

ECMA

Standardizing Information and Communication Systems

**Services for Computer Supported
Telecommunications Applications
(CSTA) Phase III**

ECMA

Standardizing Information and Communication Systems

**Services for Computer Supported
Telecommunications Applications
(CSTA) Phase III**

Brief History

This Standard ECMA-269 defines Phase III of Services for Computer Supported Telecommunications Applications (CSTA) for OSI Layer 7 communication between a computing network and a telecommunications network. This 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 only 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. That Technical Report reflected a common understanding of ECMA member companies.

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 attach services, 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.)
- Use of a consistent template for services and events that includes initial/final connection state, connection state transitions, event monitoring sequences, etc.

The First Edition of Standard ECMA-269 was published in December 1997. This Second Edition extends the First Edition in the following areas:

- Call and connection state modelling
- Additional call control services and events
- Features such as correlator data, user data, dynamic feature availability
- Additional conformance criteria, generic service requirements, etc.

Adopted as 2nd Edition of Standard ECMA-269 by the General Assembly of June 1998.

Table of Contents

1	Scope	1
1.1	Status of this Standard	1
1.2	Evolution of CSTA Phase III	1
1.3	Impacts of this Standard	1
2	Conformance	1
2.1	Switching Function	1
2.1.1	Conformant Services	2
2.1.2	Conformant Events	2
2.1.3	CSTA Profiles	2
2.1.4	Support of Service Requests And Manual Mode	2
2.2	Special Resource Function Conformance (to be defined)	3
2.3	Computing Function Conformance	3
3	References	3
3.1	ECMA References	3
3.2	ITU-T References	4
3.3	ETSI References	4
4	Definitions and Abbreviations	4
5	Functional Architecture	4
6	CSTA Operational Model	4
6.1	Switching Sub-Domain Model	5
6.1.1	Switching Sub-Domain Name	6
6.1.2	Application Working Domain	6
6.1.3	Device	6
6.1.4	Call	21
6.1.5	Connection	24
6.1.6	Call State Definitions	27
6.1.7	Referencing Devices, Elements, Appearances and Device Configurations	28
6.1.8	Management of Dynamically-Assigned Identifiers	30
6.2	Special Resource Functions (to be defined)	30
6.2.1	Voice Unit (to be defined)	30
6.2.2	I/O Services (to be defined)	30
6.3	Call Detail Recording (CDR) Services (to be defined)	30
6.4	Capabilities Exchange (to be defined)	30
6.4.1	Switching Function Capabilities (to be defined)	30
6.4.2	Device Capabilities (to be defined)	30
6.4.3	Dynamic Feature Availability	31
6.5	Switching Function Information Synchronization (to be defined)	31

6.5.1	Switching Function Level Information (to be defined)	31
6.5.2	Device Level Information (to be defined)	31
6.5.3	Call Level Information (to be defined)	31
6.6	Status Reporting Services	31
6.6.1	System Status	31
6.6.2	Monitoring	32
6.6.3	Snapshot Services	35
6.7	Additional Services, Features & Behaviour	35
6.7.1	Forwarding	36
6.7.2	Connection Failure	38
6.7.3	Recall	39
6.7.4	Call Back	40
6.7.5	External Calls	40
6.7.6	Tracking a Diverted Call	41
6.7.7	Media Stream Access (to be defined)	41
6.7.8	Routeing Services	41
6.7.9	Device Maintenance	46
6.7.10	Prompting	46
7	Association Establishment (to be defined)	46
8	Security Service (to be defined)	46
9	Generic Service Requirements	46
9.1	Service Request	46
9.2	Service Response (Acknowledgements)	47
9.2.1	Positive Acknowledgement Models	47
9.2.2	Negative Acknowledgement	48
9.3	Diagnostic Error Definitions (to be defined)	48
9.3.1	Operation errors (to be defined)	48
9.3.2	Security errors (to be defined)	48
9.3.3	State incompatibility errors (to be defined)	48
9.3.4	System resource availability errors (to be defined)	48
9.3.5	Subscribed resource availability errors (to be defined)	48
9.3.6	Performance management errors (to be defined)	48
9.3.7	CSTA Private Data Information Errors (to be defined)	48
9.3.8	Unspecified errors (to be defined)	48
9.4	Vendor Specific Extensions (to be defined)	48
9.4.1	Private Data (to be defined)	48
9.4.2	Private Data Version Negotiation (to be defined)	48
9.4.3	Defined Services and Events (to be defined)	48
9.4.4	Escape Services and Private Event (to be defined)	48
9.5	General Services and Event Functional Requirements	48
9.5.1	Services	48
9.5.2	Events	50

10	CSTA Device Identifier Formats	50
10.1	Device Identifier Formats	50
10.1.1	Diallable Digits	50
10.1.2	Switching Function Representation	51
10.1.3	Device Number	53
10.2	Functional Requirements	54
11	Template Descriptions	54
11.1	Service Template	54
11.1.1	Service Description	54
11.1.2	Service Request	55
11.1.3	Service Response	55
11.1.4	Operational Model	55
11.2	Event Template	56
11.2.1	Event Description	56
11.2.2	Event Parameters	56
11.2.3	Event Causes	56
11.2.4	Functional Requirements	56
11.3	Parameter Type Template	56
11.3.1	Parameter Type Description	56
11.3.2	Format	56
11.3.3	Functional Requirements	56
12	Parameter Types	57
12.1	Definitions	57
12.2	Defined Parameter Types	58
12.2.1	AccountInfo	59
12.2.2	AuditoryApparatusList	59
12.2.3	AuthCode	60
12.2.4	MediaCallCharacteristics	60
12.2.5	CallCharacteristics	61
12.2.6	ChargingInfo	61
12.2.7	ConnectionInformation	62
12.2.8	ConnectionList	62
12.2.9	CorrelatorData	63
12.2.10	DisplayList	64
12.2.11	ErrorValue	65
12.2.12	EventCause (to be defined)	66
12.2.13	LocalConnectionState	66
12.2.14	CSTAPrivateData (to be defined)	67
12.2.15	CSTA SecurityData (to be defined)	67
12.2.16	ServicesPermitted	67
12.2.17	SystemStatus	68
12.2.18	SimpleCallState	68

12.2.19	UserData	69
12.3	Identifier Parameter Types	71
12.3.1	AgentID	72
12.3.2	AssociatedCalledDeviceID	72
12.3.3	AssociatedCallingDeviceID	72
12.3.4	AuditoryApparatusID	73
12.3.5	ButtonID	73
12.3.6	CalledDeviceID	73
12.3.7	CallingDeviceID	73
12.3.8	ConnectionID	74
12.3.9	DeviceID	75
12.3.10	DisplayID	76
12.3.11	EscapeRegisterID	76
12.3.12	HookswitchID	76
12.3.13	LampID	76
12.3.14	MediaServiceInstanceID (to be defined)	76
12.3.15	MediaStreamID (to be defined)	76
12.3.16	MonitorCrossRefID	76
12.3.17	NetworkCalledDeviceID	76
12.3.18	NetworkCallingDeviceID	77
12.3.19	RedirectionDeviceID	77
12.3.20	RingerID	78
12.3.21	RouteingCrossRefID	78
12.3.22	RouteRegisterReqID	78
12.3.23	ServiceCrossRefID	79
12.3.24	SubjectDeviceID	79
12.3.25	SysStatRegisterID	79
12.4	Capability Bitmaps Parameter Types (to be defined)	79
13	Capability Exchange Services	80
13.1	Services	80
13.1.1	Get Logical Device Information (to be defined)	80
13.1.2	Get Physical Device Information (to be defined)	80
13.1.3	Get Switching Function Capabilities (to be defined)	80
13.1.4	Get Switching Function Devices (to be defined)	80
13.1.5	Switching Function Devices (to be defined)	80
14	System Services	81
14.1	Registration Services	81
14.1.1	Change System Status Filter	82
14.1.2	System Register	84
14.1.3	System Register Abort	87
14.1.4	System Register Cancel	88
14.2	Services	89
14.2.1	Request System Status	90

