CHOICE
{extensions CSTACommonArguments,
 noData NULL}

END -- of CSTA-reconnect-call

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 15.1.22 Retrieve call

```

CSTA-retrieve-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) retrieve-call( 14) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

retrieveCall          OPERATION ::=
{
    ARGUMENT          RetrieveCallArgument
    RESULT            RetrieveCallResult
    ERRORS            {universalFailure}
    CODE              local: 14
}

RetrieveCallArgument ::=
SEQUENCE
{
    callToBeRetrieved          ConnectionID,
    extensions                 CSTACommonArguments OPTIONAL}

RetrieveCallResult ::=
CHOICE
{extensions                 CSTACommonArguments,
 noData                     NULL}

END -- of CSTA-retrieve-call

```

## 15.1.23 Single step conference call

```

CSTA-single-step-conference
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) single-step-conference( 20) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
AccountInfo, AuthCode, CorrelatorData, ParticipationType, UserData
FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

singleStepConf OPERATION ::=
{
    ARGUMENT    SingleStepConfArgument
    RESULT      SingleStepConfResult
    ERRORS      {universalFailure}
    CODE        local: 20
}

SingleStepConfArgument ::=
SEQUENCE
{activeCall      ConnectionID,
 deviceToJoin    DeviceID,
 participationType ParticipationType           DEFAULT active,
 accountCode     [0] IMPLICIT AccountInfo     OPTIONAL,
 authCode        [1] IMPLICIT AuthCode        OPTIONAL,
 correlatorData  [2] IMPLICIT CorrelatorData   OPTIONAL,
 userData        UserData                      OPTIONAL,
 extensions      CSTACCommonArguments         OPTIONAL}

SingleStepConfResult ::=
SEQUENCE
{
    conferencedCall ConnectionID,
    conferencedCallInfo ConnectionInformation  OPTIONAL,
    extensions      CSTACCommonArguments     OPTIONAL}

END -- of CSTA-single-step-conference

```

## 15.1.24 Single step transfer call

```

CSTA-single-step-transfer
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) single-step-conference( 50) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
ConnectionList FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
AccountInfo, AuthCode, CorrelatorData, UserData
    FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

singleStepTrans OPERATION ::=
{
    ARGUMENT    SingleStepTransArgument
    RESULT      SingleStepTransResult
    ERRORS      {universalFailure}
    CODE        local: 50
}

SingleStepTransArgument ::=
SEQUENCE
{activeCall      ConnectionID,
 transferredTo   DeviceID,
 accountCode     [0] IMPLICIT AccountInfo           OPTIONAL,
 authCode        [1] IMPLICIT AuthCode               OPTIONAL,
 correlatorData  [2] IMPLICIT CorrelatorData         OPTIONAL,
 userData        UserData                             OPTIONAL,
 extensions      CSTACommonArguments                 OPTIONAL}

SingleStepTransResult ::= SEQUENCE
{transferredCall ConnectionID,
 connections     [0] IMPLICIT ConnectionList         OPTIONAL,
 transferredCallInfo [1] IMPLICIT ConnectionInformation OPTIONAL,
 extensions      CSTACommonArguments                 OPTIONAL}

END -- of CSTA-single-step-transfer

```

## 15.1.25 Transfer call

```

CSTA-transfer-call
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) transfer-call( 16) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
ConnectionList FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) }
;

transferCall      OPERATION ::=
{      ARGUMENT      TransferCallArgument
      RESULT          TransferCallResult
      ERRORS          {universalFailure}
      CODE            local: 16
}

TransferCallArgument ::= SEQUENCE
{      heldCall      ConnectionID,
      activeCall     ConnectionID,
      extensions     CSTACommonArguments OPTIONAL}

TransferCallResult ::= SEQUENCE
{      transferredCall      ConnectionID,
      connections           [0] IMPLICIT ConnectionList           OPTIONAL,
      transferredCallInfo   [1] IMPLICIT ConnectionInformation   OPTIONAL,
      extensions            CSTACommonArguments                   OPTIONAL}

END -- of CSTA-transfer-call

```

STANDARDSISO.COM: Click to view the full PDF of ISO/IEC 18052:2000

## 15.2 Events

### 15.2.1 Bridged

```

CSTA-bridged-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) bridged-event( 224) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

BridgedEvent ::= SEQUENCE
{
    bridgedConnection          ConnectionID,
    bridgedAppearance          SubjectDeviceID,
    localConnectionInfo        LocalConnectionState          OPTIONAL,
    correlatorData              [0] IMPLICIT CorrelatorData    OPTIONAL,
    userData                    UserData                      OPTIONAL,
    cause                       EventCause,
    servicesPermitted           [1] IMPLICIT ServicesPermitted  OPTIONAL,
    mediaCallCharacteristics[2] [2] IMPLICIT MediaCallCharacteristics  OPTIONAL,
    callCharacteristics         [3] IMPLICIT CallCharacteristics  OPTIONAL,
    bridgedConnectionInfo      [4] IMPLICIT ConnectionInformation  OPTIONAL,
    callLinkageData             [5] IMPLICIT CallLinkageData      OPTIONAL,
    extensions                  CSTACommonArguments           OPTIONAL}

END -- of CSTA-bridged-event

```

## 15.2.2 Call cleared

```

CSTA-call-cleared-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-cleared-event( 22) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, UserData FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) }
MediaCallCharacteristics FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) }
;

CallClearedEvent ::=
SEQUENCE
{
  clearedCall          ConnectionID,
  correlatorData       [1] IMPLICIT CorrelatorData          OPTIONAL,
  userData             UserData                              OPTIONAL,
  cause               EventCause,
  mediaCallCharacteristics [2] IMPLICIT MediaCallCharacteristics OPTIONAL,
  callCharacteristics  [3] IMPLICIT CallCharacteristics      OPTIONAL,
  callLinkageData     [4] IMPLICIT CallLinkageData          OPTIONAL,
  extensions           CSTACommonArguments                  OPTIONAL}

END -- of CSTA-call-cleared-event

```

## 15.2.3 Conferenced

```

CSTA-conferenced-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) conferenced-event( 23) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState, ConnectionList FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageDataList, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

ConferencedEvent ::= SEQUENCE
{
    primaryOldCall          ConnectionID,
    secondaryOldCall        ConnectionID          OPTIONAL,
    conferencingDevice      SubjectDeviceID,
    addedParty              SubjectDeviceID,
    conferenceConnections   ConnectionList,
    localConnectionInfo     LocalConnectionState      OPTIONAL,
    correlatorData          [1] IMPLICIT CorrelatorData  OPTIONAL,
    userData                UserData                  OPTIONAL,
    cause                   EventCause,
    servicesPermitted       [2] IMPLICIT ServicesPermitted  OPTIONAL,
    mediaCallCharacteristics [3] IMPLICIT MediaCallCharacteristics  OPTIONAL,
    callCharacteristics     [4] IMPLICIT CallCharacteristics  OPTIONAL,
    callLinkageDataList     [6] IMPLICIT CallLinkageDataList  OPTIONAL,
    extensions              [5] IMPLICIT CSTACommonArguments  OPTIONAL}

END -- of CSTA-conferenced-event

```

## 15.2.4 Connection cleared

```

CSTA-connection-cleared-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) connection-cleared-event( 24) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) }
ChargingInfo FROM CSTA-charge-info
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) charge-info( 133) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) };

ConnectionClearedEvent ::=
SEQUENCE
{
  droppedConnection          ConnectionID,
  releasingDevice            SubjectDeviceID,
  localConnectionInfo        LocalConnectionState          OPTIONAL,
  correlatorData              [0] IMPLICIT CorrelatorData    OPTIONAL,
  userData                    UserData                      OPTIONAL,
  chargingInfo                [1] IMPLICIT ChargingInfo      OPTIONAL,
  cause                       EventCause,
  servicesPermitted           [2] IMPLICIT ServicesPermitted  OPTIONAL,
  mediaCallCharacteristics     [3] IMPLICIT MediaCallCharacteristics  OPTIONAL,
  callCharacteristics         [4] IMPLICIT CallCharacteristics  OPTIONAL,
  droppedConnectionInfo       [5] IMPLICIT ConnectionInformation  OPTIONAL,
  callLinkageData             [6] IMPLICIT CallLinkageData      OPTIONAL,
  extensions                   CSTACommonArguments          OPTIONAL}
END -- of CSTA-connection-cleared-event

```

## 15.2.5 Delivered

```

CSTA-delivered-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) delivered-event( 25) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, CallingDeviceID, CalledDeviceID, RedirectionDeviceID,
AssociatedCalledDeviceID, AssociatedCallingDeviceID,
NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

DeliveredEvent ::=
SEQUENCE
{
    connection                ConnectionID,
    alertingDevice            SubjectDeviceID,
    callingDevice              CallingDeviceID,
    calledDevice               CalledDeviceID,
    lastRedirectionDevice      RedirectionDeviceID,
    originatingNIDConnection  ConnectionID                OPTIONAL,
    localConnectionInfo        LocalConnectionState            OPTIONAL,
    correlatorData              [0] IMPLICIT CorrelatorData      OPTIONAL,
    userData                    UserData                        OPTIONAL,
    cause                       EventCause,
    servicesPermitted           [1] IMPLICIT ServicesPermitted   OPTIONAL,
    networkCallingDevice        NetworkCallingDeviceID         OPTIONAL,
    networkCalledDevice         NetworkCalledDeviceID         OPTIONAL,
    associatedCallingDevice      AssociatedCallingDeviceID     OPTIONAL,
    associatedCalledDevice       AssociatedCalledDeviceID       OPTIONAL,
    mediaCallCharacteristics     [2] IMPLICIT MediaCallCharacteristics  OPTIONAL,
    callCharacteristics         [3] IMPLICIT CallCharacteristics  OPTIONAL,
    connectionInfo              [4] IMPLICIT ConnectionInformation  OPTIONAL,
    callLinkageData              [5] IMPLICIT CallLinkageData      OPTIONAL,
    extensions                   CSTACommonArguments            OPTIONAL}

END -- of CSTA-delivered-event

```

## 15.2.6 Digits dialed

```

CSTA-digits-dialed-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) digits-dialed-event( 225) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, DeviceID, AssociatedCalledDeviceID, AssociatedCallingDeviceID,
NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) };

DigitsDialedEvent ::= SEQUENCE
  {
    dialingConnection      ConnectionID,
    dialingDevice          SubjectDeviceID,
    dialingSequence        DeviceID,
    localConnectionInfo    LocalConnectionState      OPTIONAL,
    correlatorData         [0] IMPLICIT CorrelatorData  OPTIONAL,
    cause                  EventCause,
    servicesPermitted      [1] IMPLICIT ServicesPermitted  OPTIONAL,
    networkCallingDevice   NetworkCallingDeviceID    OPTIONAL,
    networkCalledDevice    NetworkCalledDeviceID      OPTIONAL,
    associatedCallingDevice AssociatedCallingDeviceID  OPTIONAL,
    associatedCalledDevice AssociatedCalledDeviceID    OPTIONAL,
    dialingConnectionInfo  [2] IMPLICIT ConnectionInformation  OPTIONAL,
    callCharacteristics     [3] IMPLICIT CallCharacteristics    OPTIONAL,
    callLinkageData        [4] IMPLICIT CallLinkageData        OPTIONAL,
    extensions              CSTACommonArguments        OPTIONAL}

END -- of CSTA-digits-dialed-event

```

## 15.2.7 Diverted

```

CSTA-diverted-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) diverted-event( 26) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID , CallingDeviceID, CalledDeviceID, RedirectionDeviceID,
    AssociatedCalledDeviceID, AssociatedCallingDeviceID,
    NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

DivertedEvent ::= SEQUENCE
{
    connection                ConnectionID,
    divertingDevice           SubjectDeviceID,
    newDestination            SubjectDeviceID,
    callingDevice             CallingDeviceID,
    calledDevice              CalledDeviceID,
    lastRedirectionDevice     RedirectionDeviceID,
    localConnectionInfo       LocalConnectionState,
    correlatorData            [0] IMPLICIT CorrelatorData,
    userData                  UserData,
    cause                     EventCause,
    servicesPermitted         [1] IMPLICIT ServicesPermitted,
    mediaCallCharacteristics  [2] IMPLICIT MediaCallCharacteristics,
    callCharacteristics       [3] IMPLICIT CallCharacteristics,
    connectionInfo           [4] IMPLICIT ConnectionInformation,
    networkCallingDevice      NetworkCallingDeviceID,
    networkCalledDevice       NetworkCalledDeviceID,
    associatedCallingDevice   AssociatedCallingDeviceID,
    associatedCalledDevice    AssociatedCalledDeviceID,
    callLinkageData          [5] IMPLICIT CallLinkageData,
    extensions                CSTACommonArguments
}
END -- of diverted-event

```

## 15.2.8 Established

```

CSTA-established-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) established-event( 27) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, CalledDeviceID, CallingDeviceID, RedirectionDeviceID,
AssociatedCalledDeviceID, AssociatedCallingDeviceID,
NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

EstablishedEvent ::=
SEQUENCE
{
    establishedConnection      ConnectionID,
    answeringDevice            SubjectDeviceID,
    callingDevice               CallingDeviceID,
    calledDevice                CalledDeviceID,
    lastRedirectionDevice       RedirectionDeviceID,
    originatingNIDConnection   ConnectionID                OPTIONAL,
    localConnectionInfo         LocalConnectionState        OPTIONAL,
    correlatorData               [1] IMPLICIT CorrelatorData  OPTIONAL,
    userData                     UserData                    OPTIONAL,
    cause                        EventCause,
    servicesPermitted            [2] IMPLICIT ServicesPermitted OPTIONAL,
    networkCallingDevice         NetworkCallingDeviceID      OPTIONAL,
    networkCalledDevice          NetworkCalledDeviceID        OPTIONAL,
    associatedCallingDevice       AssociatedCallingDeviceID    OPTIONAL,
    associatedCalledDevice        AssociatedCalledDeviceID     OPTIONAL,
    mediaCallCharacteristics      [3] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics           [4] IMPLICIT CallCharacteristics  OPTIONAL,
    establishedConnectionInfo     [5] IMPLICIT ConnectionInformation  OPTIONAL,
    callLinkageData              [6] IMPLICIT CallLinkageData      OPTIONAL,
    extensions                    CSTACommonArguments         OPTIONAL}

END -- of CSTA-established-event

```

## 15.2.9 Failed

```

CSTA-failed-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) failed-event( 28) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, CallingDeviceID, CalledDeviceID, RedirectionDeviceID,
AssociatedCalledDeviceID, AssociatedCallingDeviceID,
NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

FailedEvent ::= SEQUENCE
    {
        failedConnection          ConnectionID,
        failingDevice              SubjectDeviceID,
        callingDevice              CallingDeviceID,
        calledDevice              CalledDeviceID,
        lastRedirectionDevice      RedirectionDeviceID,
        originatingNIDConnection  ConnectionID          OPTIONAL,
        localConnectionInfo       LocalConnectionState  OPTIONAL,
        correlatorData            [0] IMPLICIT CorrelatorData  OPTIONAL,
        userData                  UserData              OPTIONAL,
        cause                     EventCause,
        servicesPermitted         [1] IMPLICIT ServicesPermitted  OPTIONAL,
        networkCallingDevice      NetworkCallingDeviceID  OPTIONAL,
        networkCalledDevice       NetworkCalledDeviceID  OPTIONAL,
        associatedCallingDevice    AssociatedCallingDeviceID  OPTIONAL,
        associatedCalledDevice     AssociatedCalledDeviceID  OPTIONAL,
        mediaCallCharacteristics   [2] IMPLICIT MediaCallCharacteristics  OPTIONAL,
        callCharacteristics        [3] IMPLICIT CallCharacteristics  OPTIONAL,
        establishedConnectionInfo [4] IMPLICIT ConnectionInformation  OPTIONAL,
        callLinkageData           [5] IMPLICIT CallLinkageData  OPTIONAL,
        extensions                 CSTACommonArguments      OPTIONAL}

END -- of CSTA-failed-event

```

## 15.2.10 Held

```

CSTA-held-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) held-event( 29) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

HeldEvent ::=
SEQUENCE
{
    heldConnection          ConnectionID,
    holdingDevice           SubjectDeviceID,
    localConnectionInfo     LocalConnectionState          OPTIONAL,
    correlatorData          [0] IMPLICIT CorrelatorData    OPTIONAL,
    cause                   EventCause,
    servicesPermitted       [1] IMPLICIT ServicesPermitted OPTIONAL,
    mediaCallCharacteristics [2] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics     [3] IMPLICIT CallCharacteristics OPTIONAL,
    heldConnectionInfo     [4] IMPLICIT ConnectionInformation OPTIONAL,
    callLinkageData        [5] IMPLICIT CallLinkageData    OPTIONAL,
    extensions              CSTACommonArguments           OPTIONAL}

END -- of CSTA-held-event

```

## 15.2.11 Network capabilities changed

```

CSTA-network-capabilities-changed-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) network-capabilities-changed-event( 226) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, CalledDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, NetworkCapability, ProgressIndicator,
ServicesPermitted FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) };

NetworkCapabilitiesChangedEvent ::= SEQUENCE
{
  outboundConnection      ConnectionID,
  networkInterfaceUsed    SubjectDeviceID,
  calledDevice            CalledDeviceID,
  progressIndicator       ProgressIndicator,
  localConnectionInfo     LocalConnectionState      OPTIONAL,
  correlatorData          CorrelatorData            OPTIONAL,
  userData                UserData                OPTIONAL,
  networkCapability       [0] IMPLICIT NetworkCapability OPTIONAL,
  cause                   EventCause,
  servicesPermitted       [1] IMPLICIT ServicesPermitted      OPTIONAL,
  mediaCallCharacteristics [2] IMPLICIT MediaCallCharacteristics OPTIONAL,
  callCharacteristics     [3] IMPLICIT CallCharacteristics    OPTIONAL,
  outboundConnectionInfo [4] IMPLICIT ConnectionInformation  OPTIONAL,
  callLinkageData         [5] IMPLICIT CallLinkageData        OPTIONAL,
  extensions              CSTACommonArguments        OPTIONAL}

END -- of CSTA-network-capabilities-changed-event