14.2.2	System Status	92
14.2.3	Switching Function Capabilities Changed (to be defined)	93
14.2.4	Switching Function Devices Changed (to be defined)	93
15	Monitoring Services	94
15.1	Services	94
15.1.1	Change Monitor Filter	95
15.1.2	Monitor Start	97
15.1.3	Monitor Stop	102
16	Snapshot Services	103
16.1	Services	103
16.1.1	Snapshot Call	104
16.1.2	Snapshot Device	107
16.1.3	Snapshot CallData	110
16.1.4	Snapshot DeviceData	112
17	Call Control Services & Events	115
17.1	Services	115
17.1.1	Accept Call	116
17.1.2	Alternate Call	118
17.1.3	Answer Call	121
17.1.4	Call Back Call-Related	123
17.1.5	Call Back Message Call-Related	126
17.1.6	Camp On Call	129
17.1.7	Clear Call	131
17.1.8	Clear Connection	134
17.1.9	Conference Call	138
17.1.10	Consultation Call	141
17.1.11	Deflect Call	147
17.1.12	Dial Digits	150
17.1.13	Directed Pickup Call	153
17.1.14	Group Pickup Call	156
17.1.15	Hold Call	159
17.1.16	Intrude Call	161
17.1.17	Join Call	165
17.1.18	Make Call	169
17.1.19	Make Predictive Call (to be defined)	174
17.1.20	Park Call	175
17.1.21	Reconnect Call	178
17.1.22	Retrieve Call	180
17.1.23	Single Step Conference Call	182
17.1.24	Single Step Transfer Call	186
17.1.25	Transfer Call	189
17.2	Events	192

17.2.1	Bridged	193
17.2.2	Call Cleared	195
17.2.3	Conferenced	198
17.2.4	Connection Cleared	203
17.2.5	Delivered	207
17.2.6	Digits Dialed	211
17.2.7	Diverted	214
17.2.8	Established	218
17.2.9	Failed	222
17.2.10	Held	227
17.2.11	Network Capabilities Changed	229
17.2.12	Network Reached	232
17.2.13	Offered	236
17.2.14	Originated	240
17.2.15	Queued	243
17.2.16	Retrieved	247
17.2.17	Service Initiated	249
17.2.18	Transferred	252
18	Call Associated Features	256
18.1	Services	256
18.1.1	Associate Data (to be defined)	256
18.1.2	Cancel Telephony Tones (to be defined)	256
18.1.3	Generate Digits (to be defined)	256
18.1.4	Generate Telephony Tones (to be defined)	256
18.1.5	Send User Information (to be defined)	256
18.1.6	Start DTMF Digits Collection (to be defined)	256
18.1.7	Start Telephony Tones Collection (to be defined)	256
18.1.8	Stop DTMF Digits Collection (to be defined)	256
18.1.9	Stop Telephony Tones Collection (to be defined)	256
18.2	Events	257
18.2.1	Call Information (to be defined)	257
18.2.2	Charging (to be defined)	257
18.2.3	DTMF Digits Detected (to be defined)	257
18.2.4	Telephony Tones Detected (to be defined)	257
18.2.5	Service Completion Failure (to be defined)	257
19	Media Attachment Services & Events	258
19.1	Services	258
19.1.1	Attach Media Service (to be defined)	258
19.1.2	Detach Media Service (to be defined)	258
19.2	Events	258
19.2.1	Media Attached (to be defined)	258
19.2.2	Media Detached (to be defined)	258

20	Routeing Services	259
20.1	Registration Services	259
20.1.1	Route Register	260
20.1.2	Route Register Abort	262
20.1.3	Route Register Cancel	263
20.2	Services	264
20.2.1	Re-Route	265
20.2.2	Route End	266
20.2.3	Route Reject	268
20.2.4	Route Request	270
20.2.5	Route Select	272
20.2.6	Route Used	274
21	Physical Device Features	276
21.1	Services	276
21.1.1	Button Press (to be defined)	277
21.1.2	Get Auditory Apparatus Information (to be defined)	277
21.1.3	Get Button Information (to be defined)	277
21.1.4	Get Display (to be defined)	277
21.1.5	Get Hookswitch Status (to be defined)	277
21.1.6	Get Lamp Information (to be defined)	277
21.1.7	Get Lamp Mode (to be defined)	277
21.1.8	Get Message Waiting Indicator (to be defined)	277
21.1.9	Get Microphone Gain (to be defined)	277
21.1.10	Get Microphone Mute (to be defined)	277
21.1.11	Get Ringer Status (to be defined)	277
21.1.12	Get Speaker Mute (to be defined)	277
21.1.13	Get Speaker Volume (to be defined)	277
21.1.14	Set Button Information (to be defined)	277
21.1.15	Set Display (to be defined)	277
21.1.16	Set Hookswitch Status (to be defined)	277
21.1.17	Set Lamp Mode (to be defined)	277
21.1.18	Set Message Waiting Indicator (to be defined)	277
21.1.19	Set Microphone Gain (to be defined)	277
21.1.20	Set Microphone Mute (to be defined)	277
21.1.21	Set Ringer Status (to be defined)	277
21.1.22	Set Speaker Mute (to be defined)	277
21.1.23	Set Speaker Volume (to be defined)	277
21.2	Events	278
21.2.1	Button Information (to be defined)	278
21.2.2	Button Press (to be defined)	278
21.2.3	Display Updated (to be defined)	278
21.2.4	Hookswitch (to be defined)	278
21.2.5	Lamp Mode (to be defined)	278
21.2.6	Message Waiting (to be defined)	278

21.2.7	Microphone Gain (to be defined)	278
21.2.8	Microphone Mute (to be defined)	278
21.2.9	Ringer Status (to be defined)	278
21.2.10	Speaker Mute (to be defined)	278
21.2.11	Speaker Volume (to be defined)	278

22 Logical Device Features 279

22.1	Services	279
22.1.1	Call Back Non-Call-Related (to be defined)	280
22.1.2	Call Back Message Non-Call-Related (to be defined)	280
22.1.3	Cancel Call Back (to be defined)	280
22.1.4	Cancel Call Back Message (to be defined)	280
22.1.5	Get Agent Status (to be defined)	280
22.1.6	Get Auto Answer (to be defined)	280
22.1.7	Get Caller ID Status (to be defined)	280
22.1.8	Get Do Not Disturb (to be defined)	280
22.1.9	Get Forwarding (to be defined)	280
22.1.10	Get Last Number Dialed (to be defined)	280
22.1.11	Get Routeing Mode (to be defined)	280
22.1.12	Set Agent Status (to be defined)	280
22.1.13	Set Auto Answer (to be defined)	280
22.1.14	Set Caller ID Status (to be defined)	280
22.1.15	Set Do Not Disturb (to be defined)	280
22.1.16	Set Forwarding (to be defined)	280
22.1.17	Set Routeing Mode (to be defined)	280
22.2	Events	281
22.2.1	Agent Busy (to be defined)	281
22.2.2	Agent Logged Off (to be defined)	281
22.2.3	Agent Logged On (to be defined)	281
22.2.4	Agent Not Ready (to be defined)	281
22.2.5	Agent Ready (to be defined)	281
22.2.6	Agent Working After Call (to be defined)	281
22.2.7	Auto Answer (to be defined)	281
22.2.8	Call Back (to be defined)	281
22.2.9	Call Back Message (to be defined)	281
22.2.10	Caller ID Status (to be defined)	281
22.2.11	Do Not Disturb (to be defined)	281
22.2.12	Forwarding (to be defined)	281
22.2.13	Routeing Mode (to be defined)	281

23 Device Maintenance Events 282

23.1	Events	282
23.1.1	Back In Service (to be defined)	282
23.1.2	Device Capabilities Changed (to be defined)	282
23.1.3	Out Of Service (to be defined)	282

24	I/O Services	283
24.1	Registration Services	283
24.1.1	I/O Register (to be defined)	283
24.1.2	I/O Register Abort (to be defined)	283
24.1.3	I/O Register Cancel (to be defined)	283
24.2	I/O Services	284
24.2.1	Data Path Resumed (to be defined)	284
24.2.2	Data Path Suspended (to be defined)	284
24.2.3	Fast Data (to be defined)	284
24.2.4	Resume Data Path (to be defined)	284
24.2.5	Send Broadcast Data (to be defined)	284
24.2.6	Send Data (to be defined)	284
24.2.7	Send Multicast Data (to be defined)	284
24.2.8	Start Data Path (to be defined)	284
24.2.9	Stop Data Path (to be defined)	284
24.2.10	Suspend Data Path (to be defined)	284
25	Voice Unit Services & Events	285
25.1	Services	285
25.1.1	Concatenate Message (to be defined)	285
25.1.2	Delete Message (to be defined)	285
25.1.3	Play Message (to be defined)	285
25.1.4	Query Voice Attribute (to be defined)	285
25.1.5	Record Message (to be defined)	285
25.1.6	Reposition (to be defined)	285
25.1.7	Resume (to be defined)	285
25.1.8	Review (to be defined)	285
25.1.9	Set Voice Attribute (to be defined)	285
25.1.10	Stop (to be defined)	285
25.1.11	Suspend (to be defined)	285
25.1.12	Synthesize Message (to be defined)	285
25.2	Events	286
25.2.1	Play (to be defined)	286
25.2.2	Record (to be defined)	286
25.2.3	Review (to be defined)	286
25.2.4	Stop (to be defined)	286
25.2.5	Suspend Play (to be defined)	286
25.2.6	Suspend Record (to be defined)	286
25.2.7	Voice Attribute Changed (to be defined)	286
26	Call Detail Record (CDR) Services	287
26.1	Services	287
26.1.1	Get Call Detail Records (to be defined)	287
26.1.2	Call Detail Recording Report (to be defined)	287
26.1.3	Send Stored Call Detail Records (to be defined)	287

26.1.4	Start Call Detail Records Transmission (to be defined)	287
26.1.5	Stop Call Detail Records Transmission (to be defined)	287
27	Vendor Specific Extensions Services & Events	288
27.1	Registration Services	288
27.1.1	Escape Register (to be defined)	288
27.1.2	Escape Register Abort (to be defined)	288
27.1.3	Escape Register Cancel (to be defined)	288
27.2	Services	288
27.2.1	Escape (to be defined)	288
27.2.2	Private Data Version (to be defined)	288
27.3	Events	288
27.3.1	Private Event (to be defined)	288
Annex A	Connection State Transition Examples (Informative)	289
Annex B	Device Appearances (Normative)	299
B.1	Standard Appearance	299
B.1.1	Selected-Standard Appearance	299
B.1.2	Basic-Standard Appearance	300
B.2	Bridged Appearance	300
B.2.1	Basic-Bridged	301
B.2.2	Exclusive-Bridged	302
B.2.3	Shared-Bridged	303
Annex C	Examples of Device Appearances (Informative) (to be defined)	307
Annex D	ISDN User-User Information Element Encoding for CSTA (Normative)	309

1 Scope

Services and Events Reports supported by Computer-Supported Telecommunications Applications, Phase III (CSTA) are defined in this Standard.

This Standard is focused on providing application service interfaces to a Switching Function, Computing Function and a Special Resource Function. A CSTA application interface is disassociated from the various user-network interfaces and network-network interfaces CSTA applications may serve, observe or manipulate. Because CSTA operates with existing telecommunications interfaces indirectly, it operates generically, so that differences among various existing interfaces are hidden from CSTA applications. Support of user-to-network interfaces is outside the scope of CSTA.

Although most terminal equipment (TE) are suitable for use with CSTA there will be instances of TE that will not be suitable in certain circumstances. Examples are:

- FAX terminals and modems that are unable to adjust their transmission modes to prevent carrier conflict when both parties are alerted via CSTA during call establishment;
- Functional terminals that perform telecommunication functions outside the control of the Switching Function.

Services defined in this Standard allow functional integration between a computing network and a telecommunications network. Computing platforms (i.e., Application Programming Interfaces - APIs) that support such functionally-integrated applications are outside the scope of this Standard.

Communication between the computing and switching (i.e., telecommunications) networks may take place via intervening networks ranging from simple point-to-point connections to local- or wide-area telecommunications networks.

This Standard is part of a suite of CSTA Standards and Technical Reports that provide a comprehensive description of the architectural and practical issues involved in applying, implementing, and utilizing CSTA-based CTI applications.

1.1 Status of this Standard

The contents of this Standard will form part of a future completed Standard, and provides guidance to the structure of the complete Standard.

This Standard provides some of the many new features of CSTA Phase III.

1.2 Evolution of CSTA Phase III

The basis of the contents of the completed Standard currently exists and will be published by ECMA in new editions of this Standard as they are finalised, extending the content until the Standard is complete.

1.3 Impacts of this Standard

Within this Standard, currently known extensions are identified as clauses “to be defined”, with occasional reference to such clauses, and these clauses will be defined over the span of new editions of this Standard.

2 Conformance

This Clause specifies the conformance requirements for a Switching Function, Special Resource Function, and a Computing Function.

Conformance requirements specify the parts of this Standard that a CSTA conformant implementation shall support.

This Standard specifies an operational model (Clause 6, “CSTA Operational Model” and Clause 9, “Generic Service Requirements”) that defines a collection of objects (e.g. domains and sub-domains, logical and physical elements, calls) and the relationships between these objects.

The behaviours of CSTA-conformant services, features, and event reports are determined by this model.

2.1 Switching Function

In order to conform to this Standard a switching function shall support the following as a minimum:

1. the requirements pertaining to CSTA features as specified in Clause 6, “CSTA Operational Model”.

2. the requirements as specified in Clause 9, “Generic Service Requirements”.
3. the requirements of the capability exchange services to be specified in the future 3rd edition of this Standard.
4. at least one of the profiles as specified in 2.1.3, “CSTA Profiles”.