```

## 15.2.12 Network reached

```

CSTA-network-reached-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) network-reached-event( 30) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, CallingDeviceID, CalledDeviceID, RedirectionDeviceID,
AssociatedCallingDeviceID, NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, NetworkCapability,
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

NetworkReachedEvent ::=
SEQUENCE
{
    outboundConnection      ConnectionID,
    networkInterfaceUsed    SubjectDeviceID,
    callingDevice           CallingDeviceID,
    calledDevice            CalledDeviceID,
    lastRedirectionDevice   RedirectionDeviceID,
    originatingNIDConnectio ConnectionID          OPTIONAL,
    localConnectionInfo    LocalConnectionState  OPTIONAL,
    correlatorData         [0] IMPLICIT CorrelatorData  OPTIONAL,
    userData               UserData            OPTIONAL,
    networkCapability      [1] IMPLICIT NetworkCapability  OPTIONAL,
    cause                  EventCause,
    servicesPermitted     [2] IMPLICIT ServicesPermitted  OPTIONAL,
    mediaCallCharacteristics [3] IMPLICIT MediaCallCharacteristics  OPTIONAL,
    callCharacteristics    [4] IMPLICIT CallCharacteristics  OPTIONAL,
    outboundConnectionInfo [5] IMPLICIT ConnectionInformation  OPTIONAL,
    networkCallingDevice   NetworkCallingDeviceID  OPTIONAL,
    networkCalledDevice    NetworkCalledDeviceID  OPTIONAL,
    associatedCallingDevice AssociatedCallingDeviceID  OPTIONAL,
    callLinkageData        [6] IMPLICIT CallLinkageData  OPTIONAL,
    extensions             CSTACommonArguments  OPTIONAL}

END -- of CSTA-network-reached-event

```

## 15.2.13 Offered

```

CSTA-offered-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) offered-event( 227) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, CallingDeviceID, CalledDeviceID, RedirectionDeviceID, AssociatedCalledDeviceID,
AssociatedCallingDeviceID, NetworkCalledDeviceID, NetworkCallingDeviceID
FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

OfferedEvent ::= SEQUENCE
    {
        offeredConnection          ConnectionID,
        offeredDevice               SubjectDeviceID,
        callingDevice               CallingDeviceID,
        calledDevice                CalledDeviceID,
        lastRedirectionDevice       RedirectionDeviceID,
        originatingNIDConnection   ConnectionID                OPTIONAL,
        localConnectionInfo        LocalConnectionState        OPTIONAL,
        correlatorData              CorrelatorData              OPTIONAL,
        userData                    UserData                    OPTIONAL,
        cause                       EventCause,
        servicesPermitted           [0] IMPLICIT ServicesPermitted OPTIONAL,
        networkCallingDevice        NetworkCallingDeviceID     OPTIONAL,
        networkCalledDevice         NetworkCalledDeviceID     OPTIONAL,
        associatedCallingDevice     AssociatedCallingDeviceID   OPTIONAL,
        associatedCalledDevice      AssociatedCalledDeviceID    OPTIONAL,
        mediaCallCharacteristics    [1] IMPLICIT MediaCallCharacteristics OPTIONAL,
        callCharacteristics         [2] IMPLICIT CallCharacteristics    OPTIONAL,
        offeredConnectionInfo      [3] IMPLICIT ConnectionInformation OPTIONAL,
        callLinkageData             [4] IMPLICIT CallLinkageData      OPTIONAL,
        extensions                  CSTACommonArguments         OPTIONAL}

END -- of CSTA-offered-event

```

## 15.2.14 Originated

```

CSTA-originated-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) originated-event( 31) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID, CallingDeviceID, CalledDeviceID, AssociatedCalledDeviceID,
AssociatedCallingDeviceID, NetworkCalledDeviceID, NetworkCallingDeviceID, SubjectDeviceID
FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) };

OriginatedEvent ::= SEQUENCE
{
  originatedConnection      ConnectionID,
  callingDevice             SubjectDeviceID,
  calledDevice              CalledDeviceID,
  originatingDevice         DeviceID                               OPTIONAL,
  localConnectionInfo       LocalConnectionState                 OPTIONAL,
  correlatorData             [2] IMPLICIT CorrelatorData          OPTIONAL,
  cause                     EventCause,
  servicesPermitted          [3] IMPLICIT ServicesPermitted       OPTIONAL,
  networkCallingDevice       NetworkCallingDeviceID              OPTIONAL,
  networkCalledDevice        NetworkCalledDeviceID               OPTIONAL,
  associatedCallingDevice     AssociatedCallingDeviceID           OPTIONAL,
  associatedCalledDevice      AssociatedCalledDeviceID            OPTIONAL,
  mediaCallCharacteristics    [4] IMPLICIT MediaCallCharacteristics OPTIONAL,
  callCharacteristics        [5] IMPLICIT CallCharacteristics     OPTIONAL,
  originatedConnectionInfo   [6] IMPLICIT ConnectionInformation   OPTIONAL,
  callLinkageData            [7] IMPLICIT CallLinkageData         OPTIONAL,
  extensions                 CSTACommonArguments                 OPTIONAL}

END -- of CSTA-originated-event

```

## 15.2.15 Queued

```

CSTA-queued-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) queued-event( 32) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
CalledDeviceID, CallingDeviceID, RedirectionDeviceID, SubjectDeviceID, AssociatedCalledDeviceID,
AssociatedCallingDeviceID, NetworkCalledDeviceID, NetworkCallingDeviceID
    FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

QueuedEvent ::=
SEQUENCE
{
    queuedConnection      ConnectionID,
    queue                 SubjectDeviceID,
    callingDevice         CallingDeviceID,
    calledDevice          CalledDeviceID,
    lastRedirectionDevice RedirectionDeviceID,
    numberQueued          [0] IMPLICIT INTEGER          OPTIONAL,
    callsInFront          [1] IMPLICIT INTEGER          OPTIONAL,
    localConnectionInfo   LocalConnectionState         OPTIONAL,
    correlatorData        [2] IMPLICIT CorrelatorData   OPTIONAL,
    userData              UserData                     OPTIONAL,
    cause                 EventCause,
    servicesPermitted     [3] IMPLICIT ServicesPermitted OPTIONAL,
    networkCallingDevice  NetworkCallingDeviceID       OPTIONAL,
    networkCalledDevice   NetworkCalledDeviceID        OPTIONAL,
    associatedCallingDevice AssociatedCallingDeviceID   OPTIONAL,
    associatedCalledDevice AssociatedCalledDeviceID     OPTIONAL,
    mediaCallCharacteristics [4] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics   [5] IMPLICIT CallCharacteristics   OPTIONAL,
    queuedConnectionInfo [6] IMPLICIT ConnectionInformation   OPTIONAL,
    callLinkageData       [7] IMPLICIT CallLinkageData       OPTIONAL,
    extensions            CSTACommonArguments           OPTIONAL}

END -- of CSTA-queued-event

```

## 15.2.16 Retrieved

```

CSTA-retrieved-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) retrieved-event( 33) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

RetrievedEvent ::=
SEQUENCE
{
    retrievedConnection      ConnectionID,
    retrievingDevice         SubjectDeviceID,
    localConnectionInfo     LocalConnectionState      OPTIONAL,
    correlatorData          [0] IMPLICIT CorrelatorData      OPTIONAL,
    cause                   EventCause,
    servicesPermitted       [1] IMPLICIT ServicesPermitted    OPTIONAL,
    mediaCallCharacteristics [2] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics     [3] IMPLICIT CallCharacteristics  OPTIONAL,
    retrievedConnectionInfo [4] IMPLICIT ConnectionInformation OPTIONAL,
    callLinkageData         [5] IMPLICIT CallLinkageData      OPTIONAL,
    extensions              CSTACommonArguments           OPTIONAL}

END -- of CSTA-retrieved-event

```

## 15.2.17 Service initiated

```

CSTA-service-initiated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) service-initiated-event( 34) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID, AssociatedCallingDeviceID, NetworkCalledDeviceID,
NetworkCallingDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

ServiceInitiatedEvent ::=
SEQUENCE
{
    initiatedConnection      ConnectionID,
    initiatingDevice         SubjectDeviceID,
    localConnectionInfo     LocalConnectionState      OPTIONAL,
    correlatorData          [0] IMPLICIT CorrelatorData  OPTIONAL,
    cause                   EventCause,
    servicesPermitted       [1] IMPLICIT ServicesPermitted  OPTIONAL,
    mediaCallCharacteristics [2] IMPLICIT MediaCallCharacteristics  OPTIONAL,
    callCharacteristics     [3] IMPLICIT CallCharacteristics  OPTIONAL,
    initiatedConnectionInfo [4] IMPLICIT ConnectionInformation  OPTIONAL,
    networkCallingDevice    NetworkCallingDeviceID      OPTIONAL,
    networkCalledDevice     NetworkCalledDeviceID       OPTIONAL,
    associatedCallingDevice  AssociatedCallingDeviceID   OPTIONAL,
    callLinkageData         [5] IMPLICIT CallLinkageData    OPTIONAL,
    extensions              CSTACommonArguments         OPTIONAL}
END -- of CSTA-service-initiated-event

```

## 15.2.18 Transferred

```

CSTA-transferred-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) transferred-event( 35) }
DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState, ConnectionList FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageDataList, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ChargingInfo FROM CSTA-charge-info
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) charge-info( 133) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

TransferredEvent ::=
SEQUENCE
{
    primaryOldCall          ConnectionID,
    secondaryOldCall        [0] IMPLICIT ConnectionID          OPTIONAL,
    transferringDevice      SubjectDeviceID,
    transferredToDevice     SubjectDeviceID,
    transferredConnections  [1] IMPLICIT ConnectionList,
    localConnectionInfo    LocalConnectionState                OPTIONAL,
    correlatorData          [2] IMPLICIT CorrelatorData         OPTIONAL,
    userData                UserData                            OPTIONAL,
    chargingInfo            [3] IMPLICIT ChargingInfo           OPTIONAL,
    cause                   EventCause,
    servicesPermitted       [4] IMPLICIT ServicesPermitted     OPTIONAL,
    mediaCallCharacteristics [5] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics     [6] IMPLICIT CallCharacteristics   OPTIONAL,
    callLinkageDataList    [7] IMPLICIT CallLinkageDataList   OPTIONAL,
    extensions              CSTACommonArguments                OPTIONAL}

END -- of CSTA-transferred-event

```

## 16 Call associated features

### 16.1 Services

#### 16.1.1 Associate data

```

CSTA-associate-data
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) associate-data( 230) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
AccountInfo, AuthCode, CorrelatorData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallQualifyingData FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) };

associateData      OPERATION ::=
{
    ARGUMENT          AssociateDataArgument
    RESULT            AssociateDataResult
    ERRORS            {universalFailure}
    CODE              local: 230
}

AssociateDataArgument ::= SEQUENCE
{
    existingCall      ConnectionID,
    accountCode       [0] IMPLICIT AccountInfo          OPTIONAL,
    authCode          [1] IMPLICIT AuthCode             OPTIONAL,
    correlatorData    [2] IMPLICIT CorrelatorData        OPTIONAL,
    callQualifyingData [3] IMPLICIT CallQualifyingData   OPTIONAL,
    extensions        CSTACommonArguments              OPTIONAL}

AssociateDataResult ::=
CHOICE
{extensions         CSTACommonArguments,
 noData             NULL}

END -- of CSTA-associate-data

```

16.1.2 Cancel telephony tones

```

CSTA-cancel-telephony-tones
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) cancel-telephony-tones( 231) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

cancelTelephonyTones          OPERATION ::=
{
    ARGUMENT    CancelTelephonyTonesArgument
    RESULT      CancelTelephonyTonesResult
    ERRORS      {universalFailure}
    CODE        local: 231
}

CancelTelephonyTonesArgument ::= SEQUENCE
{
    connectionToStopTone    ConnectionID,
    extensions              CSTACCommonArguments    OPTIONAL}

CancelTelephonyTonesResult ::=
CHOICE
{extensions              CSTACCommonArguments,
 noData                  NULL}

END -- of CSTA-cancel-telephony-tones

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 16.1.3 Generate digits

```

CSTA-generate-digits
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) generate-digits( 232) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

generateDigits      OPERATION ::=
{
    ARGUMENT      GenerateDigitsArgument
    RESULT        GenerateDigitsResult
    ERRORS        {universalFailure}
    CODE          local: 232
}

GenerateDigitsArgument ::= SEQUENCE
{
    connectionToSendDigits  ConnectionID,
    digitMode                DigitMode                OPTIONAL,
    charactersToSend        IA5String,
    toneDuration            [0] IMPLICIT INTEGER      OPTIONAL,
    pulseRate               [1] IMPLICIT INTEGER      OPTIONAL,
    pauseDuration           [2] IMPLICIT INTEGER      OPTIONAL,
    extensions              CSTACommonArguments      OPTIONAL}

GenerateDigitsResult ::=
CHOICE
{
    extensions              CSTACommonArguments,
    noData                  NULL}

DigitMode ::= ENUMERATED
{
    rotaryPulse (0),
    dtmf        (1)}

END -- of CSTA-generate-digits

```

## 16.1.4 Generate telephony tones

```

CSTA-generate-telephony-tones
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) generate-telephony-tones( 233) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
TelephonyTone FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

generateTelephonyTones      OPERATION ::=
{
  ARGUMENT    GenerateTelephonyTonesArgument
  RESULT      GenerateTelephonyTonesResult
  ERRORS      {universalFailure}
  CODE        local: 233
}

GenerateTelephonyTonesArgument ::= SEQUENCE
{
  connectionToSendTone    ConnectionID,
  toneToSend              TelephonyTone,
  toneDuration            INTEGER                OPTIONAL,
  extensions               CSTACommonArguments  OPTIONAL}

GenerateTelephonyTonesResult ::=
CHOICE
{extensions      CSTACommonArguments,
 noData         NULL}

END -- of CSTA-generate-telephony-tones

```

STANDARDSISO.COM · Click to view the full PDF of ISO/IEC 18052:2000

## 16.1.5 Send user information

```

CSTA-send-user-information
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) send-user-information( 234) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

sendUserInfo      OPERATION ::=
{
    ARGUMENT      SendUserInfoArgument
    RESULT        SendUserInfoResult
    ERRORS        {universalFailure}
    CODE          local: 234
}

SendUserInfoArgument ::= SEQUENCE
{
    existingCall      ConnectionID,
    userData          UserData,
    extensions        CSTACommonArgumentsOPTIONAL}

SendUserInfoResult ::=
CHOICE
{extensions          CSTACommonArguments,
 noData              NULL}

END -- of CSTA-send-user-information

```

16.2 Events

16.2.1 Call information

```

CSTA-call-information-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-information-event( 41) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
CallingDeviceID, SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
AccountInfo, AuthCode, CorrelatorData, UserData FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallLinkageDataList, CallQualifyingData, ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

CallInformationEvent ::= SEQUENCE
{
    connection          ConnectionID,
    device              SubjectDeviceID,
    callingDevice       CallingDeviceID                OPTIONAL,
    accountInfo         [0] IMPLICIT AccountInfo       OPTIONAL,
    authCode            [1] IMPLICIT AuthCode          OPTIONAL,
    correlatorData      [2] IMPLICIT CorrelatorData    OPTIONAL,
    servicesPermitted   [3] IMPLICIT ServicesPermitted OPTIONAL,
    userData            UserData                      OPTIONAL,
    callQualifyingData  [4] IMPLICIT CallQualifyingData OPTIONAL,
    connectionInfo      ConnectionInformation          OPTIONAL,
    callLinkageDataList [5] IMPLICIT CallLinkageDataList OPTIONAL,
    extensions          CSTACommonArguments           OPTIONAL}

END -- of CSTA-call-information-event

```



## 16.2.2 Charging

```

CSTA-charging-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) charging-event( 240) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ChargingInfo FROM CSTA-charge-info
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) charge-info( 133) };

ChargingEvent ::= SEQUENCE
{
  connection          ConnectionID,
  chargedDevice       DeviceID,
  chargingInfo        ChargingInfo,
  cause               EventCause          OPTIONAL,
  extensions          CSTACommonArguments OPTIONAL}

END -- of CSTA-charging-event

```

16.2.3 Digits generated

```
CSTA-digits-generated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) digits-generated-event( 241) }
```

```
DEFINITIONS ::=
BEGIN
```

```
IMPORTS
```

```
-- Data Types --
```

```
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
```

```
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
```

```
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };
```

```
DigitsGeneratedEvent ::= SEQUENCE
```

```
{
    connection          ConnectionID,
    digitGeneratedList  IA5String,
    digitDurationList   [0] IMPLICIT SEQUENCE OF INTEGER OPTIONAL,
    pauseDurationList  [1] IMPLICIT SEQUENCE OF INTEGER OPTIONAL,
    connectionInfo      ConnectionInformation OPTIONAL,
    extensions          CSTACommonArguments OPTIONAL}
```

```
END -- of CSTA-digits-generated-event
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 16.2.4 Telephony tones generated

```

CSTA-telephony-tones-generated-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) telephony-tones-generated-event( 242) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
TelephonyTone FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

TelephonyTonesGeneratedEvent ::= SEQUENCE
{
    connection          ConnectionID,
    toneGenerated       TelephonyTone OPTIONAL,
    toneFrequency       [0] IMPLICIT INTEGER OPTIONAL,
    toneDuration        [1] IMPLICIT INTEGER OPTIONAL,
    pauseDuration       [2] IMPLICIT INTEGER OPTIONAL,
    connectionInfo      ConnectionInformation OPTIONAL,
    extensions          CSTACommonArguments OPTIONAL}

END -- of CSTA-telephony-tones-generated-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

16.2.5 Service completion failure

```

CSTA-service-completion-failure-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) service-completion-failure-event( 243) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
MediaCallCharacteristics, ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

ServiceCompletionFailureEvent ::= SEQUENCE
{
    primaryCall                PrimaryOrSecondaryCall,
    secondaryCall              [0] IMPLICIT PrimaryOrSecondaryCall    OPTIONAL,
    otherDevsPrimaryCallList   [1] IMPLICIT SEQUENCE OF OtherCall     OPTIONAL,
    otherDevsSecondaryCallList [2] IMPLICIT SEQUENCE OF OtherCall     OPTIONAL,
    mediaCallCharacteristics    [3] IMPLICIT MediaCallCharacteristics  OPTIONAL,
    cause                      EventCause,
    extensions                  CSTACCommonArguments                   OPTIONAL}

PrimaryOrSecondaryCall ::= SEQUENCE
{
    deviceID                    DeviceID,
    connectionID                ConnectionID,
    localConnectionState        LocalConnectionState,
    connectionInfo              ConnectionInformation    OPTIONAL}

OtherCall ::= SEQUENCE
{
    deviceID                    DeviceID,
    connectionID                ConnectionID,
    localConnectionState        LocalConnectionState    OPTIONAL,
    connectionInfo              ConnectionInformation    OPTIONAL}

END -- of CSTA-charging-event

```



## 17 Media attachment services and events

### 17.1 Services

#### 17.1.1 Attach media service