2.1.1 Conformant Services

In order to conform to a specific CSTA *service* an implementation shall support the following as a minimum:

1. the requirements of the service as specified by its service description, service request parameters, service response parameters, and operational model including connection state transitions, monitoring event sequences, and functional requirements.
2. the requirements associated with each parameter used in the service as specified by its parameter description, format, and functional requirements in Clause 12, “Parameter Types”.
3. all of the events that are associated with its service completion criteria, as documented in its event monitoring tables.
4. service requests that contain a device identifier parameter, an implementation shall support, at a minimum, the Diallable Digits format as specified in 10.1.1, “Diallable Digits”.

2.1.2 Conformant Events

In order to conform to a specific CSTA *event* an implementation shall support the following as a minimum:

1. the requirements of the event as specified by its event description, event parameters, event causes, and functional requirements.
2. the requirements associated with each parameter used in the event as specified by its parameter description, format, and functional requirements in Clause 12, “Parameter Types”.
3. events that contain a device identifier parameter, an implementation shall support, at a minimum, the Switching Function Representation format as specified in 10.1.2, “Switching Function Representation”.

2.1.3 CSTA Profiles

Some CSTA services and events are grouped together as profiles.

2.1.3.1 Basic Telephony Profile

This profile includes the following:

1. CSTA Services: Answer Call, Clear Connection, Make Call, Monitor Start (with the monitorType of device-type), and Monitor Stop.
2. CSTA Events: Connection Cleared, Delivered, Established, Failed, Network Reached, Originated, and Service Initiated.

Other CSTA services and events may be provided in any combination in addition to this set.

2.1.3.2 Routeing Profile

If the switching function supports Routeing Services as specified in Clause 20, “Routeing Services”, it shall support a minimum set of Routeing Services that includes: Route Request, Route Select, and Route End (from the switching function only).

Other Routeing services may be provided in any combination in addition to this set.

If a switching function supports the routeing for digital data calls, then the Route Register and CSTA Route Register Cancel shall also be included in the minimum set.

2.1.4 Support of Service Requests And Manual Mode

A conformant switching function may support a given service defined in this Standard through the CSTA service boundary but is not required to support the equivalent service in a manual mode.

A conformant switching function may support a feature associated with an equivalent CSTA service defined in this Standard through manual mode but is not required to support the equivalent service through the service boundary.

2.2 Special Resource Function Conformance (to be defined)

2.3 Computing Function Conformance

In order to conform to this Standard a computing function shall support the following as a minimum:

1. the requirements pertaining to CSTA features as specified in Clause 6, "CSTA Operational Model".
2. the requirements as specified in Clause 9, "Generic Service Requirements".
3. for a supported service, the computing function shall not reject as unsupported all of the events specified in the monitoring event sequences associated with the service.
4. the "Single Physical and Logical Element" and "Logical Element Only" device configurations as specified in 6.1.3.3, "Device Configurations".
5. for service requests, the Diallable Digits format of Device Identifiers as specified in 10.1.1, "Diallable Digits".
6. for events, all formats of Device Identifiers as specified in 10.1, "Device Identifier Formats".
7. the "No Appearance Addressability" and the "Individual Appearance Addressability" of referencing device elements as specified in 6.1.7, "Referencing Devices, Elements, Appearances and Device Configurations".
8. both types of service request acknowledgment models (e.g., Atomic and Multi-Step) as specified in 9.2, "Service Response (Acknowledgements)".
9. all failure models as specified in 6.7.2, "Connection Failure".
10. both switching function options of handling unsupported parameters in service requests as specified in the capability exchange services.
11. both the fixed and local view of the primaryOldCall and the secondaryOldCall parameters in the Conferenced and the Transferred events.
12. all bi-directional services for which it registered, whether explicitly (i.e., via a service registration service such as System Status Register) or implicitly (i.e., the switching function does not support registration but does support (as indicated through the capabilities exchange services) a particular bi-directional service and therefore may issue a service request to the computing function).

3 References

The references used in this Standard are defined in the following sections.

3.1 ECMA References

- ECMA-143** Private Integrated Services Network (PISN) - Circuit mode bearer services - Inter-Exchange Signalling Procedures and Protocol (QSIG-BC), 3rd edition (June 1997)
- ECMA-155** Private Integrated Services Network (PISN) - Addressing, 2nd edition (June 1997)
- ECMA TR/72** Glossary of Definitions and Terminology for Computer Supported Telecommunications Applications (CSTA) Phase III (June 1998)

3.2 ITU-T References

- E.131** Subscriber Control Procedures for Supplementary Telephone Services (7) (1988)
- E.160** Definitions Relating to National and International Numbering Plans (6) (1993)
- E.161** Arrangement of Digits, Letters and Symbols on Telephones and Other Devices that can be used for Gaining Access to a Telephone Network (7) (1993)
- E.164** The International Public Telecommunication Numbering Plan (7) (1997)
- Q.931** Digital Subscriber Signalling System No. 1 (DSS 1) - ISDN User-Network Interface layer 3 Specification for Basic Call Control (9) (1993)

3.3 ETSI References

- ETS 300 182** Integrated Services Digital Network (ISDN); Advice of Charge (AOC) Supplementary Service; Digital Subscriber Signalling System No. One (DSS1) Protocol; Part 1: Protocol Specification (1993)

4 Definitions and Abbreviations

The definitions and abbreviations used in this Standard are defined in *Glossary of Definitions and Terminology for Computer Supported Telecommunications Applications (CSTA) Phase III*, ECMA TR/72.

5 Functional Architecture

The objective of CSTA Architecture is to define the inter working mechanisms among Computing, Switching and Special Resource Functions independently from their physical implementations.

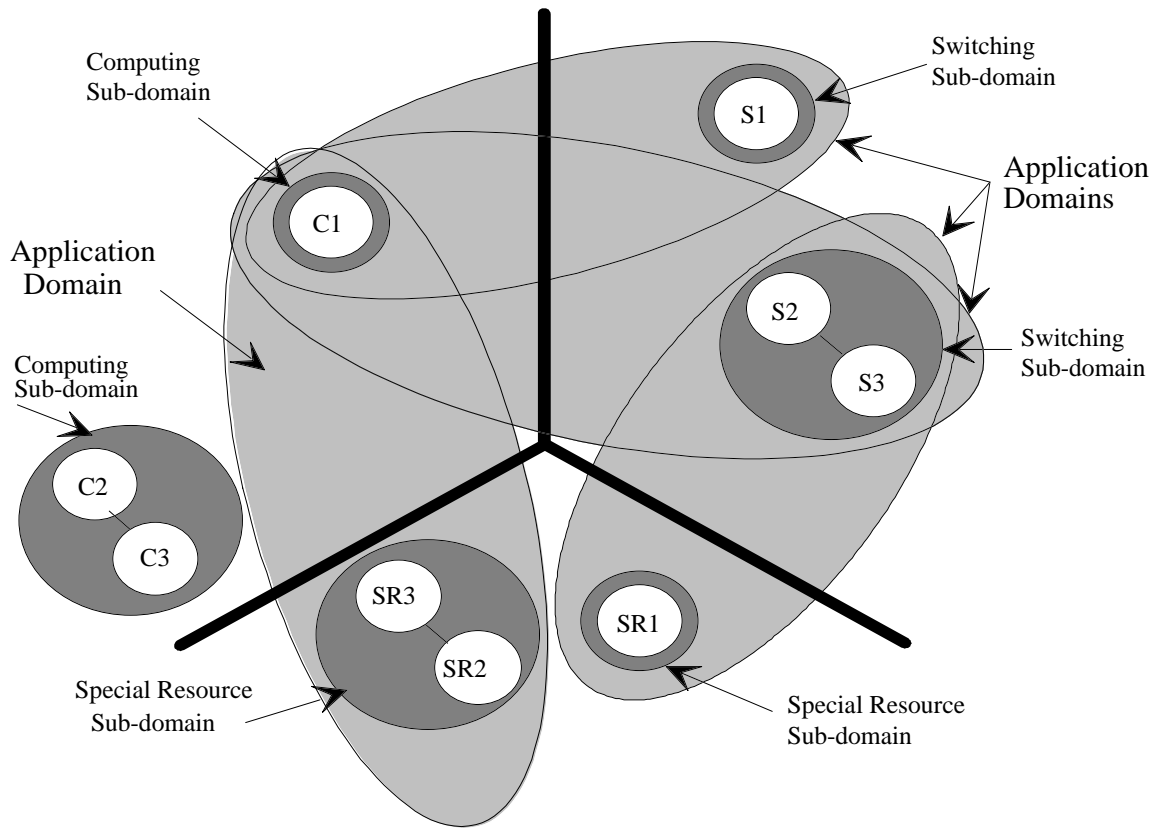
The concepts of: distribution of Computing, Switching and Special Resource Functions, CSTA Service, client server model, and CSTA objects as abstracted at a CSTA Service Boundary, are specified in another part (to be defined) of the CSTA Phase III Suite.

6 CSTA Operational Model

The operational model considered for CSTA is summarized in this clause.

The set of accessible Computing, Switching and Special Resource Functions from which an application might receive service defines a CSTA domain. An example of a CSTA domain is shown in the next figure. The CSTA domain contains switching, computing and special resource domains that are divided in the figure by the heavy lines. The special resource, switching and computing domains comprise Computing Functions (C1, C2 and C3), Switching Functions (S1, S2 and S3), and Special Resource Functions (SR1, SR2 and SR3). Each function can provide to a CSTA application, a view of the domain in which the function resides. Each such view defines a sub-domain. If one or more functions provides an identical view, then these functions are part of the same sub-domain. CSTA applications encompass at least two different sub-domains, and are represented in the next figure as application domains.

Figure 6-1 Domains and Sub-Domains



Note that a function may provide a view to an application that includes not only the objects within its sub-domain, but also the objects it can view in another (presumably related) sub-domain. For example (in Figure 6-1), a computing domain {C1} may receive a view of a switching sub-domain from a switching domain {S2+S3}. That switching sub-domain may receive a view of a special resource sub-domain from a special resource domain {SR1}, and relay that view, in addition to the view of its sub-domain, to the Computing Function. This relay may preserve two views of separate switching and special resource sub-domains, or it may provide a combined view of a switching/special-resource sub-domain. As shown in the figure, {C1} also may have its own, direct view of a special resource sub-domain {SR2+SR3}. Finally, {C2+C3} represent a computing domain that is potentially, but not yet, involved in CSTA transactions with other sub-domains because an association has not yet been established between any other sub-domain and {C2+C3}.

6.1 Switching Sub-Domain Model

The tools needed to provide an abstract view of the Switching Function are defined by the switching sub-domain model. This model allows an application to conceptualize the Switching Function's operation. To provide this abstract view, CSTA defines several CSTA switching sub-domain model Objects that can be observed and acted upon by the Switching Function on behalf of the Computing Function. Those objects include CSTA Devices, Calls, and Connections.

The concepts discussed in this section have also been introduced in another part (to be defined) of the CSTA Phase III Suite. Refer to this reference for an introduction to these concepts.

6.1.1 Switching Sub-Domain Name

The switching function is identified by a unique name within the switching domain known at association time. This name may be used by the computing function to identify that different CSTA applications resulting from different associations are operating in the same switching sub-domain.

6.1.2 Application Working Domain

The application working domain is the subset of devices (and the calls and connections associated with those devices) inside a switching sub-domain that are controllable and/or monitorable over a CSTA Service Boundary. This subset is known at association time. The scope of this working domain may result from considerations like the application's design, licensing policy, security constraints, etc., and is under administration of the switching sub-domain. Different CSTA applications operating in the same switching sub-domain can have different working domains or share totally or partially the same working domain.

6.1.3 Device

CSTA enables manipulation and observation of devices that allow users to access telecommunications services.

NOTE

It is not claimed that this Standard alone supports ISDN (or any other) devices because, for example, of the additional information required to support such devices in PISNs. CSTA only provides a facility for passing ISDN (or other) specific information to allow, for example, a selection among ISDN devices sharing the same directory number (bearer capability, subaddress, etc.). Another example, that applies generally to telecommunications networks (including ISDN and OSI), is specifying the originator for a call that is established via CSTA. With the current signalling support, each party in a call can act only as a called party because the "network" is acting to originate the call. This situation has implications for both the network-to-terminal signalling and any application-level signalling that is significant to the calling party (e.g., issuing A_Associate).

Devices that are visible or controllable via CSTA are known as *CSTA Devices*.

CSTA Devices can be either physical devices (such as buttons, lines, trunks, and stations) or logical devices (such as groups of devices, pilot numbers, and automatic call distribution groups). CSTA Devices have attributes that allow CSTA to monitor and manipulate them. The attributes of any CSTA Device shall be:

1. **Device Type** - differing types of CSTA Device can be used for various purposes and can be manipulated and observed differently within CSTA. CSTA Device Types are listed and defined in 6.1.3.4, "Device Categories".
2. **Media Characteristics** - CSTA devices have distinct capabilities and characteristics defined by their media features. CSTA represents these characteristics by the following attributes.
 - **Media Class:** A CSTA Device shall belong to at least one and may belong to more than one media class. The Media Class can be used in Call Control services to help select a device for a call or it can be used in call control events to report the media class associated with the call. The following media classes are defined in CSTA:
 - **Audio** - 3.1 KHz audio. Devices in this class are used to make audio calls excluding speech calls. It includes G3 FAX and facsimile machines.
 - **Data** - Devices in this class are used to make digital data calls (both circuit switched and packet switched). This class includes digital computer interfaces and G4 facsimile machines.
 - **Image** - Devices in this class are used to make digital data calls involving imaging, or high-speed, circuit-switched data in general. This class includes digital video telephones and CODECs.
 - **Voice** - Devices in this class are used to make speech calls. This class includes standard telephones.
 - **Other** - A class comprising devices not in the Data, Image, Audio or Voice classes.
 - **Media Stream Information:** The media stream associated with a CSTA Device has attributes such as **Connection Rate**, **Bit Rate**, and **Delay Tolerance**. This information can be used in CSTA services to help select the media stream information for a call or to report the media stream information associated with an existing call.

- **Protocol Specific Information:** Many protocols provide additional information beyond what is standardized in CSTA to help distinguish devices. CSTA provides a mechanism where protocol specific information can be passed in CSTA messages to help select a specific device for a call or to provide additional information about the protocol specific information associated with the call. This information consists of:
 - The type of call control information elements (ISDN, for example).
 - A character string that contains the protocol specific information elements. For example, in ISDN, the information may include Bearer Capability, Subaddress (for both calling and called devices), High Layer compatibility, and Low Layer compatibility as defined in IS 11572: 1993.

Refer to 12.2.4, “MediaCallCharacteristics”, for a description of the Media Characteristics that are used in Call Control services to select devices for a call and in Call Control events to report the media Characteristics associated with the devices involved with the call.