```

CSTA-attach-media-service
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) attach-media-service( 244) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
MediaServiceType, MediaServiceInstanceID, ConnectionMode,
ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

attachMediaService OPERATION ::=
{
    ARGUMENT    AttachMediaServiceArgument
    RESULT      AttachMediaServiceResult
    ERRORS      {universalFailure}
    CODE        local: 244
}

AttachMediaServiceArgument ::= SEQUENCE
{
    connection          ConnectionID,
    mediaServiceType   MediaServiceType,
    mediaServiceVersion INTEGER           OPTIONAL,
    mediaServiceInstanceID MediaServiceInstanceID OPTIONAL,
    connectionMode     ConnectionMode,
    requestedConnectionState LocalConnectionState OPTIONAL,
    extensions          CSTACommonArguments OPTIONAL}

AttachMediaServiceResult ::= SEQUENCE
{
    mediaConnection      ConnectionID           OPTIONAL,
    mediaDevice          [0] IMPLICIT DeviceID  OPTIONAL,
    mediaServiceInstanceID MediaServiceInstanceID OPTIONAL,
    mediaConnectionInfo [1] IMPLICIT ConnectionInformation OPTIONAL,
    extensions           CSTACommonArguments    OPTIONAL}

END -- of CSTA-attach-media-service

```

## 17.1.2 Detach media service

```

CSTA-detach-media-service
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) detach-media-service( 245) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
MediaServiceType FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) }

detachMediaService OPERATION ::=
{
  ARGUMENT    DetachMediaServiceArgument
  RESULT      DetachMediaServiceResult
  ERRORS      {universalFailure}
  CODE        local: 245
}

DetachMediaServiceArgument ::= SEQUENCE
{
  connection      ConnectionID,
  mediaServiceType MediaServiceType,
  extensions       CSTACommonArguments OPTIONAL}

DetachMediaServiceResult ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData      NULL}

END -- of CSTA-detach-media-service

```

## 17.2 Events

### 17.2.1 Media attached

```

CSTA-media-attached-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-attached-event( 246) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
MediaServiceType, MediaServiceInstanceID, MediaStreamID,
MediaCallCharacteristics, ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

MediaAttachedEvent ::= SEQUENCE
{
    mediaConnection          ConnectionID,
    mediaDevice              SubjectDeviceID,
    mediaServiceType         MediaServiceType,
    mediaServiceVersion      INTEGER OPTIONAL,
    mediaServiceInstanceID  [0] IMPLICIT MediaServiceInstanceID OPTIONAL,
    mediaStreamID            [1] IMPLICIT MediaStreamID OPTIONAL,
    mediaCallCharacteristics [2] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics      CallCharacteristics OPTIONAL,
    localConnectionInfo     LocalConnectionState OPTIONAL,
    mediaConnectionInfo     [3] IMPLICIT ConnectionInformation OPTIONAL,
    extension                CSTACommonArguments OPTIONAL}

END -- of CSTA-media-attached-service

```

17.2.2 Media detached

```

CSTA-media-detached-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-detached-event( 247) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
LocalConnectionState FROM CSTA-connection-states
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) connection-states( 125) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
MediaServiceType, MediaServiceInstanceID, MediaStreamID,
MediaCallCharacteristics, ConnectionInformation FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

MediaDetachedEvent ::= SEQUENCE
{
    mediaConnection          ConnectionID,
    mediaDevice              SubjectDeviceID,
    mediaServiceType         MediaServiceType,
    mediaServiceVersion      INTEGER OPTIONAL,
    mediaServiceInstanceID   [0] IMPLICIT MediaServiceInstanceID OPTIONAL,
    mediaStreamID            [1] IMPLICIT MediaStreamID OPTIONAL,
    mediaCallCharacteristics [2] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics      CallCharacteristics OPTIONAL,
    localConnectionInfo     LocalConnectionState OPTIONAL,
    mediaConnectionInfo     [3] IMPLICIT ConnectionInformation OPTIONAL,
    extension                CSTACommonArguments OPTIONAL}

END -- of CSTA-media-detached-service

```



## 18 Routing services

### 18.1 Registration services

#### 18.1.1 Route register

```

CSTA-route-register
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) route-register( 248) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
RouteRegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
MediaClass FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

routeRegister OPERATION ::=
{
    ARGUMENT    RouteRegisterArgument
    RESULT      RouteRegisterResult
    ERRORS      {universalFailure}
    CODE        local: 248
}

RouteRegisterArgument ::= SEQUENCE
{
    routeingDevice           DeviceID           OPTIONAL,
    requestedRouteingMediaClass MediaClass      OPTIONAL,
    extensions                CSTACommonArguments OPTIONAL}

RouteRegisterResult ::= SEQUENCE
{
    routeRegisterReqID      RouteRegisterReqID,
    actualRouteingMediaClassMediaClass      OPTIONAL,
    extensions                CSTACommonArguments OPTIONAL}

END -- of CSTA-route-register

```

18.1.2 Route register abort

```

CSTA-route-register-abort
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) route-register-abort( 249) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
RouteRegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

routeRegisterAbort OPERATION ::=
{
    ARGUMENT          RouteRegisterAbortArgument
    ERRORS             {universalFailure}
    ALWAYS RESPONDS   FALSE
    CODE               local: 249
}

RouteRegisterAbortArgument ::= SEQUENCE
{
    routeRegisterReqID  RouteRegisterReqID,
    extensions           CSTACommonArguments OPTIONAL}

END -- of CSTA-route-register-abort

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 18.1.3 Route register cancel

```

CSTA-route-register-cancel
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) route-register-cancel( 250) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
RouteRegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }

routeRegisterCancel OPERATION ::=
{
    ARGUMENT    RouteRegisterCancelArgument
    RESULT      RouteRegisterCancelResult
    ERRORS      {universalFailure}
    CODE        local: 250
}

RouteRegisterCancelArgument ::= SEQUENCE
{
    routeRegisterReqID    RouteRegisterReqID,
    extensions            CSTACommonArguments    OPTIONAL}

RouteRegisterCancelResult ::= CHOICE
{
    extensions    CSTACommonArguments,
    noData        NULL}

END -- of CSTA-route-register-cancel

```

18.2 Services

18.2.1 Re-Route

```

CSTA-re-route-request
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) re-route-request( 82) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
CorrelatorData, RouteRegisterReqID, RouteingCrossRefID
    FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallLinkageData FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) };

reRouteRequest OPERATION ::=
{
    ARGUMENT          ReRouteRequestArgument
    ERRORS             {universalFailure}
    ALWAYS RESPONDS   FALSE
    CODE               local: 32
}

ReRouteRequestArgument ::= SEQUENCE
{
    crossRefIdentifier    RouteingCrossRefID,
    routeRegisterReqID    [0] IMPLICIT RouteRegisterReqID    OPTIONAL,
    replyTimeout          [1] IMPLICIT INTEGER                OPTIONAL,
    correlatorData        [2] IMPLICIT CorrelatorData         OPTIONAL,
    callLinkageData       [3] IMPLICIT CallLinkageData         OPTIONAL,
    extensions            CSTACommonArguments                 OPTIONAL}

END -- of CSTA-re-route-request

```

STANDARDSISO.COM - Click to view the full PDF of ISO/IEC 18052:2000

## 18.2.2 Route end

```

CSTA-route-end-request
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) route-end-request( 85) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure, ErrorValue FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CorrelatorData, RouteRegisterReqID, RouteingCrossRefID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

routeEndRequest OPERATION ::=
{
    ARGUMENT          RouteEndRequestArgument
    ERRORS             {universalFailure}
    ALWAYS RESPONDS   FALSE
    CODE               local: 35
}

RouteEndRequestArgument ::= SEQUENCE {
    crossRefIdentifier    RouteingCrossRefID,
    routeRegisterReqID   [ 0] IMPLICIT RouteRegisterReqID      OPTIONAL,
    errorValue            [ 1] ErrorValue                       OPTIONAL,
    correlatorData        [ 2] IMPLICIT CorrelatorData          OPTIONAL,
    extensions            CSTACommonArguments                  OPTIONAL}

END -- of CSTA-route-end-request

```

## 18.2.3 Route reject

```

CSTA-route-reject
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) route-reject( 86) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CorrelatorData, RouteRegisterReqID, RouteingCrossRefID
  FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

routeReject OPERATION ::=
{
  ARGUMENT          RouteRejectArgument
  ERRORS            {universalFailure}
  ALWAYS RESPONDS  FALSE
  CODE              local: 36
}

RouteRejectArgument ::=
  SEQUENCE
  {crossRefIdentifier      RouteingCrossRefID,
   routeRegisterReqID     [0] IMPLICIT RouteRegisterReqID OPTIONAL,
   rejectCause             [1] IMPLICIT RejectCause          OPTIONAL,
   correlatorData         [2] IMPLICIT CorrelatorData        OPTIONAL,
   extensions              CSTACommonArguments              OPTIONAL}

RejectCause ::= ENUMERATED
{
  busyOverflow             (1),
  queueTimeOverflow       (2),
  capacityOverflow        (3),
  calendarOverflow        (4),
  unknownOverflow         (5)}

END -- of CSTA-route-reject

```

## 18.2.4 Route request

```

CSTA-route-request
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) route-request( 81) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
AssociatedCallingDeviceID, AssociatedCalledDeviceID, CalledDeviceID,
CallingDeviceID, SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
CorrelatorData, SelectValue, RouteRegisterReqID, RouteingCrossRefID
FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallCharacteristics, CallLinkageData FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
ConnectionInformation, MediaCallCharacteristics FROM CSTA-media-services
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) media-services( 136) };

routeRequest OPERATION ::=
{
    ARGUMENT      RouteRequestArgument
    ERRORS        { universalFailure }
    ALWAYS RESPONDS  FALSE
    CODE          local: 31
}

RouteRequestArgument ::= SEQUENCE
{
    crossRefIdentifier      RouteingCrossRefID,
    routeRegisterReqID     [0] IMPLICIT RouteRegisterReqID      OPTIONAL,
    currentRoute            CalledDeviceID,
    callingDevice           CallingDeviceID                      OPTIONAL,
    routeingDevice         SubjectDeviceID                      OPTIONAL,
    routedCall              ConnectionID                        OPTIONAL,
    routeSelAlgorithm       [1] IMPLICIT SelectValue            OPTIONAL,
    associatedCallingDevice AssociatedCallingDeviceID           OPTIONAL,
    associatedCalledDevice  AssociatedCalledDeviceID            OPTIONAL,
    priority                [2] IMPLICIT BOOLEAN                OPTIONAL,
    replyTimeout            [3] IMPLICIT INTEGER                 OPTIONAL,
    correlatorData          [4] IMPLICIT CorrelatorData         OPTIONAL,
    mediaCallCharacteristics [5] IMPLICIT MediaCallCharacteristics OPTIONAL,
    callCharacteristics     [6] IMPLICIT CallCharacteristics    OPTIONAL,
    routeCallInfo           [7] IMPLICIT ConnectionInformation  OPTIONAL,
    callLinkageData         [8] IMPLICIT CallLinkageData        OPTIONAL,
    extensions              CSTACommonArguments                 OPTIONAL
}

END -- of CSTA-route-request

```

18.2.5 Route select

```

CSTA-route-select-request
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) route-select-request( 83) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CorrelatorData, RetryValue, RouteingCrossRefID, RouteRegisterReqID
FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }

routeSelectRequest      OPERATION ::=
{
    ARGUMENT              RouteSelectRequestArgument
    ERRORS                 {universalFailure}
    ALWAYS RESPONDS      FALSE
    CODE                   local: 33
}

RouteSelectRequestArgument ::=
SEQUENCE
{crossRefIdentifier      RouteingCrossRefID,
 routeRegisterReqID     [0] IMPLICIT RouteRegisterReqID           OPTIONAL,
 routeSelected           [1] DeviceID,
 alternateRoutes         [2] IMPLICIT SEQUENCE OF DeviceID       OPTIONAL,
 remainRetries           [3] RetryValue                           OPTIONAL,
 routeUsedReq            [4] IMPLICIT BOOLEAN                     OPTIONAL,
 correlatorData          [5] IMPLICIT CorrelatorData              OPTIONAL,
 extensions              CSTACommonArguments                      OPTIONAL}

END -- of CSTA-route-select-request

```



## 18.2.6 Route used

```

CSTA-route-used-request
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) route-used-request( 84) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CallingDeviceID, CalledDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CorrelatorData, RouteRegisterReqID, RouteingCrossRefID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CallLinkageData FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
;

routeUsedRequest      OPERATION ::=
{
    ARGUMENT          RouteUsedRequestArgument
    ERRORS             { universalFailure }
    ALWAYS RESPONDS   FALSE
    CODE               local: 34
}

RouteUsedRequestArgument ::= SEQUENCE
{
    crossRefIdentifier  RouteingCrossRefID,
    routeRegisterReqID [0] IMPLICIT RouteRegisterReqID    OPTIONAL,
    routeUsed           CalledDeviceID,
    callingDevice       CallingDeviceID                    OPTIONAL,
    domain              BOOLEAN                            OPTIONAL,
    correlatorData      [1] IMPLICIT CorrelatorData        OPTIONAL,
    callLinkageData     [2] IMPLICIT CallLinkageData        OPTIONAL,
    extensions          CSTACommonArguments                OPTIONAL
}

END -- of CSTA-route-used-request

```

**19 Physical device features****19.1 Services****19.1.1 Button press**

```

CSTA-button-press
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) button-press( 260) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ButtonID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

buttonPress OPERATION ::=
{
  ARGUMENT    ButtonPressArgument
  RESULT      ButtonPressResult
  ERRORS      {universalFailure}
  CODE        local: 260
}

ButtonPressArgument ::= SEQUENCE
{
  device      DeviceID,
  button      ButtonID,
  extensions  CSTACommonArguments OPTIONAL}

ButtonPressResult ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData      NULL}

END -- of CSTA-button-press

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 19.1.2 Get auditory apparatus information

```

CSTA-get-auditory-apparatus-information
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-auditory-apparatus-information( 261) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID, AuditoryApparatusList FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

getAuditoryApparatusInformation OPERATION ::=
{
  ARGUMENT    GetAuditoryApparatusInformationArgument
  RESULT      GetAuditoryApparatusInformationResult
  ERRORS      {universalFailure}
  CODE        local: 261
}

GetAuditoryApparatusInformationArgument ::= SEQUENCE
{
  device          DeviceID,
  auditoryApparatus  AuditoryApparatusID  OPTIONAL,
  extensions      CSTACommonArguments  OPTIONAL}

GetAuditoryApparatusInformationResult ::= SEQUENCE
{
  auditoryApparatusList  AuditoryApparatusList,
  extensions            CSTACommonArguments  OPTIONAL}

END -- of CSTA-get-auditory-apparatus-information

```

## 19.1.3 Get button information

```

CSTA-get-button-information
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-button-information( 262) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ButtonID, LampID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

getButtonInformation OPERATION ::=
{
  ARGUMENT   GetButtonInformationArgument
  RESULT     GetButtonInformationResult
  ERRORS     {universalFailure}
  CODE       local: 262
}

GetButtonInformationArgument ::= SEQUENCE
{
  device      DeviceID,
  button      ButtonID OPTIONAL,
  extensions  CSTACommonArguments OPTIONAL}

GetButtonInformationResult ::= SEQUENCE
{
  buttonList  ButtonList,
  extensions  CSTACommonArguments OPTIONAL}

ButtonList ::= SEQUENCE OF SEQUENCE
{
  button          ButtonID,
  buttonLabel     [0] IMPLICIT IA5String OPTIONAL,
  buttonLabelSettable [1] IMPLICIT BOOLEAN OPTIONAL,
  buttonFunction  [2] IMPLICIT IA5String OPTIONAL,
  buttonAssociatedNumber [3] IMPLICIT DeviceID OPTIONAL,
  buttonAssociatedNumberSettable [4] IMPLICIT BOOLEAN OPTIONAL,
  buttonPressIndicator [5] IMPLICIT BOOLEAN OPTIONAL,
  lampList       SEQUENCE OF LampID OPTIONAL}

END -- of CSTA-get-button-information

```

## 19.1.4 Get display

```

CSTA-get-display
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-display( 263) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DisplayID, CharacterSet FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

getDisplay OPERATION ::=
{
  ARGUMENT      GetDisplayArgument
  RESULT        GetDisplayResult
  ERRORS        {universalFailure}
  CODE          local: 263
}

GetDisplayArgument ::= SEQUENCE
{
  device        DeviceID,
  display       DisplayID          OPTIONAL,
  extensions    CSTACommonArguments  OPTIONAL}

GetDisplayResult ::= SEQUENCE
{
  displayList   DisplayList,
  extensions    CSTACommonArguments  OPTIONAL}

DisplayList ::= SEQUENCE OF SEQUENCE
{
  displayID      DisplayID,
  logicalRows   INTEGER,
  logicalColumns INTEGER,
  physicalRows  [0] IMPLICIT INTEGER  OPTIONAL,
  physicalColumns [1] IMPLICIT INTEGER  OPTIONAL,
  physicalBaseRowNumber [2] IMPLICIT INTEGER  OPTIONAL,
  physicalBaseColumnNumber [3] IMPLICIT INTEGER  OPTIONAL,
  characterSet  CharacterSet          OPTIONAL,
  contentsOfDisplay IA5String}
END -- of CSTA-get-display

```

## 19.1.5 Get hookswitch status

```

CSTA-get-hookswitch-status
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-hookswitch-status( 264) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
HookswitchID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

getHookswitchStatus OPERATION ::=
{
  ARGUMENT    GetHookswitchStatusArgument
  RESULT      GetHookswitchStatusResult
  ERRORS      {universalFailure}
  CODE        local: 264
}

GetHookswitchStatusArgument ::= SEQUENCE
{
  device      DeviceID,
  hookswitch  HookswitchID OPTIONAL,
  extensions  CSTACommonArguments OPTIONAL}

GetHookswitchStatusResult ::= SEQUENCE
{
  hookswitchStatusList  HookswitchStatusList,
  extensions            CSTACommonArguments OPTIONAL}

HookswitchStatusList ::= SEQUENCE OF SEQUENCE
{
  hookswitch      HookswitchID,
  hookswitchOnHook  BOOLEAN}

END -- of CSTA-get-hookswitch-status

```

STANDARDSISO.COM Click to view the full PDF of ISO/IEC 18052:2000

## 19.1.6 Get lamp information

```

CSTA-get-lamp-information
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-lamp-information( 265) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ButtonID, LampID, LampColor FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

getLampInformation OPERATION ::=
{
  ARGUMENT    GetLampInformationArgument
  RESULT      GetLampInformationResult
  ERRORS      {universalFailure}
  CODE        local: 265
}

GetLampInformationArgument ::= SEQUENCE
{
  device      DeviceID,
  lamp        LampID          OPTIONAL,
  extensions  CSTACommonArguments  OPTIONAL}

GetLampInformationResult ::= SEQUENCE
{
  lampList    LampList,
  extensions  CSTACommonArguments  OPTIONAL}

LampList ::= SEQUENCE OF SEQUENCE
{
  lamp        LampID,
  lampLabel   [0] IMPLICIT OCTET STRING          OPTIONAL,
  button      [1] IMPLICIT ButtonID              OPTIONAL,
  lampColor   LampColor                          OPTIONAL}

END -- of CSTA-get-lamp-information

```

## 19.1.7 Get lamp mode

```

CSTA-get-lamp-mode
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-lamp-mode( 266) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ButtonID, LampID, LampColor, LampMode, LampBrightness
FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

getLampMode OPERATION ::=
{
  ARGUMENT    GetLampModeArgument
  RESULT      GetLampModeResult
  ERRORS      {universalFailure}
  CODE        local: 266
}

GetLampModeArgument ::= SEQUENCE
{
  device      DeviceID,
  lamp        LampID          OPTIONAL,
  extensions  CSTACommonArguments  OPTIONAL}

GetLampModeResult ::= SEQUENCE
{
  lampModeList  LampModeList,
  lamp          LampID          OPTIONAL,
  extensions    CSTACommonArguments  OPTIONAL}

LampModeList ::= SEQUENCE OF SEQUENCE
{
  lamp          LampID,
  lampMode      LampMode,
  lampBrightness [0] IMPLICIT LampBrightness  OPTIONAL,
  lampColor      [1] IMPLICIT LampColor        OPTIONAL,
  button         [2] IMPLICIT ButtonID        OPTIONAL}

END -- of CSTA-get-lamp-mode

```

STANDARDSISO.COM · Click to view the full PDF of ISO/IEC 18052:2000

## 19.1.8 Get message waiting indicator

```

CSTA-get-message-waiting-indicator
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-message-waiting-indicator( 267) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

getMessageWaitingIndicator OPERATION ::=
{
  ARGUMENT  GetMessageWaitingIndicatorArgument
  RESULT    GetMessageWaitingIndicatorResult
  ERRORS    {universalFailure}
  CODE      local: 267
}

GetMessageWaitingIndicatorArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetMessageWaitingIndicatorResult ::= SEQUENCE
{
  messageWaitingOn    BOOLEAN,
  deviceForMessage    DeviceID          OPTIONAL,
  lampIsPresent       BOOLEAN          OPTIONAL,
  extensions          CSTACommonArguments OPTIONAL}

END -- of CSTA-get-message-waiting-indicator

```

## 19.1.9 Get microphone gain

```

CSTA-get-microphone-gain
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-microphone-gain( 268) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID, MicGainAbs FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

getMicrophoneGain OPERATION ::=
{
  ARGUMENT    GetMicrophoneGainArgument
  RESULT      GetMicrophoneGainResult
  ERRORS      {universalFailure}
  CODE        local: 268
}

GetMicrophoneGainArgument ::= SEQUENCE
{
  device          DeviceID,
  auditoryApparatus AuditoryApparatusID OPTIONAL,
  extensions      CSTACommonArguments OPTIONAL}

GetMicrophoneGainResult ::= SEQUENCE
{
  microphoneGainList MicrophoneGainList,
  extensions          CSTACommonArguments OPTIONAL}

MicrophoneGainList ::= SEQUENCE OF SEQUENCE
{
  auditoryApparatus AuditoryApparatusID,
  micGainAbs        MicGainAbs          OPTIONAL}

END -- of CSTA-get-microphone-gain

```

STANDARDSISO.COM Click to view the full PDF of ISO/IEC 18052:2000

## 19.1.10 Get microphone mute

```

CSTA-get-microphone-mute
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-microphone-mute( 269) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

getMicrophoneMute OPERATION ::=
{
  ARGUMENT    GetMicrophoneMuteArgument
  RESULT      GetMicrophoneMuteResult
  ERRORS      {universalFailure}
  CODE        local: 269
}

GetMicrophoneMuteArgument ::= SEQUENCE
{
  device                DeviceID,
  auditoryApparatus     AuditoryApparatusID    OPTIONAL,
  extensions             CSTACommonArguments   OPTIONAL}

GetMicrophoneMuteResult ::= SEQUENCE
{
  microphoneMuteList    MicrophoneMuteList,
  extensions             CSTACommonArguments   OPTIONAL}

MicrophoneMuteList ::= SEQUENCE OF SEQUENCE
{
  auditoryApparatus     AuditoryApparatusID,
  microphoneMuteOn      BOOLEAN}

END -- of CSTA-get-microphone-mute

```

## 19.1.11 Get ringer status

```

CSTA-get-ringer-status
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-ringer-status( 270) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
RingerID, RingMode FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

getRingerStatus OPERATION ::=
{
  ARGUMENT    GetRingerStatusArgument
  RESULT      GetRingerStatusResult
  ERRORS      {universalFailure}
  CODE        local: 270
}

GetRingerStatusArgument ::= SEQUENCE
{
  device      DeviceID,
  ringer      RingerID OPTIONAL,
  extensions  CSTACommonArguments OPTIONAL}

GetRingerStatusResult ::= SEQUENCE
{
  ringerStatusList  RingerStatusList,
  extensions        CSTACommonArguments OPTIONAL}

RingerStatusList ::= SEQUENCE OF SEQUENCE
{
  ringer      RingerID,
  ringMode    RingMode,
  ringCount   [0] IMPLICIT INTEGER (0..1000)    OPTIONAL,
  ringPattern [1] IMPLICIT INTEGER              OPTIONAL,
  ringVolAbs  [2] IMPLICIT INTEGER (0..100)     OPTIONAL}

END -- of CSTA-get-ringer-status

```

## 19.1.12 Get speaker mute

```

CSTA-get-speaker-mute
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-speaker-mute( 271) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

getSpeakerMute OPERATION ::=
{
  ARGUMENT    GetSpeakerMuteArgument
  RESULT      GetSpeakerMuteResult
  ERRORS      {universalFailure}
  CODE        local: 271
}

GetSpeakerMuteArgument ::= SEQUENCE
{
  device                DeviceID,
  auditoryApparatus    AuditoryApparatusID    OPTIONAL,
  extensions            CSTACommonArguments    OPTIONAL}

GetSpeakerMuteResult ::= SEQUENCE
{
  speakerMuteList      SpeakerMuteList,
  extensions            CSTACommonArguments    OPTIONAL}

SpeakerMuteList ::= SEQUENCE OF SEQUENCE
{
  auditoryApparatus    AuditoryApparatusID,
  speakerMuteOn        BOOLEAN}

END -- of CSTA-get-speaker-mute

```

19.1.13 Get speaker volume

```

CSTA-get-speaker-volume
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-speaker-volume( 272) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID, VolAbs FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

getSpeakerVolume OPERATION ::=
{
  ARGUMENT    GetSpeakerVolumeArgument
  RESULT      GetSpeakerVolumeResult
  ERRORS      {universalFailure}
  CODE        local: 272
}

GetSpeakerVolumeArgument ::= SEQUENCE
{
  device          DeviceID,
  auditoryApparatus AuditoryApparatusID OPTIONAL,
  extensions      CSTACommonArguments OPTIONAL}

GetSpeakerVolumeResult ::= SEQUENCE
{
  speakerVolumeList SpeakerVolumeList,
  extensions         CSTACommonArguments OPTIONAL}

SpeakerVolumeList ::= SEQUENCE OF SEQUENCE
{
  auditoryApparatus AuditoryApparatusID,
  speakerVolAbs     VolAbs OPTIONAL}

END -- of CSTA-get-speaker-volume

```

STANDARDSISO.COM Click to view the full PDF of ISO/IEC 18052:2000

## 19.1.14 Set button information

```

CSTA-set-button-information
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-button-information( 273) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ButtonID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

setButtonInformation OPERATION ::=
{
  ARGUMENT      SetButtonInformationArgument
  RESULT        SetButtonInformationResult
  ERRORS        {universalFailure}
  CODE          local: 273
}

SetButtonInformationArgument ::= SEQUENCE
{
  device          DeviceID,
  button          ButtonID,
  buttonLabel     IA5String (SIZE(0..64)) OPTIONAL,
  buttonAssociatedNumber DeviceID OPTIONAL,
  extensions      CSTACommonArguments OPTIONAL}

SetButtonInformationResult ::= CHOICE
{
  extensions      CSTACommonArguments,
  noData          NULL}

END -- of CSTA-set-button-information

```

19.1.15 Set display

```

CSTA-set-display
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-display( 274) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DisplayID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

setDisplay OPERATION ::=
{
  ARGUMENT      SetDisplayArgument
  RESULT        SetDisplayResult
  ERRORS        {universalFailure}
  CODE          local: 274
}

SetDisplayArgument ::= SEQUENCE
{
  device          DeviceID,
  display         DisplayID
  physicalBaseRowNumber [0] IMPLICIT INTEGER OPTIONAL,
  physicalBaseColumnNumber [1] IMPLICIT INTEGER OPTIONAL,
  contentsOfDisplay IA5String (SIZE(0..240)),
  offset         [2] IMPLICIT INTEGER OPTIONAL,
  extensions     CSTACommonArguments OPTIONAL}

SetDisplayResult ::= CHOICE
{
  extensions     CSTACommonArguments,
  noData        NULL}

END -- of CSTA-set-display

```

STANDARDS150.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 19.1.16 Set hookswitch status

```

CSTA-set-hookswitch-status
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-hookswitch-status( 275) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
HookswitchID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

setHookswitchStatus OPERATION ::=
{
  ARGUMENT      SetHookswitchStatusArgument
  RESULT        SetHookswitchStatusResult
  ERRORS        {universalFailure}
  CODE          local: 275
}

SetHookswitchStatusArgument ::= SEQUENCE
{
  device          DeviceID,
  hookswitch      HookswitchID,
  hookswitchOnHook  BOOLEAN,
  extensions      CSTACommonArguments OPTIONAL}

SetHookswitchStatusResult ::= CHOICE
{
  extensions      CSTACommonArguments,
  noData          NULL}

END -- of CSTA-set-hookswitch-status

```

## 19.1.17 Set lamp mode

```

CSTA-set-lamp-mode
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-lamp-mode( 276) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
LampID, LampColor, LampMode, LampBrightness
FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

setLampMode OPERATION ::=
{
  ARGUMENT      SetLampModeArgument
  RESULT        SetLampModeResult
  ERRORS        {universalFailure}
  CODE          local: 276
}

SetLampModeArgument ::= SEQUENCE
{
  device          DeviceID,
  lamp            LampID,
  lampMode        LampMode,
  lampBrightness  LampBrightness    OPTIONAL,
  lampColor       LampColor         OPTIONAL,
  extensions      CSTACommonArguments OPTIONAL}

SetLampModeResult ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData      NULL}

END -- of CSTA-set-lamp-mode

```

## 19.1.18 Set message waiting indicator

```

CSTA-set-message-waiting-indicator
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-message-waiting-indicator( 277) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

setMessageWaitingIndicator OPERATION ::=
{
  ARGUMENT    SetMessageWaitingIndicatorArgument
  RESULT      SetMessageWaitingIndicatorResult
  ERRORS      {universalFailure}
  CODE        local: 277
}

SetMessageWaitingIndicatorArgument ::= SEQUENCE
{
  device                DeviceID,
  messageWaitingOn      BOOLEAN,
  deviceForMessage      DeviceID          OPTIONAL,
  extensions             CSTACommonArguments  OPTIONAL}

SetMessageWaitingIndicatorResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData      NULL}

END -- of CSTA-set-message-waiting-indicator

```

## 19.1.19 Set microphone gain

```

CSTA-set-microphone-gain
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-microphone-gain( 278) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID, MicrophoneGain FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

setMicrophoneGain OPERATION ::=
{
  ARGUMENT    SetMicrophoneGainArgument
  RESULT      SetMicrophoneGainResult
  ERRORS      {universalFailure}
  CODE        local: 278
}

SetMicrophoneGainArgument ::= SEQUENCE
{
  device          DeviceID,
  auditoryApparatus AuditoryApparatusID,
  microphoneGain  MicrophoneGain,
  extensions      CSTACommonArguments OPTIONAL}

SetMicrophoneGainResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-set-microphone-gain

```

## 19.1.20 Set microphone mute

```

CSTA-set-microphone-mute
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-microphone-mute( 279) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

setMicrophoneMute OPERATION ::=
{
  ARGUMENT    SetMicrophoneMuteArgument
  RESULT      SetMicrophoneMuteResult
  ERRORS      {universalFailure}
  CODE        local: 279
}

SetMicrophoneMuteArgument ::= SEQUENCE
{
  device          DeviceID,
  auditoryApparatus AuditoryApparatusID,
  microphoneMuteOn BOOLEAN,
  extensions      CSTACommonArguments OPTIONAL}

SetMicrophoneMuteResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData      NULL}

END -- of CSTA-set-microphone-mute

```

19.1.21 Set ringer status

```

CSTA-set-ringer-status
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-ringer-status( 280) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
RingerID, RingMode, Volume FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

setRingerStatus OPERATION ::=
{
  ARGUMENT    SetRingerStatusArgument
  RESULT      SetRingerStatusResult
  ERRORS      {universalFailure}
  CODE        local: 280
}

SetRingerStatusArgument ::= SEQUENCE
{
  device          DeviceID,
  ringer          RingerID,
  ringMode        RingMode,
  ringPattern     [1] IMPLICIT INTEGER,
  ringVolume      [2] Volume,
  extensions      CSTACommonArguments
}

SetRingerStatusResult ::= CHOICE
{
  extensions      CSTACommonArguments,
  noData          NULL}

END -- of CSTA-set-ringer-status

```

STANDARDSISO.COM: Click to view the full PDF of ISO/IEC 18052:2000

## 19.1.22 Set speaker mute

```

CSTA-set-speaker-mute
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-speaker-mute( 281) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

setSpeakerMute OPERATION ::=
{
  ARGUMENT      SetSpeakerMuteArgument
  RESULT        SetSpeakerMuteResult
  ERRORS        {universalFailure}
  CODE          local: 281
}

SetSpeakerMuteArgument ::= SEQUENCE
{
  device          DeviceID,
  auditoryApparatus AuditoryApparatusID,
  speakerMuteOn   BOOLEAN,
  extensions      CSTACommonArguments OPTIONAL}

SetSpeakerMuteResult ::= CHOICE
{
  extensions      CSTACommonArguments,
  noData          NULL}

END -- of CSTA-set-speaker-mute

```

## 19.1.23 Set speaker volume

```

CSTA-set-speaker-volume
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-speaker-volume( 282) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID, Volume FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) }

setSpeakerVolume OPERATION ::=
{
  ARGUMENT    SetSpeakerVolumeArgument
  RESULT      SetSpeakerVolumeResult
  ERRORS      {universalFailure}
  CODE        local: 282
}

SetSpeakerVolumeArgument ::= SEQUENCE
{
  device          DeviceID,
  auditoryApparatus AuditoryApparatusID,
  speakerVolume   Volume,
  extensions      CSTACommonArguments OPTIONAL}

SetSpeakerVolumeResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-set-speaker-volume

```

## 19.2 Events

### 19.2.1 Button information

```

CSTA-button-information-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) button-information-event( 283) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ButtonID FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) };

ButtonInformationEvent ::= SEQUENCE
{
    device                SubjectDeviceID,
    button                ButtonID,
    buttonLabel           IA5String (SIZE(0..64)) OPTIONAL,
    buttonAssociatedNumber DeviceID OPTIONAL,
    buttonPressIndicator  BOOLEAN OPTIONAL,
    extensions            CSTACommonArguments OPTIONAL}

END -- of CSTA-button-information-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

19.2.2 Button press

```

CSTA-button-press-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) button-press-event( 284) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ButtonID FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) }

ButtonPressEvent ::= SEQUENCE
{
    device                               SubjectDeviceID,
    button                               ButtonID,
    buttonLabel                          IA5String (SIZE(0..64)) OPTIONAL,
    buttonAssociatedNumber DeviceID      OPTIONAL,
    extensions                           CSTACCommonArguments OPTIONAL}

END -- of CSTA-button-press-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 19.2.3 Display updated

```

CSTA-display-updated-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) display-updated-event( 285) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DisplayID, CharacterSet FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

DisplayUpdatedEvent ::= SEQUENCE
{
  device                               SubjectDeviceID,
  displayID                             DisplayID           OPTIONAL,
  logicalRows                           INTEGER,
  logicalColumns                         INTEGER,
  physicalRows                           [0] IMPLICIT INTEGER OPTIONAL,
  physicalColumns                         [1] IMPLICIT INTEGER OPTIONAL,
  physicalBaseRowNumber                   [2] IMPLICIT INTEGER OPTIONAL,
  physicalBaseColumnNumber                [3] IMPLICIT INTEGER OPTIONAL,
  characterSet                            CharacterSet         OPTIONAL,
  contentsOfDisplay                       IA5String,
  extensions                              CSTACommonArguments OPTIONAL}

END -- of CSTA-display-updated-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

19.2.4 Hookswitch

```

CSTA-hookswitch-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) hookswitch-event( 286) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
HookswitchID FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) }

HookswitchEvent ::= SEQUENCE
{
    device                SubjectDeviceID,
    hookswitch            HookswitchID,
    hookswitchOnHook     BOOLEAN,
    extensions            CSTACommonArguments    OPTIONAL
}

END -- of CSTA-hookswitch-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 19.2.5 Lamp mode

```

CSTA-lamp-mode-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) lamp-mode-event( 287) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
LampID, LampMode, LampBrightness, LampColor
FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) };

LampModeEvent ::= SEQUENCE
{
    device                SubjectDeviceID,
    lamp                  LampID,
    lampLabel             OCTET STRING (SIZE(0..64))           OPTIONAL,
    lampMode              LampMode,
    lampBrightness        LampBrightness                       OPTIONAL,
    lampColor             LampColor                            OPTIONAL,
    extensions            CSTACommonArguments                 OPTIONAL}

END -- of CSTA-lamp-mode-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

19.2.6 Message waiting

```

CSTA-message-waiting-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) message-waiting-event( 44) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

MessageWaitingEvent ::=SEQUENCE
{
    targetDevice          SubjectDeviceID,
    deviceForMessage      DeviceID          OPTIONAL,
    messageWaitingOn      BOOLEAN,
    extensions            CSTACommonArguments  OPTIONAL}

END -- of CSTA-message-waiting-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**19.2.7 Microphone gain**

```

CSTA-microphone-gain-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) microphone-gain-event( 288) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID, MicrophoneGain FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) };