3. **CSTA Device Identifier** - Each device that can be observed and/or manipulated shall be referenced across the CSTA Service Boundary. To accomplish this, each device shall be identified using a Device Identifier.
 - Throughout this standard, the term *Device Identifier* shall always mean *CSTA Device Identifier*.
 - Device Identifiers may be static or, only when used in the context of a connection identifier, dynamically-assigned.
 - A static Device Identifier shall be stable over time. It shall remain constant and unique between calls, associations and within both the switching and computing functions. An example of a static Device Identifier is an ITU-T E.164 Directory Number.
 - It may be useful for the Switching Function to convert a Device Identifier to another static form for use in service interactions. An example, it might be useful to transform a Public Directory Number into a Private Directory Number. This transformation allows service interactions to be independent of the identification mechanism and allows reduction in the amount of data exchanged. This shortened form of Device Identifier is known as a CSTA Short Form Device Identifier.
 - A static Device Identifier may be used in conjunction with “MediaCallCharacteristics”, as specified in 12.2.4, “MediaCallCharacteristics”, on page 60, in order to distinguish among CSTA Devices that share a Device Identifier.
 - A dynamically-assigned Device Identifier is temporary (lasting for the duration of a call) and may be created at any appropriate time. Once a CSTA Device has been included in a call, it may be desirable to continue to refer to the particular instance of the CSTA Device associated with this call for manipulation or tracking. A static Device Identifier may not always be sufficient because it may not be available or because it is too long and cumbersome for efficient use. In these cases the Switching Function can dynamically assign a Device Identifier as a device reference or handle for the duration of the call. Management of the dynamically-assigned Device Identifier is discussed in 6.1.8.
 - The **Device Identifier Status** indicates if an actual Device Identifier is being provided in a parameter or the reason why it is not being provided. The set of possible values for the Device Identifier Status is:
 - *Provided* - A Device Identifier is present.
 - *Not Known* - Indicates that the switching function cannot provide the Device Identifier but knows that the device exists.
 - *Not Required* - Indicates that the device is not relevant in this case.
 - *Not Specified* - Indicates that the device cannot be specified.
 - The parameter type associated with a particular Device Identifier determines how it is interpreted, restrictions on its use, and the Device Identifier Statuses that are applicable. These parameter types (AssociatedCalledDeviceID, AssociatedCallingDeviceID, CallingDeviceID, CalledDeviceID, DeviceID,

RedirectionDeviceID, and SubjectDeviceID) are specified in 12.3, “Identifier Parameter Types”, on page 71.

- The format of a Device Identifier is specified in Clause 10, “CSTA Device Identifier Formats”.
4. **Device State** - A CSTA Device itself does not have a state or status directly associated with it. The elements, components and calls associated with the CSTA Device do have states and statuses associated with them. The following is the list of these states and statuses associated with a CSTA Device:
- A connection state is the state of a CSTA Device’s logical element’s connection into a call. This state is associated with Call Control features/services. For more information on connection states, refer to Clause 6.1.5, “Connection”, beginning on page 24.
 - The status of the physical components associated with a physical element of a CSTA Device. (e.g., the hookswitch status). For more information, refer to Clause 21, “Physical Device Features”, beginning on page 276.
 - The status of the logical device features/services associated with a logical element of a CSTA Device. (e.g., the forwarding and do not disturb status). For more information, refer to Clause 22, “Logical Device Features”, beginning on page 279.
5. **Device Elements** - A CSTA Device represents various types of telephony endpoints in a switching sub-domain and allows access to telephony services. A CSTA Device can range from a single endpoint (e.g., station) to a set of associated endpoints that form a group. Each CSTA Device is represented by its attributes (e.g., identifier, state(s), type) as well as its features/services. These attributes/features/services are grouped into two categories which are referred to as *device elements*. A device element encompasses the control and observation of a specific set of CSTA Device attributes/features/services. The device elements are: *physical element* and *logical element*.

The logical element of a CSTA Device encompasses the set of attributes/features/services (e.g., Make Call, Set Forward) that have any association with the control and observation of a call at a CSTA Device (i.e., connection). The physical element of a CSTA Device encompasses the set of attributes/feature/services that have any association with physical components of the CSTA Device that would potentially make up the user interface of the device.

All addressable (i.e., has a single identifier) CSTA Devices consist of one of the following device element combinations.

Figure 6-2 Logical Element Only

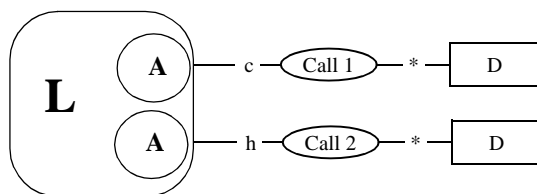


Figure 6-3 Physical Element Only

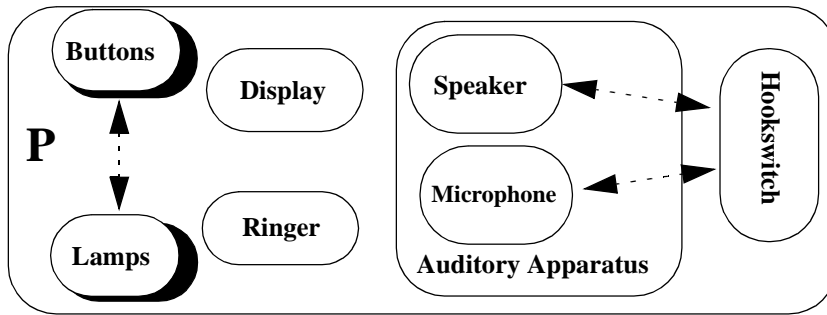
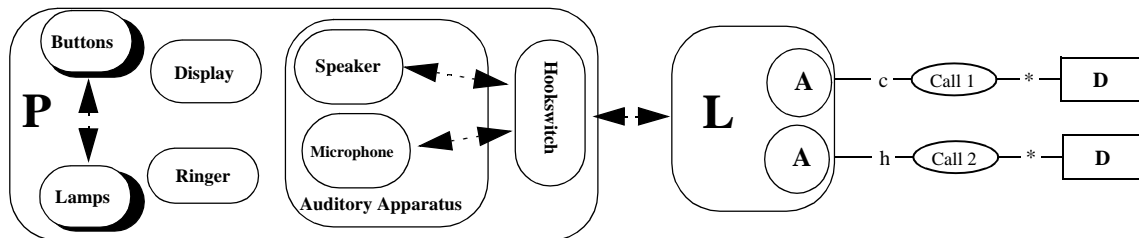
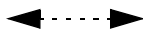


Figure 6-4 A Logical and Physical Element



- D represents another device
- P represents the identifiers for the physical elements
- L represents the identifiers for the logical elements
- A represents an appearance of a logical element



indicates that there is an interaction and/or association between the elements or components of an element.

For example, a “plain old telephone set” (POTS) consists of a logical and physical element. A computing function learns about the devices, their elements, and their associated attributes/features/services in a switching sub-domain by using the capabilities exchange services (refer to 13.1 beginning on page 80). The following sections describes the device elements and the device attributes/features/services that are associated with the elements in detail.

6.1.3.1 Physical Element

The physical element represents the attributes of the physical components and their associated features/services that make up the user interface of a device (e.g., the components of a telephone set). A physical component at a CSTA Device can be a piece of hardware or a virtual (e.g., software) representation of a piece of hardware. For example, a set of buttons on a device (i.e., physical components of the device) could be comprised of a piece of hardware with 12 buttons and a switching function software representation of 12 more buttons. As a result, the device has a set of 24 buttons associated with its physical element.