MicrophoneGainEvent ::= SEQUENCE
{
    invokingDevice          SubjectDeviceID,
    auditoryApparatus      AuditoryApparatusID,
    microphoneGain         MicrophoneGain,
    extensions             CSTACommonArguments    OPTIONAL}

END -- of CSTA-microphone-gain-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

19.2.8 Microphone mute

```

CSTA-microphone-mute-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) microphone-mute-event( 45) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) }

MicrophoneMuteEvent ::= SEQUENCE
{
    invokingDevice          SubjectDeviceID,
    auditoryApparatus      AuditoryApparatusID,
    microphoneMuteOn       BOOLEAN,
    extensions              CSTACommonArguments    OPTIONAL
}

END -- of CSTA-microphone-mute-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**19.2.9 Ringer status**

```

CSTA-ringer-status-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) ringer-status-event( 289) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
RingerID, RingMode, Volume FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) };

RingerStatusEvent ::= SEQUENCE
{
    device                SubjectDeviceID,
    ringer                RingerID,
    ringMode              RingMode                OPTIONAL,
    ringCount             [0] IMPLICIT INTEGER (0..1000)  OPTIONAL,
    ringPattern           [1] IMPLICIT INTEGER            OPTIONAL,
    ringVolume            [2] Volume                  OPTIONAL,
    extensions            CSTACommonArguments          OPTIONAL}

END -- of CSTA-ringer-status-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

19.2.10 Speaker mute

```

CSTA-speaker-mute-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) speaker-mute-event( 46) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) }

SpeakerMuteEvent ::= SEQUENCE
{
    invokingDevice          SubjectDeviceID,
    auditoryApparatus      AuditoryApparatusID,
    speakerMuteOn          BOOLEAN,
    extensions              CSTACommonArguments    OPTIONAL
}

END -- of CSTA-speaker-mute-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 19.2.11 Speaker volume

```

CSTA-speaker-volume-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) speaker-volume-event( 47) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
AuditoryApparatusID, Volume FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

SpeakerVolumeEvent ::= SEQUENCE
{
  invokingDevice          SubjectDeviceID,
  auditoryApparatus      AuditoryApparatusID,
  speakerVolume          Volume,
  extensions              CSTACommonArguments    OPTIONAL}

END -- of CSTA-speaker-volume-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20 Logical device features

### 20.1 Services

#### 20.1.1 Call back non-call-related

```

CSTA-call-back-non-call-related
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) call-back-non-call-related( 300) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

callBackNonCallRelated OPERATION ::=
{
  ARGUMENT    CallbackNonCallRelatedArgument
  RESULT      CallbackNonCallRelatedResult
  ERRORS      {universalFailure}
  CODE        local: 300
}

CallbackNonCallRelatedArgument ::= SEQUENCE
{
  originatingDevice    DeviceID,
  targetDevice         DeviceID,
  extensions            CSTACommonArguments OPTIONAL}

CallbackNonCallRelatedResult ::= CHOICE
{
  extensions    CSTACommonArguments,
  noData        NULL}

END -- of CSTA-call-back-non-call-related

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**20.1.2 Call back message non-call-related**

```

CSTA-call-back-message-non-call-related
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) call-back-message-non-call-related( 301) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

callBackMessageNonCallRelated OPERATION ::=
{
  ARGUMENT    CallBackMessageNonCallRelatedArgument
  RESULT      CallBackMessageNonCallRelatedResult
  ERRORS      {universalFailure}
  CODE        local: 301
}

CallBackMessageNonCallRelatedArgument ::= SEQUENCE
{
  originatingDevice    DeviceID,
  targetDevice         DeviceID,
  extensions           CSTACommonArguments OPTIONAL}

CallBackMessageNonCallRelatedResult ::= CHOICE
{
  extensions    CSTACommonArguments,
  noData       NULL}

END -- of CSTA-call-back-message-non-call-related

```

20.1.3 Cancel call back

```

CSTA-cancel-call-back
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) cancel-call-back( 302) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

cancelCallBack OPERATION ::=
{
  ARGUMENT    CancelCallBackArgument
  RESULT      CancelCallBackResult
  ERRORS      {universalFailure}
  CODE        local: 302
}

CancelCallBackArgument ::= SEQUENCE
{
  originatingDevice    DeviceID,
  targetDevice         DeviceID,
  extensions            CSTACommonArguments OPTIONAL}

CancelCallBackResult ::= CHOICE
{
  extensions    CSTACommonArguments,
  noData        NULL}

END -- of CSTA-cancel-call-back

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20.1.4 Cancel call back message

```

CSTA-cancel-call-back-message
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) cancel-call-back-message( 303) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

cancelCallbackMessage OPERATION ::=
{
  ARGUMENT    CancelCallbackMessageArgument
  RESULT      CancelCallbackMessageResult
  ERRORS      {universalFailure}
  CODE        local: 303
}

CancelCallbackMessageArgument ::= SEQUENCE
{
  originatingDevice    DeviceID,
  targetDevice         DeviceID,
  extensions            CSTACommonArguments OPTIONAL}

CancelCallbackMessageResult ::= CHOICE
{
  extensions    CSTACommonArguments,
  noData       NULL}

END -- of CSTA-cancel-call-back-message

```

## 20.1.5 Get agent state

```

CSTA-get-agent-state
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-agent-state( 304) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID, AgentState, PendingAgentState FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

getAgentState OPERATION ::=
{
  ARGUMENT    GetAgentStateArgument
  RESULT      GetAgentStateResult
  ERRORS      {universalFailure}
  CODE        local: 304
}

GetAgentStateArgument ::= SEQUENCE
{
  device      DeviceID,
  acdGroup    DeviceID          OPTIONAL,
  extensions  CSTACommonArguments  OPTIONAL}

GetAgentStateResult ::= SEQUENCE
{
  agentStateList  AgentStateList,
  extensions      CSTACommonArguments  OPTIONAL}

AgentStateList ::= SEQUENCE SIZE (1..32) OF AgentStateEntry

AgentStateEntry ::= SEQUENCE
{
  agentID      AgentID          OPTIONAL,
  loggedOnState  BOOLEAN,
  agentInfo    SEQUENCE OF AgentInfo  OPTIONAL}

AgentInfo ::= SEQUENCE
{
  acdGroup      DeviceID          OPTIONAL,
  agentState    AgentState,
  pendingAgentState  [0] IMPLICIT PendingAgentState  OPTIONAL,
  agentStateCondition  [1] IMPLICIT AgentStateCondition  OPTIONAL}

AgentStateCondition ::= ENUMERATED
{
  forcedPause (0),
  other       (1)}

END -- of CSTA-get-agent-state

```

## 20.1.6 Get auto answer

```

CSTA-get-auto-answer
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-auto-answer( 305) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

getAutoAnswer OPERATION ::=
{
  ARGUMENT    GetAutoAnswerArgument
  RESULT      GetAutoAnswerResult
  ERRORS      {universalFailure}
  CODE        local: 305
}

GetAutoAnswerArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetAutoAnswerResult ::= SEQUENCE
{
  autoAnswerOn      BOOLEAN,
  numberOfRings     INTEGER,
  extensions         CSTACommonArguments OPTIONAL}

END -- of CSTA-get-auto-answer

```

20.1.7 Get auto work mode

```

CSTA-get-auto-work-mode
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-auto-work-mode( 306) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

getAutoWorkMode OPERATION ::=
{
  ARGUMENT    GetAutoWorkModeArgument
  RESULT      GetAutoWorkModeResult
  ERRORS      {universalFailure}
  CODE        local: 306
}

GetAutoWorkModeArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetAutoWorkModeResult ::= SEQUENCE
{
  autoWorkOn      BOOLEAN,
  autoWorkInterval INTEGER,
  extensions      CSTACommonArguments OPTIONAL}

END -- of CSTA-get-auto-work-mode

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20.1.8 Get caller id status

```

CSTA-get-caller-id-status
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-caller-id-status( 307) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

getCallerIDStatus OPERATION ::=
{
  ARGUMENT    GetCallerIDStatusArgument
  RESULT      GetCallerIDStatusResult
  ERRORS      {universalFailure}
  CODE        local: 307
}

GetCallerIDStatusArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetCallerIDStatusResult ::= SEQUENCE
{
  callerIDProvided  BOOLEAN,
  extensions        CSTACommonArguments OPTIONAL}

END -- of CSTA-get-caller-id-status

```

20.1.9 Get do not disturb

```

CSTA-get-do-not-disturb
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-do-not-disturb( 308) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CallOrigination FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

getDoNotDisturb OPERATION ::=
{
  ARGUMENT    GetDoNotDisturbArgument
  RESULT      GetDoNotDisturbResult
  ERRORS      {universalFailure}
  CODE        local: 308
}

GetDoNotDisturbArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetDoNotDisturbResult ::= SEQUENCE
{
  doNotDisturbOn      BOOLEAN,
  callOrigination     CallOrigination      OPTIONAL,
  callingDeviceList   SEQUENCE OF DeviceID OPTIONAL,
  extensions          CSTACommonArguments  OPTIONAL}

END -- of CSTA-get-do-not-disturb

```

STANDARDSISO.COM Click to view the full PDF of ISO/IEC 18052:2000

## 20.1.10 Get forwarding

```

CSTA-get-forwarding
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-forwarding( 309) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ForwardList FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

getForwarding OPERATION ::=
{
  ARGUMENT    GetForwardingArgument
  RESULT      GetForwardingResult
  ERRORS      {universalFailure}
  CODE        local: 309
}

GetForwardingArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetForwardingResult ::= SEQUENCE
{
  forwardingList ForwardList,
  extensions     CSTACommonArguments OPTIONAL}

END -- of CSTA-get-forwarding

```

**20.1.11 Get last number dialed**

```

CSTA-get-last-number-dialed
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-last-number-dialed( 310) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

getLastNumberDialed OPERATION ::=
{
  ARGUMENT    GetLastNumberDialedArgument
  RESULT      GetLastNumberDialedResult
  ERRORS      {universalFailure}
  CODE        local: 310
}

GetLastNumberDialedArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetLastNumberDialedResult ::= SEQUENCE
{
  numberDialed DeviceID,
  extensions   CSTACommonArguments OPTIONAL}

END -- of CSTA-get-last-number-dialed

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20.1.12 Get routing mode

```

CSTA-get-routing-mode
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) get-routing-mode( 311) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

getRoutingMode OPERATION ::=
{
  ARGUMENT    GetRoutingModeArgument
  RESULT      GetRoutingModeResult
  ERRORS      {universalFailure}
  CODE        local: 311
}

GetRoutingModeArgument ::= SEQUENCE
{
  device      DeviceID,
  extensions  CSTACommonArguments OPTIONAL}

GetRoutingModeResult ::= SEQUENCE
{
  routingMode    BOOLEAN,
  extensions     CSTACommonArguments    OPTIONAL}

END -- of CSTA-get-routing-mode

```

## 20.1.13 Set agent state

```

CSTA-set-agent-state
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-agent-state( 312) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID, AgentPassword, PendingAgentState FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

setAgentState OPERATION ::=
{
  ARGUMENT      SetAgentStateArgument
  RESULT        SetAgentStateResult
  ERRORS        {universalFailure}
  CODE          local: 312
}

SetAgentStateArgument ::= SEQUENCE
{
  device          DeviceID,
  requestedAgentState ReqAgentState,
  agentID         [2] IMPLICIT AgentID           OPTIONAL,
  password        [3] IMPLICIT AgentPassword     OPTIONAL,
  group           DeviceID                       OPTIONAL,
  extensions      CSTACommonArguments           OPTIONAL}

SetAgentStateResult ::= SEQUENCE
{
  pendingAgentState PendingAgentState           OPTIONAL,
  extensions        CSTACommonArguments         OPTIONAL}

ReqAgentState ::= ENUMERATED
{
  loggedOn          (0),
  loggedOff         (1),
  notReady          (2),
  ready             (3),
  workingAfterCall (4)}

END -- of CSTA-set-agent-state

```

## 20.1.14 Set auto answer

```

CSTA-set-auto-answer
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-auto-answer( 313) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

setAutoAnswer OPERATION ::=
{
  ARGUMENT    SetAutoAnswerArgument
  RESULT      SetAutoAnswerResult
  ERRORS      {universalFailure}
  CODE        local: 313
}

SetAutoAnswerArgument ::= SEQUENCE
{
  device          DeviceID,
  autoAnswerOn   BOOLEAN,
  numberOfRings  INTEGER          OPTIONAL,
  extensions     CSTACommonArguments OPTIONAL}

SetAutoAnswerResult ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData     NULL}

END -- of CSTA-set-auto-answer

```

20.1.15 Set auto work mode

```

CSTA-set-auto-work-mode
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-auto-work-mode( 314) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

setAutoWorkMode OPERATION ::=
{
  ARGUMENT    SetAutoWorkModeArgument
  RESULT      SetAutoWorkModeResult
  ERRORS      {universalFailure}
  CODE        local: 314
}

SetAutoWorkModeArgument ::= SEQUENCE
{
  device          DeviceID,
  autoWorkOn      BOOLEAN,
  autoWorkInterval INTEGER (0..6000)    OPTIONAL,
  extensions      CSTACommonArguments  OPTIONAL}

SetAutoWorkModeResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-set-auto-work-mode

```

STANDARDSISO.COM Click to view the full PDF of ISO/IEC 18052:2000

## 20.1.16 Set caller id status

```

CSTA-set-caller-id-status
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-caller-id-status( 315) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

setCallerIDStatus OPERATION ::=
{
  ARGUMENT    SetCallerIDStatusArgument
  RESULT      SetCallerIDStatusResult
  ERRORS      {universalFailure}
  CODE        local: 315
}

SetCallerIDStatusArgument ::= SEQUENCE
{
  device          DeviceID,
  callerIDProvided  BOOLEAN,
  extensions      CSTACommonArguments OPTIONAL}

SetCallerIDStatusResult ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData     NULL}

END -- of CSTA-set-caller-id-status

```

20.1.17 Set do not disturb

```

CSTA-set-do-not-disturb
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-do-not-disturb( 316) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CallOrigination FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

setDoNotDisturb OPERATION ::=
{
  ARGUMENT      SetDoNotDisturbArgument
  RESULT        SetDoNotDisturbResult
  ERRORS        {universalFailure}
  CODE          local: 316
}

SetDoNotDisturbArgument ::= SEQUENCE
{
  device          DeviceID,
  doNotDisturbOn  BOOLEAN,
  callOrigination CallOrigination      OPTIONAL,
  callingDeviceList SEQUENCE OF DeviceID OPTIONAL,
  extensions      CSTACommonArguments  OPTIONAL}

SetDoNotDisturbResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-set-do-not-disturb

```

STANDARDSISO.COM Client to view the full PDF of ISO/IEC 18052:2000

## 20.1.18 Set forwarding

```

CSTA-set-forwarding
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-forwarding( 317) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ForwardingType FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

setForwarding OPERATION ::=
{
  ARGUMENT      SetForwardingArgument
  RESULT        SetForwardingResult
  ERRORS        {universalFailure}
  CODE          local: 317
}

SetForwardingArgument ::= SEQUENCE
{
  device          DeviceID
  forwardingType  ForwardingType      OPTIONAL,
  activateForward BOOLEAN,
  forwardDDN     DeviceID              OPTIONAL,
  ringCount      INTEGER (1..100)     OPTIONAL,
  extensions     CSTACommonArguments  OPTIONAL}

SetForwardingResult ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData     NULL}

END -- of CSTA-set-forwarding

```

20.1.19 Set routing mode

```

CSTA-set-routeing-mode
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) set-routeing-mode( 318) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

setRouteingMode OPERATION ::=
{
  ARGUMENT    SetRouteingModeArgument
  RESULT      SetRouteingModeResult
  ERRORS      {universalFailure}
  CODE        local: 318
}

SetRouteingModeArgument ::= SEQUENCE
{
  device          DeviceID,
  routeingMode    BOOLEAN,
  extensions      CSTACommonArguments OPTIONAL}

SetRouteingModeResult ::= CHOICE
{
  extensions      CSTACommonArguments,
  noData          NULL}

END -- of CSTA-set-routeing-mode

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20.2 Events

### 20.2.1 Agent busy

```

CSTA-agent-busy-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) agent-busy-event( 319) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID, PendingAgentState FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AgentBusyEvent ::= SEQUENCE
{
  agentDevice          SubjectDeviceID,
  agentID              AgentID              OPTIONAL,
  acdGroup             DeviceID             OPTIONAL,
  pendingAgentState    [2] IMPLICIT PendingAgentState  OPTIONAL,
  cause                [3] IMPLICIT EventCause         OPTIONAL,
  extensions           CSTACommonArguments  OPTIONAL}

END -- of CSTA-agent-busy-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

20.2.2 Agent logged off

```

CSTA-agent-logged-off-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) agent-logged-off-event( 320) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID, AgentPassword FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AgentLoggedOffEvent ::= SEQUENCE
{
  agentDevice          SubjectDeviceID,
  agentID              [2] IMPLICIT AgentID          OPTIONAL,
  acdGroup             DeviceID                      OPTIONAL,
  agentPassword        [3] IMPLICIT AgentPassword    OPTIONAL,
  cause                EventCause                   OPTIONAL,
  extensions           CSTACommonArguments          OPTIONAL}

END -- of CSTA-agent-logged-off-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20.2.3 Agent logged on

```

CSTA-agent-logged-on-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) agent-logged-on-event( 321) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID, AgentPassword FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AgentLoggedOnEvent ::= SEQUENCE
{
  agentDevice          SubjectDeviceID,
  agentID              [2] IMPLICIT AgentID          OPTIONAL,
  acdGroup             DeviceID                      OPTIONAL,
  agentPassword        [3] IMPLICIT AgentPassword    OPTIONAL,
  cause                EventCause                   OPTIONAL,
  extensions           CSTACommonArguments           OPTIONAL}

END -- of CSTA-agent-logged-on-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

20.2.4 Agent not ready

```

CSTA-agent-not-ready-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) agent-not-ready-event( 322) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AgentNotReadyEvent ::= SEQUENCE
{
  agentDevice          SubjectDeviceID,
  agentID              AgentID          OPTIONAL,
  acdGroup             DeviceID         OPTIONAL,
  cause                EventCause      OPTIONAL,
  extensions           CSTACommonArguments OPTIONAL}
END -- of CSTA-agent-not-ready-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20.2.5 Agent ready

```

CSTA-agent-ready-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) agent-ready-event( 323) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID, AgentPassword FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AgentReadyEvent ::= SEQUENCE
{
  agentDevice          SubjectDeviceID,
  agentID              AgentID          OPTIONAL,
  acdGroup             DeviceID         OPTIONAL,
  cause               EventCause       OPTIONAL,
  extensions           CSTACommonArguments OPTIONAL}

END -- of CSTA-agent-ready-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

20.2.6 Agent working after call

```

CSTA-agent-working-after-call-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) agent-working-after-call-event( 324) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
AgentID, PendingAgentState FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AgentWorkingAfterCallEvent ::= SEQUENCE
{
  agentDevice          SubjectDeviceID,
  agentID              AgentID              OPTIONAL,
  acdGroup             DeviceID             OPTIONAL,
  pendingAgentState   [2] IMPLICIT ENUMERATED
    {notReady         (0),
     ready             (1),
     null              (2)}                OPTIONAL,
  cause                [3] IMPLICIT EventCause  OPTIONAL,
  extensions           CSTACommonArguments    OPTIONAL}

END -- of CSTA-agent-working-after-call-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 20.2.7 Auto answer

```

CSTA-auto-answer-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) auto-answer-event( 40) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AutoAnswerEvent ::= SEQUENCE
{
  invokingDevice          SubjectDeviceID,
  autoAnswerOn            BOOLEAN,
  numberOfRings           INTEGER          OPTIONAL,
  extensions              CSTACommonArguments  OPTIONAL}

END -- of CSTA-auto-answer-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

20.2.8 Auto work mode

```
CSTA-auto-work-mode-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) auto-work-mode-event( 326) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

AutoWorkModeEvent ::= SEQUENCE
{
  invokingDevice      SubjectDeviceID,
  autoWorkOn          BOOLEAN,
  autoWorkInterval    INTEGER,
  extensions          CSTACommonArguments OPTIONAL}

END -- of CSTA-auto-work-mode-event
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**20.2.9 Call back**

```

CSTA-call-back-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) call-back-event( 327) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

CallBackEvent ::= SEQUENCE
{
  originatingDevice      SubjectDeviceID,
  targetDevice           SubjectDeviceID,
  callBackSetCanceled   BOOLEAN,
  extensions             CSTACommonArguments OPTIONAL;
}

END -- of CSTA-call-back-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

20.2.10 Call back message

```
CSTA-call-back-message-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) call-back-message-event( 328) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

CallBackMessageEvent ::= SEQUENCE
{
  originatingDevice      SubjectDeviceID,
  targetDevice           SubjectDeviceID,
  callBackMsgSetCanceled BOOLEAN,
  extensions              CSTACommonArguments OPTIONAL}

END -- of CSTA-call-back-message-event
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**20.2.11 Caller id status**

```

CSTA-caller-id-status-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) caller-id-status-event( 329) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

CallerIDStatusEvent ::= SEQUENCE
{
  device                DeviceID,
  callerIDProvided      BOOLEAN,
  extensions             CSTACommonArguments OPTIONAL}

END -- of CSTA-caller-id-status-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

20.2.12 Do not disturb

```

CSTA-do-not-disturb-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) do-not-disturb-event( 42) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CallOrigination FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

DoNotDisturbEvent ::= SEQUENCE
{
  device                SubjectDeviceID,
  doNotDisturbOn        BOOLEAN,
  callOrigination        CallOrigination      OPTIONAL,
  callingDeviceList      SEQUENCE OF DeviceID OPTIONAL,
  extensions             CSTACommonArguments  OPTIONAL}

END -- of CSTA-do-not-disturb-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**20.2.13 Forwarding**

```

CSTA-forwarding-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) forwarding-event( 43) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
DeviceID, SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ForwardingType, ForwardDefault FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

ForwardingEvent ::= SEQUENCE
{
  device                SubjectDeviceID,
  forwardingType        ForwardingType        OPTIONAL,
  forwardStatus         BOOLEAN,
  forwardTo             DeviceID              OPTIONAL,
  forwardDefault        ForwardDefault        OPTIONAL,
  ringCount             INTEGER (1..100)     OPTIONAL,
  extensions            CSTACommonArguments  OPTIONAL}

END -- of CSTA-forwarding-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

20.2.14 Routing mode

```
CSTA-routeing-mode-event
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) routeing-mode-event( 332) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

RouteingModeEvent ::= SEQUENCE
{
  device                SubjectDeviceID,
  routeingMode          BOOLEAN,
  extensions             CSTACommonArguments OPTIONAL}

END -- of CSTA-routeing-mode-event
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 21 Device maintenance events

### 21.1 Events

#### 21.1.1 Back in service

```

CSTA-back-in-service-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) back-in-service-event( 333) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

BackInServiceEvent ::= SEQUENCE
{
    device                SubjectDeviceID,
    cause                 EventCause          OPTIONAL,
    extensions            CSTACommonArguments OPTIONAL}

END -- of CSTA-back-in-service-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

21.1.2 Device capabilities changed

```

CSTA-device-capabilities-changed-event
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-capabilities-changed-event( 334) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

DeviceCapsChangedEvent ::= SEQUENCE
{
  device          SubjectDeviceID,
  cause          EventCause          OPTIONAL,
  extensions     CSTACommonArguments OPTIONAL}

END -- of CSTA-device-capabilities-changed-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 21.1.3 Out of service

```

CSTA-out-of-service-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) out-of-service-event( 335) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
SubjectDeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
;

OutOfServiceEvent ::= SEQUENCE
{
    device                SubjectDeviceID,
    cause                  EventCause
                           OPTIONAL,
    extensions             CSTACommonArguments
                           OPTIONAL}

END -- of CSTA-out-of-service-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

**22 I/O services**

**22.1 Registration services**

**22.1.1 I/O register**

```

CSTA-io-register
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) io-register( 340) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-identifiers( 123) }
IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

ioRegister      OPERATION ::=
{
    ARGUMENT      IORegisterArgument
    RESULT        IORegisterResult
    ERRORS        {universalFailure}
    CODE          local: 340
}

IORegisterArgument ::= SEQUENCE
{
    ioDevice      DeviceID          OPTIONAL,
    extensions    CSTACommonArguments OPTIONAL}

IORegisterResult ::= SEQUENCE
{
    ioRegisterReqID IORegisterReqID,
    extensions      CSTACommonArguments OPTIONAL}

END -- of CSTA-io-register

```

STANDARDSISO.COM · Click to view the full PDF of ISO/IEC 18052:2000

## 22.1.2 I/O register abort

```

CSTA-io-register-abort
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) io-register-abort( 341) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
IORegisterReqID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

ioRegisterAbort OPERATION ::=
{
  ARGUMENT          IORegisterAbortArgument
  ERRORS             {universalFailure}
  ALWAYS RESPONDS   FALSE
  CODE               local: 341
}

IORegisterAbortArgument ::= SEQUENCE
{
  ioRegisterReqID   IORegisterReqID,
  extensions         CSTACommonArguments OPTIONAL}

END -- of CSTA-io-register-abort

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

22.1.3 I/O register cancel

```

CSTA-io-register-cancel
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) io-register-cancel( 342) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

ioRegisterCancelOPERATION ::=
{
    ARGUMENT    IORegisterCancelArgument
    RESULT      IORegisterCancelResult
    ERRORS      {universalFailure}
    CODE        local: 342
}

IORegisterCancelArgument ::= SEQUENCE
{
    ioRegisterReqID      IORegisterReqID,
    extensions           CSTACommonArguments OPTIONAL}

IORegisterCancelResult ::= CHOICE
{
    extensions CSTACommonArguments,
    noData     NULL}

END -- of CSTA-io-register-cancel

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 22.2 Services

### 22.2.1 Data path resumed

```

CSTA-data-path-resumed
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) data-path-resumed( 118) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
IOCrossRefID, IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

dataPathResumedOPERATION ::=
{
    ARGUMENT    DataPathResumedArgument
    RESULT      DataPathResumedResult
    ERRORS      {universalFailure}
    CODE        local: 118
}

DataPathResumedArgument ::= SEQUENCE
{
    ioCrossRefID      IOCrossRefID,
    ioRegisterReqID   IORegisterReqID      OPTIONAL,
    extensions        CSTACommonArguments  OPTIONAL}

DataPathResumedResult ::= CHOICE
{
    extensions  CSTACommonArguments,
    noData     NULL }

END -- of CSTA-data-path-resumed

```

22.2.2 Data path suspended

```

CSTA-data-path-suspended
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) data-path-suspended( 116) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
IOCrossRefID, IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

dataPathSuspendedOPERATION ::=
{
    ARGUMENT    DataPathSuspendedArgument
    RESULT      DataPathSuspendedResult
    ERRORS      {universalFailure}
    CODE        local: 116
}

DataPathSuspendedArgument ::= SEQUENCE
{
    ioCrossRefID      IOCrossRefID,
    ioReqRegisterID   IORegisterReqID      OPTIONAL,
    extensions        CSTACommonArguments  OPTIONAL}

DataPathSuspendedResult ::= CHOICE
{
    extensions  CSTACommonArguments,
    noData     NULL }

END -- of CSTA-data-path-suspended

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 22.2.3 Fast data

```

CSTA-fast-data
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) fast-data( 119) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTAObject FROM CSTA-switching-function-objects
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) switching-function-objects( 122) }
DataPathType, IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
DisplayID FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) };

fastData          OPERATION ::=
{
    ARGUMENT      FastDataArgument
    RESULT        FastDataResult
    ERRORS        {universalFailure}
    CODE          local: 119
}

FastDataArgument ::= SEQUENCE
{
    ioRegisterReqID  IORegisterReqID          OPTIONAL,
    object           CSTAObject,
    dataPathType    DataPathType             OPTIONAL,
    displayAttributes DisplayAttribute        OPTIONAL,
    ioData           OCTET STRING (SIZE(0..240)) OPTIONAL,
    extensions       CSTACommonArguments     OPTIONAL}

FastDataResult ::= CHOICE
{
    extensions  CSTACommonArguments,
    noData     NULL}

DisplayAttribute ::= SEQUENCE
{
    displayID           DisplayID          OPTIONAL,
    physicalBaseRowNumber [0] IMPLICIT INTEGER OPTIONAL,
    physicalBaseColumnNumber [1] IMPLICIT INTEGER OPTIONAL,
    offset              [2] IMPLICIT INTEGER OPTIONAL}

END -- of CSTA-fast-data

```

22.2.4 Resume data path

```

CSTA-resume-data-path
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) resume-data-path( 117) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
IOCrossRefID, IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

resumeDataPath      OPERATION ::=
{
    ARGUMENT      ResumeDataPathArgument
    RESULT        ResumeDataPathResult
    ERRORS        {universalFailure}
    CODE          local: 117
}

ResumeDataPathArgument ::= SEQUENCE
{
    ioCrossRefID      IOCrossRefID,
    ioRegisterReqID   IORegisterReqID      OPTIONAL,
    extensions        CSTACommonArguments  OPTIONAL}

ResumeDataPathResult ::= CHOICE
{
    extensions        CSTACommonArguments,
    noData            NULL }

END -- of CSTA-resume-data-path

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 22.2.5 Send broadcast data

```

CSTA-send-broadcast-data
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) send-broadcast-data( 114) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
DataPathType, DisplayAttributeList FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }

sendBroadcastData OPERATION ::=
{
    ARGUMENT    SendBroadcastDataArgument
    RESULT      SendBroadcastDataResult
    ERRORS      {universalFailure}
    CODE        local: 114
}

SendBroadcastDataArgument ::= SEQUENCE
{
    ioData          OCTET STRING (SIZE(0..240)),
    dataPathType    DataPathType                OPTIONAL,
    displayAttributes DisplayAttributeList        OPTIONAL,
    extensions      CSTACommonArguments          OPTIONAL}

SendBroadcastDataResult ::= CHOICE
{
    extensions CSTACommonArguments,
    noData     NULL}

END -- of CSTA-send-broadcast-data

```

## 22.2.6 Send data

```

CSTA-send-data
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) send-data( 112) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
EventCause FROM CSTA-event-causes
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) event-causes( 121) }
IOCrossRefID, IORegisterReqID, DisplayAttributeList FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

sendData      OPERATION ::=
{
  ARGUMENT    SendDataArgument
  RESULT      SendDataResult
  ERRORS      {universalFailure}
  CODE        local: 112
}

SendDataArgument ::= SEQUENCE
{
  ioCrossRefID      IOCrossRefID,
  ioRegisterReqID   [0] IMPLICIT IORegisterReqID      OPTIONAL,
  displayAttributes DisplayAttributeList              OPTIONAL,
  ioData            OCTET STRING (SIZE(0..240)),
  ioCause           EventCause                        OPTIONAL,
  extensions        CSTACommonArguments              OPTIONAL}

SendDataResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-send-data

```

## 22.2.7 Send multicast data

```

CSTA-send-multicast-data
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) send-multicast-data( 113) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
IOCrossRefID, DisplayAttributeList FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }

sendMulticastDataOPERATION ::=
{
    ARGUMENT    SendMulticastDataArgument
    RESULT      SendMulticastDataResult
    ERRORS      {universalFailure}
    CODE        local: 113
}

SendMulticastDataArgument ::= SEQUENCE
{
    ioCrossRefIDList    SEQUENCE OF IOCrossRefID,
    ioData              OCTET STRING (SIZE(0..240))           OPTIONAL,
    displayAttributes   DisplayAttributeList                 OPTIONAL,
    extensions          CSTACommonArguments                   OPTIONAL}

SendMulticastDataResult ::= CHOICE
{
    extensions    CSTACommonArguments,
    noData        NULL}

END -- of CSTA-send-multicast-data

```

## 22.2.8 Start data path

```

CSTA-start-data-path
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) start-data-path( 110) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTAObject FROM CSTA-switching-function-objects
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) switching-function-objects( 122) }
DataPathType, IOCrossRefID, IORegisterReqID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DisplayID FROM CSTA-physical-device-feature
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) physical-device-feature( 137) };

startDataPathOPERATION ::=
{
  ARGUMENT      StartDataPathArgument
  RESULT        StartDataPathResult
  ERRORS        {universalFailure}
  CODE          local: 110
}

StartDataPathArgument ::= SEQUENCE
{
  ioRegisterReqID      IORegisterReqID          OPTIONAL,
  object               CSTAObject,
  dataPathDirection   [0] IMPLICIT DataPathDirection  OPTIONAL,
  dataPathType         [1] IMPLICIT DataPathType        OPTIONAL,
  displayID           DisplayID                    OPTIONAL,
  numberOfCharactersToCollect [2] IMPLICIT INTEGER      OPTIONAL,
  terminationCharacter IA5String (SIZE(1..1))         OPTIONAL,
  timeout              [3] IMPLICIT INTEGER            OPTIONAL,
  extensions           CSTACommonArguments          OPTIONAL}

StartDataPathResult ::=SEQUENCE
{
  ioCrossRefID      IOCrossRefID,
  numberOfCharactersToCollect [0] IMPLICIT INTEGER  OPTIONAL,
  terminationCharacter IA5String(SIZE(1..1))  OPTIONAL,
  timeout           [1] IMPLICIT INTEGER  OPTIONAL,
  extensions       CSTACommonArguments  OPTIONAL}

DataPathDirection ::= ENUMERATED
{
  computeFunctionToObject      (0),
  objectToComputeFunction      (1),
  bidirectional                 (2)}

END -- of CSTA-start-data-path

```

## 22.2.9 Stop data path

```

CSTA-stop-data-path
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) stop-data-path( 111) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
IOCrossRefID, IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }

stopDataPathOPERATION ::=
{
    ARGUMENT      StopDataPathArgument
    RESULT        StopDataPathResult
    ERRORS        {universalFailure}
    CODE          local: 111
}

StopDataPathArgument ::= SEQUENCE
{
    ioCrossRefID      IOCrossRefID,
    ioRegisterReqID  IORegisterReqID      OPTIONAL,
    extensions        CSTACCommonArguments OPTIONAL}

StopDataPathResult ::= CHOICE
{
    extensions CSTACCommonArguments,
    noData     NULL}

END -- of CSTA-stop-data-path

```

22.2.10 Suspend data path

```

CSTA-suspend-data-path
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) suspend-data-path( 115) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
IOCrossRefID, IORegisterReqID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

suspendDataPathOPERATION ::=
{
    ARGUMENT    SuspendDataPathArgument
    RESULT      SuspendDataPathResult
    ERRORS      {universalFailure}
    CODE        local: 115
}

SuspendDataPathArgument ::= SEQUENCE
{
    ioCrossRefID      IOCrossRefID,
    ioRegisterReqID   IORegisterReqID      OPTIONAL,
    extensions        CSTACommonArguments  OPTIONAL}

SuspendDataPathResult ::= CHOICE
{
    extensions        CSTACommonArguments,
    noData            NULL }

END -- of CSTA-suspend-data-path

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 23 Data Collection Services

### 23.1 Services

#### 23.1.1 Data Collected

```

CSTA-data-collected
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) data-collected( 343) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
ConnectionInformation FROM CSTA-media-services
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) media-services( 136) }
DcollCrossRefID FROM CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) };

dataCollected OPERATION ::=
{
  ARGUMENT      DataCollectedArgument
  RESULT        DataCollectedResult
  ERRORS        {universalFailure}
  CODE          local: 343
}

DataCollectedArgument ::= SEQUENCE
{
  dcollCrossRefID      DcollCrossRefID,
  digitsData           [0] IMPLICIT DigitsData           OPTIONAL,
  telTonesData        [1] IMPLICIT TelTonesData          OPTIONAL,
  connectionInformation [2] IMPLICIT ConnectionInformation OPTIONAL,
  dcollCause          DcollCause                        OPTIONAL,
  extensions           CSTACommonArguments              OPTIONAL}

DataCollectedResult ::= CHOICE
{
  extensions  CSTACommonArguments,
  noData     NULL}

DigitsData ::= SEQUENCE
{
  digitsDetected      IA5String (SIZE(0..64)),
  digitsDuration     [0] IMPLICIT SEQUENCE OF INTEGER  OPTIONAL,
  digitsPauseDuration [1] IMPLICIT SEQUENCE OF INTEGER  OPTIONAL}

TelTonesData ::= SEQUENCE
{
  toneDetected      ToneDetected,
  toneFrequency     [0] IMPLICIT INTEGER              OPTIONAL,
  toneDuration      [1] IMPLICIT INTEGER              OPTIONAL,
  tonePauseDuration [2] IMPLICIT INTEGER              OPTIONAL}

ToneDetected ::= ENUMERATED
{
  beep              ( 0),
  billing           ( 1),
  busy              ( 2),
  carrier           ( 3),
  confirmation     ( 4),
  dial              ( 5),
  faxCNG           ( 6),
  hold              ( 7),
  howler           ( 8),
}

```