The combinations of physical components associated with the physical element are switching function specific. The following features/services are controlled and observed through the physical element:

- The Physical Device features/services, such as Button Press, Get/Set Hookswitch Status, and Get/Set Speaker Volume.
- The I/O Services such as Start Data Path and Send Data (when applied to the physical element of a device).

Note that these features/services also include the associated events and monitoring of these events.

The physical element of a device is observed and/or controlled within the switching function through an assigned Device Identifier.

Note that if the device is a combination of logical and physical elements, the assigned Device Identifier is the same for both elements.

In order for the physical components to interact or be associated with calls at the device, the device shall have a logical device element (for more details, see 6.1.3.2, “Logical Element”) and/or some association(s) with a logical element(s) from another device(s) (for more details, see 6.1.3.3, “Device Configurations”). The physical components interact with calls through these logical device element(s) but it is device and switching function specific as to how these components actually interact and are associated with the calls.

The following sections describe the physical components that can be associated with the physical element of the device. All physical components shall be controlled and observed in conjunction with the physical element (i.e., associated with the physical element’s Device Identifier).

6.1.3.1.1 **Auditory Apparatus**

An auditory apparatus is a component which is used to convert electronic signals into voice/speech (i.e., a speaker) and/or convert voice/speech into electronic signals (i.e., a microphone) but at a minimum shall have either a speaker or a microphone. A physical element can have several auditory apparatuses associated with it. Each auditory apparatus can be used independent of each other. An auditory apparatus has several attributes which can be controlled and observed by a computing function. The following are those attributes:

1. *Auditory Apparatus type* - There are several types of auditory apparatuses. Each type representing a different physical configuration and/or function. The following is the list of auditory apparatus types:
 - a. *Handset* - An auditory apparatus that is held in a person’s hand and contains a microphone and speaker.
 - b. *Headset* - An auditory apparatus that is worn on a person’s head and contains a microphone and speaker.
 - c. *Speakerphone* - An auditory apparatus that does not require a person’s body to come into contact with the apparatus and contains a microphone and speaker.
 - d. *Speaker-only phone* - A Speakerphone without a microphone.
 - e. *Microphone-only*- A type that provides only a microphone.
 - f. *Other* - An auditory apparatus that is unique to the given switching function.
2. *Auditory Apparatus Identifier* - Each auditory apparatus that can be observed and/or controlled within the switching function is referenced using an assigned identifier. The auditory apparatus identifiers associated with a given physical element’s Device Identifier are unique. This identifier is used to control and observe all auditory apparatus attributes except the hookswitch which is associated with the apparatus.
3. *Microphone* - The auditory apparatus may or may not have a microphone. If a microphone is present at the auditory apparatus, then there are two features of the microphone that may or may not be controlled (i.e., settable) and observed (i.e., readable).
 - a. *Gain* - This is the level at which the microphone is generating the electronic signal. For more details, refer to the definition of the microphoneGainValue parameter in associated services and events.
 - b. *Mute* - This is the capability to temporarily disable the microphone. For more details, refer to the definition of the microphoneMute parameter in associated services and events.Both of these features are controlled and observed using the auditory apparatus identifier.
4. *Speaker* - The auditory apparatus may or may not have a speaker. If a speaker is present at the auditory apparatus, then there are two features of the speaker that may or may not be controlled (i.e., settable) and observed (i.e., readable).
 - a. *Volume* - This is the level at which the speaker is boosting the electronic signal when generating the associated voice sound waves. For more details, refer to the definition of the speakerVolumeValue parameter in associated services and events.

- b. *Mute* - This is the capability to temporarily disable the speaker. For more details, refer to the definition of the speakerMute parameter in associated services and events.

Both of these features are control and observed using the auditory apparatus identifier.

5. *Hookswitch association* - This identifies the particular hookswitch that is used to activate (i.e., put into use) and deactivate (i.e., remove from use) the auditory apparatus. It also indicates, if the particular hookswitch can be controlled (i.e., settable) and observed. For more details on hookswitches, refer to associated services and events.

6.1.3.1.2 Hookswitch

A hookswitch is a component which is used to activate (i.e., put into use or off-hook) or deactivate (i.e., remove from use or on-hook) an auditory apparatus(es). When a hookswitch is off-hook, it enables the auditory apparatus(es) to transmit and receive the electronic signals associated with sound, and when it is on-hook, this capability is disabled. A physical element can have several hookswitches associated with it. Each hookswitch can be used independently of each other. A hookswitch has one attribute which can be controlled and observed by a computing function. This attribute is the hookswitch identifier. This identifier is assigned by the switching function. The hookswitch identifiers associated with a given physical element's Device Identifier are unique. This identifier is used to control and observe the status of the particular hookswitch (i.e., on-hook, off-hook).

6.1.3.1.3 Button

A button is a component which executes a specific feature/service that is assigned to it. The most common implementation of this component, is a piece of hardware that is pressed and released, thereby executing the feature/service assigned to it (e.g., each of the number buttons on a station). However, an implementation can use any component that can produce a similar behaviour. A button can also have the capability of toggling between two settings of a feature/service (e.g., enabling and disabling Do Not Disturb, that is you press the button once, it enables the Do Not Disturb feature/service, and press again, it disables the feature/service). The button can also have the capability of looping through a series of features/services. Almost any feature/service or set of features/services can be assigned to a button, but generally a switching function makes visible buttons only with features/services that are not available through the components/attributes/features/services defined in this Standard. A button can also be used to represent another physical component (e.g., a hookswitch). A physical element can have many buttons associated with it. Each button can be used independently of each other. A button has the following attributes which can be controlled and observed by the computing function:

1. *Button Identifier* - Each button that can be observed and/or controlled within the switching function is referenced using an assigned identifier. The button identifiers associated with a given physical element's Device Identifier are unique. This identifier is used to control and observe all other button attributes.
2. *Button Label* - This is a label by which people interacting with the physical device refer to a given button. This label is a character string which is retained by the switching function. This attribute can also be changed (if supported) by the computing function. The meaning of a Button Label is specific to the users of a particular device and changing it does not change the function of the button.
3. *Button Function* - This is a feature which can be assigned by the switching function to describe the function associated with a given button. The switching function may reassign the functionality of a button and change this attribute as required (in response to other button presses, for example) but it may not be changed directly by the computing function.
4. *Button Associated Number* - This is a diallable digits format Device Identifier which is associated with the feature/service assigned to the button. This Device Identifier is used by the feature/service when it is executed by the button being pressed. This attribute is optional and only applies to buttons that have some form of associated number. This Device Identifier is initially assigned by the switching function and can be changed (if supported) at any time by either the switching or computing function.