```
intrusion          ( 9),
modemCNG           (10),
park               (11),
recordWarning      (12),
reorder            (13),
ringback           (14),
silence            (15),
sitVC              (16),
sitIC              (17),
sitRO              (18),
sitNC              (19),
other              (20)}

DcollCause ::= ENUMERATED
{
    flushCharReceived    (0),
    charCountReached     (1),
    timeout               (2),
    sfTerminated         (3)}

END -- of CSTA-data-collected
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 23.1.2 Data Collection Resumed

```

CSTA-data-collection-resumed
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) data-collection-resumed( 344) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DcollCrossRefID FROM CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) };

dataCollectionResumed OPERATION ::=
{
  ARGUMENT    DataCollectionResumedArgument
  RESULT      DataCollectionResumedResult
  ERRORS      {universalFailure}
  CODE        local: 344
}

DataCollectionResumedArgument ::= SEQUENCE
{
  dcollCrossRefID    DcollCrossRefID,
  extensions          CSTACommonArguments    OPTIONAL}

DataCollectionResumedResult ::= CHOICE
{
  extensions    CSTACommonArguments,
  noData        NULL}

END -- of CSTA-data-collection-resumed

```

23.1.3 Data Collection Suspended

```

CSTA-data-collection-suspended
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) data-collection-suspended( 345) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DcollCrossRefID FROM CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) };

dataCollectionSuspended OPERATION ::=
{
  ARGUMENT   DataCollectionSuspendedArgument
  RESULT     DataCollectionSuspendedResult
  ERRORS     {universalFailure}
  CODE       local: 345
}

DataCollectionSuspendedArgument ::= SEQUENCE
{
  dcollCrossRefID      DcollCrossRefID,
  extensions           CSTACommonArguments OPTIONAL}

DataCollectionSuspendedResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-data-collection-suspended

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 23.1.4 Resume Data Collection

```

CSTA-resume-data-collection
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) resume-data-collection( 346) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DcollCrossRefID FROM CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) };

resumeDataCollection OPERATION ::=
{
  ARGUMENT    ResumeDataCollectionArgument
  RESULT      ResumeDataCollectionResult
  ERRORS      {universalFailure}
  CODE        local: 346
}

ResumeDataCollectionArgument ::= SEQUENCE
{
  dcollCrossRefID    DcollCrossRefID,
  extensions          CSTACommonArguments    OPTIONAL}

ResumeDataCollectionResult ::= CHOICE
{
  extensions    CSTACommonArguments,
  noData        NULL}

END -- of CSTA-resume-data-collection

```

## 23.1.5 Start Data Collection

```

CSTA-start-data-collection
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) start-data-collection( 347) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
DeviceID FROM CSTA-device-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-identifiers( 123) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DcollCrossRefID FROM CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) };

startDataCollection OPERATION ::=
{
  ARGUMENT    StartDataCollectionArgument
  RESULT      StartDataCollectionResult
  ERRORS      {universalFailure}
  CODE        local: 347
}

StartDataCollectionArgument ::= SEQUENCE
{
  object          CallObject,
  dataCollType    DataCollType          OPTIONAL,
  digitsReportingCriteria DigitsReportingCriteria OPTIONAL,
  extensions      CSTACommonArguments   OPTIONAL}

StartDataCollectionResult ::= SEQUENCE
{
  dcollCrossRefID DcollCrossRefID,
  extensions      CSTACommonArguments   OPTIONAL}

CallObject ::= CHOICE
{
  device DeviceID,
  call   ConnectionID}

DataCollType ::= ENUMERATED
{
  digits    (0),
  telTones (1)}

DigitsReportingCriteria ::= SEQUENCE
{
  numChars  [0] IMPLICIT INTEGER   OPTIONAL,
  flushChar IA5String (SIZE(1..1)) OPTIONAL,
  timeout   [1] IMPLICIT INTEGER   OPTIONAL}

END -- of CSTA-start-data-collection

```

## 23.1.6 Stop Data Collection

```

CSTA-stop-data-collection
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) stop-data-collection( 348) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DcollCrossRefID FROM CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) };

stopDataCollection OPERATION ::=
{
  ARGUMENT      StopDataCollectionArgument
  RESULT        StopDataCollectionResult
  ERRORS        {universalFailure}
  CODE          local: 348
}

StopDataCollectionArgument ::= SEQUENCE
{
  dcollCrossRefID      DcollCrossRefID,
  extensions            CSTACommonArguments      OPTIONAL}

StopDataCollectionResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-stop-data-collection

```

## 23.1.7 Suspend Data Collection

```

CSTA-suspend-data-collection
{ iso( 1) identified-organization( 3) icd-ecma( 12)
  standard( 0) csta3( 285) suspend-data-collection( 349) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACCommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
DcollCrossRefID FROM CSTA-data-collection
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) data-collection( 138) };

suspendDataCollection OPERATION ::=
{
  ARGUMENT    SuspendDataCollectionArgument
  RESULT      SuspendDataCollectionResult
  ERRORS      {universalFailure}
  CODE        local: 349
}

SuspendDataCollectionArgument ::= SEQUENCE
{
  dcollCrossRefID    DcollCrossRefID,
  extensions          CSTACCommonArguments    OPTIONAL}

SuspendDataCollectionResult ::= CHOICE
{
  extensions    CSTACCommonArguments,
  noData        NULL}

END -- of CSTA-suspend-data-collection

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24 Voice unit services and events

### 24.1 Services

#### 24.1.1 Concatenate message

```

CSTA-concatenate-message
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) concatenate-message( 500) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

concatenateMessage OPERATION ::=
{
    ARGUMENT    ConcatenateMessageArgument
    RESULT      ConcatenateMessageResult
    ERRORS      {universalFailure}
    CODE        local: 500
}

ConcatenateMessageArgument ::= SEQUENCE
{
    messagesToConcatenate SEQUENCE OF MessageID,
    extensions             CSTACommonArguments             OPTIONAL}

ConcatenateMessageResult ::= SEQUENCE
{
    concatenatedMessage MessageID,
    extensions           CSTACommonArguments             OPTIONAL}

END -- of CSTA-concatenate-message

```

24.1.2 Delete message

```

CSTA-delete-message
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) delete-message( 501) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

deleteMessageOPERATION ::=
{
    ARGUMENT    DeleteMessageArgument
    RESULT      DeleteMessageResult
    ERRORS      {universalFailure}
    CODE        local: 501
}

DeleteMessageArgument ::= SEQUENCE
{
    messageToBeDeleted    MessageID,
    extensions             CSTACommonArguments OPTIONAL}

DeleteMessageResult ::=CHOICE
{
    extensions    CSTACommonArguments,
    noData        NULL}

END -- of CSTA-delete-message

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24.1.3 Play message

```

CSTA-play-message
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) play-message( 502) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID, TerminatingConditions FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

playMessage      OPERATION ::=
{
  ARGUMENT      PlayMessageArgument
  RESULT        PlayMessageResult
  ERRORS        {universalFailure}
  CODE          local: 502
}

PlayMessageArgument ::= SEQUENCE
{
  messageToBePlayed      MessageID,
  overConnection         ConnectionID,
  duration                INTEGER OPTIONAL,
  termination             TerminatingConditions OPTIONAL,
  extensions              CSTACommonArguments OPTIONAL}

PlayMessageResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-play-message

```

## 24.1.4 Query voice attribute

```

CSTA-query-voice-attribute
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) query-voice-attribute( 503) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
AttributeInfo, MessageID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

queryVoiceAttributeOPERATION ::=
{
  ARGUMENT    QueryVoiceAttributeArgument
  RESULT      QueryVoiceAttributeResult
  ERRORS      {universalFailure}
  CODE        local: 503
}

QueryVoiceAttributeArgument ::= SEQUENCE
{
  messageToQuery      MessageID,
  attributeToQuery    AttributeToQuery,
  connection          ConnectionID          OPTIONAL,
  extensions          CSTACommonArguments   OPTIONAL}

QueryVoiceAttributeResult ::= SEQUENCE
{
  attribute      AttributeInfo,
  extensions    CSTACommonArguments   OPTIONAL}

AttributeToQuery ::= ENUMERATED
{
  encodingAlgorithm    (0),
  samplingRate         (1),
  duration              (2),
  fileName             (3),
  currentPosition      (4),
  currentSpeed         (5),
  currentVolume        (6),
  currentLevel         (7),
  currentState        (8)
}

END -- of CSTA-query-voice-attribute

```

## 24.1.5 Record message

```

CSTA-record-message
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) record-message( 511) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID, EncodingAlgorithm, TerminatingConditions
FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

recordMessageOPERATION ::=
{
    ARGUMENT    RecordMessageArgument
    RESULT      RecordMessageResult
    ERRORS      {universalFailure}
    CODE        local: 511
}

RecordMessageArgument ::= SEQUENCE
{
    callToBeRecorded      ConnectionID,
    samplingRate           [0] IMPLICIT INTEGER    OPTIONAL,
    encodingAlgorithm      EncodingAlgorithm        OPTIONAL,
    maxDuration            [1] IMPLICIT INTEGER    OPTIONAL,
    termination            TerminatingConditions  OPTIONAL,
    extensions             CSTACommonArguments     OPTIONAL}

RecordMessageResult ::= SEQUENCE
{
    resultingMessage      MessageID,
    extensions            CSTACommonArguments     OPTIONAL}

END -- of CSTA-record-message

```

## 24.1.6 Reposition

```

CSTA-reposition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) reposition( 504) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

reposition      OPERATION ::=
{
  ARGUMENT      RepositionArgument
  RESULT        RepositionResult
  ERRORS        {universalFailure}
  CODE          local: 504
}

RepositionArgument ::= SEQUENCE
{
  connection          ConnectionID,
  periodOfReposition  Period,
  messageToReposition MessageID          OPTIONAL,
  extensions          CSTACommonArguments OPTIONAL}

RepositionResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

Period ::= CHOICE
{
  absolutePosition  ENUMERATED
    {
      startOfMessage(0),
      endOfMessage(1)},
  relativePosition  INTEGER}

END -- of CSTA-reposition

```

## 24.1.7 Resume

```

CSTA-resume
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) resume( 505) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

resume          OPERATION ::=
{
  ARGUMENT      ResumeArgument
  RESULT        ResumeResult
  ERRORS        {universalFailure}
  CODE          local: 505
}

ResumeArgument ::= SEQUENCE
{
  connection          ConnectionID,
  messageToResume     MessageID,
  duration            INTEGER,
  extensions           CSTACommonArguments OPTIONAL,
                    OPTIONAL,
                    OPTIONAL}

ResumeResult ::= CHOICE
{
  extensions CSTACommonArguments,
  noData     NULL}

END -- of CSTA-resume

```

24.1.8 Review

```

CSTA-review
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) review( 506) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }

review          OPERATION ::=
{
    ARGUMENT      ReviewArgument
    RESULT        ReviewResult
    ERRORS        {universalFailure}
    CODE          local: 506
}

ReviewArgument ::= SEQUENCE
{
    connection          ConnectionID,
    periodToReview      PeriodToReview,
    messageToReview     MessageID          OPTIONAL,
    extensions          CSTACommonArguments OPTIONAL}

ReviewResult ::= CHOICE
{
    extensions  CSTACommonArguments,
    noData     NULL}

PeriodToReview ::= CHOICE
{
    startOfMessage  NULL,
    lengthOfReview  INTEGER}

END -- of CSTA-review

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24.1.9 Set voice attribute

```

CSTA-set-voice-attribute
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) set-voice-attribute( 507) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
Volume FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) };

setVoiceAttributeOPERATION ::=
{
    ARGUMENT      SetVoiceAttributeArgument
    RESULT        SetVoiceAttributeResult
    ERRORS        { universalFailure }
    CODE          local: 507
}

SetVoiceAttributeArgument ::= SEQUENCE
{
    connection          ConnectionID,
    attributeToSet      AttributeToSet,
    message             MessageID          OPTIONAL,
    extensions          CSTACommonArguments OPTIONAL}

SetVoiceAttributeResult ::= CHOICE
{
    extensions CSTACommonArguments,
    noData     NULL}

AttributeToSet ::= CHOICE
{
    currentSpeed      [0] IMPLICIT INTEGER,
    currentVolume     [1] Volume,
    currentGain       [2] IMPLICIT INTEGER (0 .. 100)}

END -- of CSTA-set-voice-attribute

```

24.1.10 Stop

```

CSTA-stop
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) stop( 508) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }

stop
  OPERATION ::=
  {
    ARGUMENT      StopArgument
    RESULT        StopResult
    ERRORS        {universalFailure}
    CODE          local: 508
  }

StopArgument ::= SEQUENCE
  {
    connection      ConnectionID,
    messageToBeStopped MessageID OPTIONAL,
    extensions      CSTACommonArguments OPTIONAL}

StopResult ::= CHOICE
  {
    extensions      CSTACommonArguments,
    noData          NULL}

END -- of CSTA-stop

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24.1.11 Suspend

```

CSTA-suspend
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) suspend( 509) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
ConnectionID FROM CSTA-call-connection-identifiers
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) device-feature-types( 127) }
CSTACCommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) };

suspend OPERATION ::=
{
  ARGUMENT    SuspendArgument
  RESULT      SuspendResult
  ERRORS      {universalFailure}
  CODE        local: 509
}

SuspendArgument ::= SEQUENCE
{
  connection  ConnectionID,
  message     MessageID          OPTIONAL,
  extensions  CSTACCommonArguments  OPTIONAL}

SuspendResult ::= CHOICE
{
  extensions  CSTACCommonArguments,
  noData     NULL}

END -- of CSTA-suspend

```

24.1.12 Synthesize message

```

CSTA-synthesize-message
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) synthesize-message( 510) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
ControlData, MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

synthesizeMessage OPERATION ::=
{
    ARGUMENT SynthesizeMessageArgument
    RESULT SynthesizeMessageResult
    ERRORS {universalFailure}
    CODE local: 510
}

SynthesizeMessageArgument ::= SEQUENCE
{
    textToBeSynthesized IA5String,
    control ControlData OPTIONAL,
    extensions CSTACommonArguments OPTIONAL}

SynthesizeMessageResult ::= SEQUENCE
{
    synthesizedMessage MessageID,
    extensions CSTACommonArguments OPTIONAL}

END -- of CSTA-synthesize-message

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24.2 Events

### 24.2.1 Play

```

CSTA-play-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) play( 75) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) };

PlayEvent ::=SEQUENCE
{
    connection          ConnectionID,
    message             MessageID,
    length              [0] IMPLICIT INTEGER           OPTIONAL,
    currentPosition    [1] IMPLICIT INTEGER           OPTIONAL,
    speed               [2] IMPLICIT INTEGER           OPTIONAL,
    cause              EventCause                     OPTIONAL,
    servicesPermitted  ServicesPermitted              OPTIONAL,
    extensions         CSTACommonArguments            OPTIONAL}

END -- of CSTA-play-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

24.2.2 Record

```

CSTA-record-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) record( 76) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) };

RecordEvent ::= SEQUENCE
{
    connection          ConnectionID,
    message             MessageID,
    length              [0] IMPLICIT INTEGER OPTIONAL,
    currentPosition    [1] IMPLICIT INTEGER OPTIONAL,
    cause              EventCause OPTIONAL,
    servicesPermitted  ServicesPermitted OPTIONAL,
    extensions         CSTACommonArguments OPTIONAL}

END -- of CSTA-record-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24.2.3 Review

```

CSTA-review-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) review( 77) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) };

ReviewEvent ::= SEQUENCE
{
    connection          ConnectionID,
    message             MessageID,
    length              [0] IMPLICIT INTEGER    OPTIONAL,
    currentPosition    [1] IMPLICIT INTEGER    OPTIONAL,
    cause               EventCause             OPTIONAL,
    servicesPermitted  ServicesPermitted       OPTIONAL,
    extensions          CSTACCommonArguments   OPTIONAL}

END -- of CSTA-review-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

24.2.4 Stop

```

CSTA-stop-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) stop( 78) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) };

StopEvent ::= SEQUENCE
{
    connection          ConnectionID,
    message             MessageID,
    length              [0] IMPLICIT INTEGER OPTIONAL,
    currentPosition    [1] IMPLICIT INTEGER OPTIONAL,
    cause               EventCause OPTIONAL,
    servicesPermitted  ServicesPermitted OPTIONAL,
    extensions          CSTACommonArguments OPTIONAL}

END -- of CSTA-stop-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24.2.5 Suspend play

```

CSTA-suspend-play-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) suspend-play( 79) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) };

SuspendPlayEvent ::= SEQUENCE
{
    connection          ConnectionID,
    message             MessageID,
    length              [0] IMPLICIT INTEGER    OPTIONAL,
    currentPosition    [1] IMPLICIT INTEGER    OPTIONAL,
    cause               EventCause             OPTIONAL,
    servicesPermitted  ServicesPermitted       OPTIONAL,
    extensions          CSTACommonArguments    OPTIONAL}

END -- of CSTA-suspend-play-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

24.2.6 Suspend record

```

CSTA-suspend-record-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) suspend-record( 80) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACCommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) };

SuspendRecordEvent ::= SEQUENCE
{
    connection          ConnectionID,
    message             MessageID,
    length              [0] IMPLICIT INTEGER OPTIONAL,
    currentPosition    [1] IMPLICIT INTEGER OPTIONAL,
    cause               EventCause OPTIONAL,
    servicesPermitted  ServicesPermitted OPTIONAL,
    extensions          CSTACCommonArguments OPTIONAL}

END -- of CSTA-suspend-record-event

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 24.2.7 Voice attribute changed

```

CSTA-voice-attributes-change-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) voice-attributes-change-event( 74) }

DEFINITIONS ::=
BEGIN
IMPORTS
-- Data Types --
EventCause FROM CSTA-event-causes
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) event-causes( 121) }
ConnectionID FROM CSTA-call-connection-identifiers
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-connection-identifiers( 124) }
MessageID FROM CSTA-device-feature-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) device-feature-types( 127) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
ServicesPermitted FROM CSTA-call-control
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-control( 130) }
Volume FROM CSTA-physical-device-feature
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) physical-device-feature( 137) };

VoiceAttributesChangeEvent ::= SEQUENCE
{
    connection          ConnectionID,
    message             MessageID,
    playVolume          [0] Volume OPTIONAL,
    recordingGain       [1] IMPLICIT INTEGER (0 .. 100) OPTIONAL,
    speed               [2] IMPLICIT INTEGER OPTIONAL,
    currentPosition    [3] IMPLICIT INTEGER OPTIONAL,
    cause               EventCause OPTIONAL,
    servicesPermitted   ServicesPermitted OPTIONAL,
    extensions          CSTACommonArguments OPTIONAL}

END -- of CSTA-voice-attributes-change-event

```

**25 Call detail record services****25.1 Services****25.1.1 Call detail records notification**

```

CSTA-call-detail-records-notification
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-detail-records-notification( 360) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CDRCrossRefID, CDRReason FROM CSTA-call-detail-record
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-detail-record( 132) };

CDRNotification OPERATION ::=
{
    ARGUMENT    CDRNotificationArgument
    RESULT      CDRNotificationResult
    ERRORS      {universalFailure}
    CODE        local: 360
}

CDRNotificationArgument ::= SEQUENCE
{
    cdrCrossRefID  CDRCrossRefID,
    cdrReason      CDRReason          OPTIONAL,
    extensions     CSTACommonArguments OPTIONAL}

CDRNotificationResult ::= CHOICE
{
    extensions     CSTACommonArguments,
    noData         NULL}

END -- of CSTA-call-detail-records-notification

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 25.1.2 Call detail records report

```

CSTA-call-detail-records-report
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-detail-records-report( 361) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
CDRCrossRefID, CDRInfo FROM CSTA-call-detail-record
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) call-detail-record( 132) };

CDRReport      OPERATION ::=
{
    ARGUMENT      CDRReportArgument
    RESULT        CDRReportResult
    ERRORS        {universalFailure}
    CODE          local: 361
}

CDRReportArgument ::= SEQUENCE
{
    cdrCrossRefID          CDRCrossRefID,
    numberOfRecordsSent   INTEGER (1..128),
    cdrInfo                CDRInfo,
    lastStoredCDRReportSent  BOOLEAN          OPTIONAL,
    extensions             CSTACommonArguments  OPTIONAL}
CDRReportResult ::= CHOICE
{
    extensions             CSTACommonArguments,
    noData                NULL}

END -- of CSTA-call-detail-records-report

```

## 25.1.3 Send stored call detail records

```

CSTA-send-stored-call-detail-records
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) send-stored-call-detail-records( 362) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CDRCrossRefID, CDRTIMEPERIOD FROM CSTA-call-detail-record
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-detail-record( 132) };

sendStoredCDR OPERATION ::=
{
  ARGUMENT      SendStoredCDRArgument
  RESULT        SendStoredCDRResult
  ERRORS        {universalFailure}
  CODE          local: 362
}

SendStoredCDRArgument ::= SEQUENCE
{
  cdrCrossRefID  CDRCrossRefID,
  timePeriod     CDRTIMEPERIOD OPTIONAL,
  extensions     CSTACommonArguments OPTIONAL}

SendStoredCDRResult ::= CHOICE
{
  extensions     CSTACommonArguments,
  noData        NULL}

END -- of CSTA-send-stored-call-detail-records

```

## 25.1.4 Start call detail records transmission

```

CSTA-start-call-detail-records-transmission
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) start-call-detail-records-transmission( 363) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CDRCrossRefID, CDRTransferMode FROM CSTA-call-detail-record
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-detail-record( 132) };

startCDRTransmission OPERATION ::=
{
  ARGUMENT StartCDRTransmissionArgument
  RESULT StartCDRTransmissionResult
  ERRORS {universalFailure}
  CODE local: 363
}

StartCDRTransmissionArgument ::= SEQUENCE
{
  transferMode CDRTransferMode,
  extensions CSTACommonArguments OPTIONAL}

StartCDRTransmissionResult ::= SEQUENCE
{
  cdrCrossRefID CDRCrossRefID,
  extensions CSTACommonArguments OPTIONAL}

END -- of CSTA-start-call-detail-records-transmission

```

25.1.5 Stop call detail records transmission

```

CSTA-stop-call-detail-records
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) stop-call-detail-records( 364) }
DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
  {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) extension-types( 129) }
CDRCrossRefID, CDRTermReason FROM CSTA-call-detail-record
  { iso( 1) identified-organization( 3) icd-ecma( 12)
    standard( 0) csta3( 285) call-detail-record( 132) };

stopCDRTransmission      OPERATION ::=
{
  ARGUMENT      StopCDRTransmissionArgument
  RESULT        StopCDRTransmissionResult
  ERRORS        {universalFailure}
  CODE          local: 364
}

StopCDRTransmissionArgument ::= SEQUENCE
{
  cdrCrossRefID   CDRCrossRefID,
  cdrTermReason   CDRTermReason OPTIONAL,
  extensions      CSTACommonArguments OPTIONAL}

StopCDRTransmissionResult ::= CHOICE
{
  extensions      CSTACommonArguments,
  noData          NULL}

END -- of CSTA-stop-call-detail-records

```

STANDARDSISO.COM Click to view the full PDF of ISO/IEC 18052:2000

## 26 Vendor specific extensions services and events

### 26.1 Registration services

#### 26.1.1 Escape register

```

CSTA-escape-register
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-register( 365) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
EscapeRegisterID FROM CSTA-escape-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-types( 135) };

escapeRegister OPERATION ::=
{
    ARGUMENT    EscapeRegisterArgument
    RESULT      EscapeRegisterResult
    ERRORS      {universalFailure}
    CODE        local: 365
}

EscapeRegisterArgument ::= CHOICE
{
    extensions   CSTACommonArguments,
    noData       NULL}

EscapeRegisterResult ::= SEQUENCE
{
    escapeRegisterID  EscapeRegisterID,
    extensions        CSTACommonArguments    OPTIONAL}

END -- of CSTA-escape-register

```

## 26.1.2 Escape register abort

```

CSTA-escape-register-abort
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-register-abort( 366) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
EscapeRegisterID FROM CSTA-escape-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-types( 135) };

escapeRegisterAbort OPERATION ::=
{
    ARGUMENT          EscapeRegisterAbortArgument
    ERRORS             {universalFailure}
    ALWAYS RESPONDS   FALSE
    CODE               local: 366
}

EscapeRegisterAbortArgument ::= SEQUENCE
{
    escapeRegisterID   EscapeRegisterID
    extensions         CSTACommonArguments    OPTIONAL}

END -- of CSTA-escape-register-abort

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 26.1.3 Escape register cancel

```

CSTA-escape-register-cancel
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-register-cancel( 367) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) }
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
EscapeRegisterID FROM CSTA-escape-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-types( 135) };

escapeRegisterCancel OPERATION ::=
{
    ARGUMENT    EscapeRegisterCancelArgument
    RESULT      EscapeRegisterCancelResult
    ERRORS      {universalFailure}
    CODE        local: 367
}

EscapeRegisterCancelArgument ::= SEQUENCE
{
    escapeRegisterID    EscapeRegisterID,
    extensions          CSTACommonArguments    OPTIONAL}

EscapeRegisterCancelResult ::= CHOICE
{
    extensions          CSTACommonArguments,
    noData              NULL}

END -- of CSTA-escape-register-cancel

```

26.2 Services

26.2.1 Escape

```

CSTA-escape-service
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-service( 91) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTASecurityData FROM CSTA-security
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) security( 128) }
CSTACommonArguments, CSTAPrivateData FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) }
EscapeRegisterID FROM CSTA-escape-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) escape-types( 135) };

escape OPERATION ::=
{
    ARGUMENT    EscapeArgument
    RESULT      EscapeResult
    ERRORS      {universalFailure}
    CODE        local: 51
}

EscapeArgument ::= SEQUENCE
{
    escapeRegisterID    EscapeRegisterID        OPTIONAL,
    security             CSTASecurityData       OPTIONAL,
    privateData          CSTAPrivateData
}

EscapeResult ::= CHOICE
{
    extensions    CSTACommonArguments,
    noData        NULL}

END -- of CSTA-escape-service

```

STANDARDSISO.COM: Click to view the full PDF of ISO/IEC 18052:2000

## 26.2.2 Private data version selection

```

CSTA-private-data-version-selection
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) private-data-version-selection( 368) }

DEFINITIONS ::=
BEGIN
IMPORTS
OPERATION, ERROR FROM Remote-Operations-Information-Objects
    {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}
-- Data Types --
universalFailure FROM CSTA-error-definition
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) error-definition( 120) }
CSTACommonArguments FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

privateDataVersionSelection OPERATION ::=
{
    ARGUMENT    PrivateDataVersionSelectionArgument
    RESULT      PrivateDataVersionSelectionResult
    ERRORS      {universalFailure}
    CODE        local: 368
}

PrivateDataVersionSelectionArgument ::= INTEGER

PrivateDataVersionSelectionResult ::= CHOICE
{
    extensions    CSTACommonArguments,
    noData        NULL}

END -- of CSTA-private-data-version-selection

```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## 26.3 Events

### 26.3.1 Private event

```
CSTA-private-event
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) private-event( 71) }

DEFINITIONS ::=
BEGIN

IMPORTS
-- Data Types --
CSTASecurityData FROM CSTA-security
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) security( 128) }
CSTAPrivateData FROM CSTA-extension-types
    { iso( 1) identified-organization( 3) icd-ecma( 12)
      standard( 0) csta3( 285) extension-types( 129) };

PrivateEvent ::= SEQUENCE
{
    security          CSTASecurityData          OPTIONAL,
    privateData      CSTAPrivateData
}

END -- of CSTA-private-event
```

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 18052:2000

## Annex A

(normative)

### Protocol Implementation Conformance Statement (PICS) Proforma

#### A.1 Introduction

The Protocol Implementation Conformance Statement (PICS) is a statement of which capabilities and options of the protocol have been implemented. The PICS can have a number of uses, including use:

- by the protocol implementor, as a check-list to reduce the risk of failure to conform to the standard through oversight;
- by the supplier and acquirer (or potential acquirer) of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma;
- by the user (or potential user) of the implementation, as a basis for initially checking the possibility of interworking with another implementation (note that, while interworking cannot be guaranteed, failure to interwork can often be predicted from incompatible PICS),
- by a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

#### A.2 Conformance

The supplier of a protocol implementation which is claimed to conform to this International Standard shall complete a copy of the Protocol Implementation Conformance Statement (PICS) proforma in A.5, "PICS proforma".

#### A.3 Instructions for completing the PICS proforma

The first part of the PICS proforma, the Implementation Identification (A.4, "Implementation identification", on page 308), is to be completed as indicated with the information necessary to identify fully both the supplier and the implementation.

The main part of the PICS proforma, A.5, "PICS proforma", is a fixed format questionnaire divided into subclauses each containing a group of individual items. These items represent options specified in ISO/IEC 18051 (mandatory items are not represented). Answers to the questionnaire items are to be provided in the appropriate columns by marking an answer to indicate a restricted choice (Yes or No), and optionally clarifying this response with a comment.

Where a service or event is not supported, as indicated by entering No for the first entry in a table, any parameters or dependent service components in the rest of that table are not applicable, it is then not necessary to complete items in any subsidiary entries in the table.

Note that ISO/IEC 18051 should be used as a reference when completing the PICS proforma.

## A.4 Implementation identification

<b>Supplier</b>	
<b>Protocol Version</b>	Phase III, First Edition
<b>Date of Statement</b>	
<b>Contact point for queries about the PICS</b>	
<b>Implementation Name(s) and Version(s)</b>	
<b>Other information necessary for full identification - e.g. Name(s) and Version(s) for machines and/or operating systems; system name(s)</b>	

The first five items are required for all implementations; other information may be completed as appropriate in meeting the requirement for full identification.

The terms Name and Version should be interpreted appropriately to correspond with a supplier's terminology (e.g. Type, Series, Model).

## A.5 PICS proforma

### A.5.1 CSTA Profiles

CSTA Profiles group together services and events, where each profile specifies a set of services and events that are supported by the implementation. Switching Function implementors shall indicate support for one or more profiles in the table below. Specifying support for a profile shall be accompanied by specifying support for the services and events encompassed by the profile, as well as any other services and events that the implementation supports.

Refer to ISO/IEC 18051 for a description of the CSTA Services and Events that must be supported for a specific CSTA profile.

<b>Description: Profile(s) Supported</b>	<b>Supported?</b>		<b>Comments</b>
	<b>Yes</b>	<b>No</b>	
<b>Basic Telephony Profile</b>			
<b>Routeing Profile</b>			

## A.5.2 Capability Exchange Services

### A.5.2.1 Get Logical Device Information

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Get Logical Device Information</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
groupDeviceAttributes			
namedDeviceTypes			
shortFormDeviceID			
agentLogOnModels			
appearanceList			
otherPhysicalDeviceList			
miscMonitorCaps			
associatedGroupList			
maxCallbacks			
maxAutoAnswerRings			
maxActiveCalls			
maxHeldCalls			
maxFwdSettings			
maxDevicesInConf			
transAndConfSetup			
deviceOnDeviceMonitorFilter			
deviceOnConnectionMonitorFilter			
callOnDeviceMonitorFilter			
callOnConnectionMonitorFilter			
mediaClassSupport			
mediaServiceCapsList			

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
connectionRateList			
delayToleranceList			
numberOfChannels			
maxChannelBind			
routeingServList			
logDevServList			
logDevEvtsList			
deviceMaintEvtsList			
security			
privateData			

A.5.2.2 Get Physical Device Information

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Get Physical Device Information</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
groupDeviceAttributes			
namedDeviceTypes			
otherLogicalDeviceList			
deviceModelName			
deviceOnDeviceMonitorFilter			
deviceOnConnectionMonitorFilter			
callOnDeviceMonitorFilter			
callOnConnectionMonitorFilter			
maxDisplays			

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
maxButtons			
maxLamps			
maxRingPatterns			
physDevServList			
physDevEvtsList			
security			
privateData			

#### A.5.2.3 Get Switching Function Capabilities

This service shall be supported by a switching function.

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Get Switching Function Capabilities</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
callCharacteristicsSupported			
mediaClassSupport			
numberOfChannels			
maxChannelBind			
miscMediaCallCharacteristics			
connectionRateList			
delayToleranceRateList			
pauseTime			
currentTime			
messageSeqNumbers			
timeStampMode			

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
securityMode			
securityFormat			
privateDataFormat			
transAndConfSetup			
deviceOnDeviceMonitorFilter			
deviceOnConnectionMonitorFilter			
callOnDeviceMonitorFilter			
callOnConnectionMonitorFilter			
miscMonitorCaps			
correlatorDataSupported			
dynamicFeatureSupported			
callLinkageOptionsSupported			
acdModels			
agentLogonModels			
agentStateModels			
routeingServList			
logDevServList			
logDevEvtsList			
physDevServList			
physDevEvtsList			
deviceMaintEvtsList			
statusReportingServList			
capExchangeServList			
cdrServList			
vendorSpecificServList			
vendorSpecificEvtsList			
privateDataVersionList			
systemStatusTimer			

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
simpleThreshold			
filterThreshold			
mediaServiceCapsList			
security			
privateData			

#### A.5.2.4 Get Switching Function Devices

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Get Switching Function Devices</b>			
<b>Service Request optional parameters</b>			
requestedDeviceID			
requestedDeviceCategory			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

#### A.5.2.5 Switching Function Devices

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Switching Function Devices</b>			
<b>Service Request optional parameters</b>			
segmentID			
security			
privateData			

**A.5.3 System Services**

**A.5.3.1 Change System Status Filter**

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
Change System Status Filter			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

**A.5.3.2 System Register**

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
System Register			
<b>Service Request optional parameters</b>			
requestedStatusFilter			
security			
privateData			
<b>Service Response optional parameters</b>			
actualStatusFilter			
security			
privateData			

## A.5.3.3 System Register Abort

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>System Register Abort</b>			
<b>Service Request optional parameters</b>			
security			
privateData			

## A.5.3.4 System Register Cancel

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>System Register Cancel</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

## A.5.3.5 Request System Status

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Request System Status</b>			
<b>Service Request optional parameters</b>			
sysStatRegisterID			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

**A.5.3.6 System Status**

The System Status service shall be supported as part of the Application Association Initialisation as specified in Clause 7 of ISO/IEC 18051.

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>System Status</b>			
<b>Service Request optional parameters</b>			
sysStatRegisterID			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

**A.5.3.7 Switching Function Capabilities Changed**

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Switching Function Capabilities Changed</b>			
<b>Service Request optional parameters</b>			
sysStatRegisterID			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

## A.5.3.8 Switching Function Devices Changed

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Switching Function Devices Changed</b>			
<b>Service Request optional parameters</b>			
sysStatRegisterID			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

## A.5.4 Monitoring Services

## A.5.4.1 Change Monitor Filter

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Change Monitor Filter</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
actualFilterList			
security			
privateData			

**A.5.4.2 Monitor Start**

This service must be supported if the Basic Telephony profile is supported (A.5.1).

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Monitor Start</b>			
<b>Service Request optional parameters</b>			
requestedMonitorFilter			
monitorType			
requestedMonitorMediaClass			
security			
privateData			
<b>Service Response optional parameters</b>			
actualMonitorFilter			
actualMonitorMediaClass			
monitorExistingCalls			
security			
privateData			

**A.5.4.3 Monitor Stop**

This service must be supported if the Basic Telephony profile is supported (A.5.1).

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Monitor Stop</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

## A.5.5 Snapshot Services

### A.5.5.1 Snapshot Call

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Snapshot Call</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
serviceCrossRefID			
snapshotData			
mediaCallCharacteristics			
callCharacteristics			
callingDevice			
calledDevice			
associatedCallingDevice			
associatedCalledDevice			
correlatorData			
security			
privateData			

### A.5.5.2 Snapshot Device

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Snapshot Device</b>			
<b>Service Request optional parameters</b>			
security			
privateData			
<b>Service Response optional parameters</b>			
serviceCrossRefID			

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
snapshotData			
security			
privateData			

A.5.5.3 Snapshot CallData

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Snapshot CallData</b>			
<b>Service Request optional parameters</b>			
segmentID			
security			
privateData			

A.5.5.4 Snapshot DeviceData

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Snapshot DeviceData</b>			
<b>Service Request optional parameters</b>			
segmentID			
security			
privateData			

## A.5.6 Call Control Services

### A.5.6.1 Accept Call

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Accept Call</b>			
<b>Service Request optional parameters</b>			
correlatorData			
userData			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

### A.5.6.2 Alternate Call

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Alternate Call</b>			
<b>Service Request optional parameters</b>			
connectionReservation			
consultOptions			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

**A.5.6.3 Answer Call**

This service must be supported if the Basic Telephony profile is supported (A.5.1).

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Answer Call</b>			
<b>Service Request optional parameters</b>			
correlatorData			
userData			
security			
privateData			
<b>Service Response optional parameters</b>			
security			
privateData			

**A.5.6.4 Call Back Call-Related**

Description: Service, optional parameters	Supported?		Comments
	Yes	No	
<b>Call Back Call-Related</b>			
<b>Service Request optional parameters</b>			
callCharateristics			
security			
privateData			
<b>Service Response optional parameters</b>			
targetDevice			
security			
privateData			