6.1.3.1.4 Lamp

A lamp is a component that represents (i.e., indicates), for example, the status of, for example, a feature/service, physical component, logical device element or other CSTA device. The most common implementation of this component is a piece of hardware that emits light. However, an implementation can use any component that can

produce a similar behaviour (e.g., an icon presented on a display). A physical element can have many lamps associated with it. Each lamp can be used independently of each other. A lamp has several attributes which can be controlled and observed by the computing function. The following are those attributes:

1. *Lamp Identifier* - Each lamp that can be observed and/or controlled within the switching function is referenced using an assigned identifier. The lamp identifiers associated with a given physical element's Device Identifier are unique. This identifier is used to control and observe all other lamp attributes.
2. *Lamp Label* - This is a character string which is assigned by the switching function to describe the feature or service's status associated with this lamp. This attribute cannot be changed by the computing function. The meaning of the Lamp Label attribute is switching function specific.
3. *Lamp Mode* - This is the output of the lamp which is used to indicate the status of the (feature/service) or physical component. The output values are represented by the various ways light can be produced from a lamp. This output can be changed at any time by either the switching or computing function. For more details, refer to definition of the lampMode parameter.
4. *Lamp Brightness* - This attribute indicates the visible brightness of the lamp when it is on. This attribute can be changed by the switching function or by the computing function. For more details, refer to the definition of the lampBrightness parameter.
5. *Lamp Color* - This attribute is an additional characteristic of the lamp which helps distinguish it from other lamps. This attribute can only be changed by the switching function. For more details, refer to the definition of the lampColor parameter.
6. *Button Association* - This identifies a button that is associated with the lamp. The lamp can be used to represent either the status of the feature/service associated with the button or to represent the status of the toggle sequence.

6.1.3.1.5 Ringer

A *ringer* is a component associated with the physical element that provides indication that a device is being rung. There may be one or more ringers associated with a device.

A ringer has attributes that can be controlled and observed by a computing function. The following is a list of those attributes:

1. *Ringer Identifier* - Each ringer that can be observed and/or controlled within the switching function is referenced using an assigned identifier. The ringer identifiers associated with a given physical element's Device Identifier are unique. This identifier is used to control and observe all ringer attributes associated with the device.
2. *Ring Mode* - This attribute describes if the ringer is engaged in a ringing cycle. It will remain *ringing* for the entire ringing cycle (e.g. across consecutive instances of a ringing pattern). Typically only one ringer on a physical element can be rung at one time.
3. *Ring Count* - This attribute describes the number of ring cycles (instances of ring pattern) that the ringer has completed. This attribute is set to 0 immediately before the first ring cycle starts. Note that this is used to query the most recent ring count even after ringing has ceased.
4. *Ring Pattern* - This attribute describes the type of Ring Pattern associated with a ringer. Each individual Ring Pattern cycle may consist of zero or more periods of audible ringing followed by a silent phase. The Ring pattern may be used to help audibly distinguish the types of calls at a device or to uniquely identify a ringer. The meaning of the Ring Pattern is switching function specific.
5. *Ring Volume* - This is the level at which the ringer is set to ring. This information is associated with the ringer until it is reset by the switching function or until it is changed via the Set Ringer Status service.

6.1.3.1.6 Display

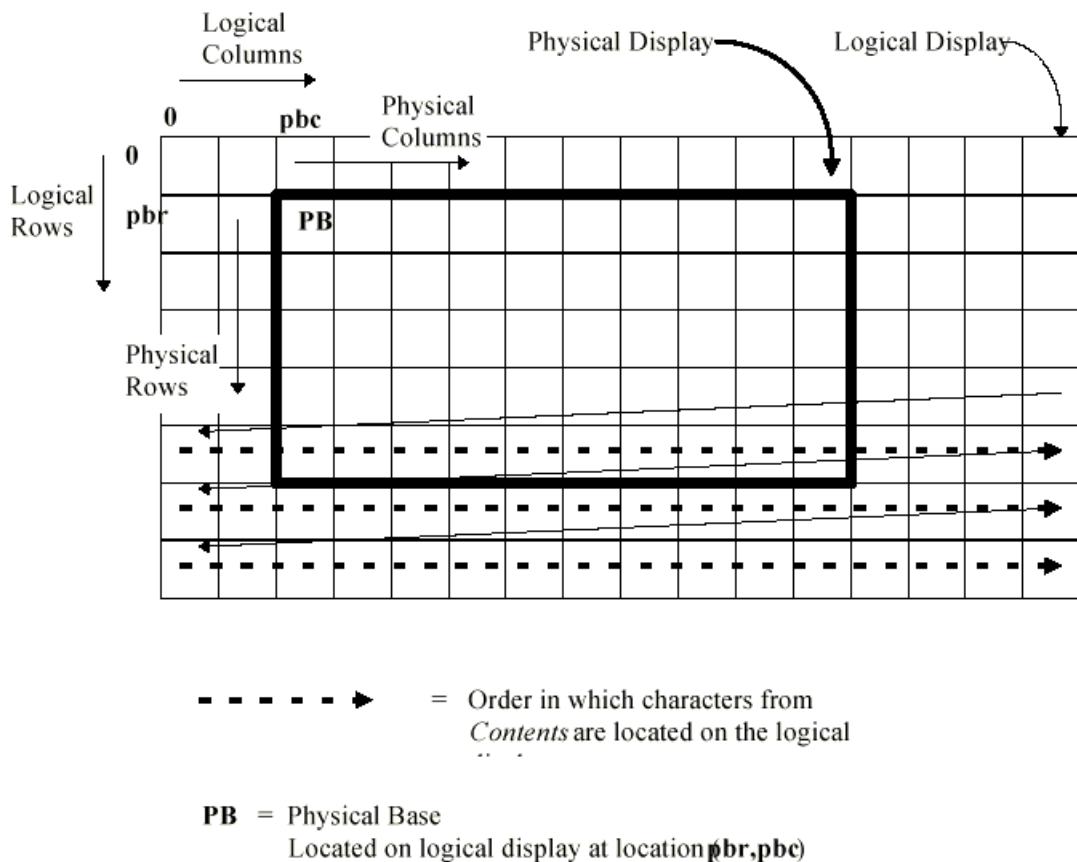
A *display* is a component which presents a two dimensional array of characters associated with the physical element. A physical element can have several displays. A display may be real or virtual; that is it may or may not

actually be present on the physical device itself. Displays have eight attributes as visualized in Figure 6-5, "Display Attributes":

1. *Display Identifier* - To identify a specific display on a physical device.
2. *Logical Rows* - The number of rows on the logical display.
3. *Logical Columns* - The number of columns on the logical display.
4. *Physical Rows* - The number of rows on the physical display. This number is always smaller or equal to *Logical Rows*.
5. *Physical Columns* - The number of columns on the physical display. This number is always smaller or equal to *Logical Columns*.
6. *Physical Base* - The location of the first character of the physical display expressed as (LogicalRowNbr,LogicalColumnNbr) and identified in Figure 6-5 as (**pbr,pbc**). Note that the top-left most position in the logical display is defined as (0,0).
7. *Character Set* - Normally ASCII, but may be also be Unicode or a proprietary character set used by the switching function. This attribute is fixed for a given display.
8. *Contents* - A character string which represents the contents of the logical display. Spaces are always present so the size of this string is always the product of the number of logical rows and logical columns. The *contents* can be observed and/or set (if supported) by the computing function. When setting the value of the *Contents*, an offset (starting point in logical display) and a length field (number of characters to be set) shall be given).

The following figure visualizes these attributes further:

Figure 6-5 Display Attributes



6.1.3.2 Logical Element

A logical element is the part of a device that is used to manage and interact with calls at a device. This element represents the isochronous media stream channels (e.g., ISDN bearer channels) and associated call handling facilities that are used by the device when involved in a call (i.e., via a connection). If a device also has a physical element, the logical element may interact with the physical element's components in order to convey call information (e.g., via lamps) to the user of the device, to provide/manage the media stream data of the call (e.g., via an auditory apparatus) for the user of the device, and to allow the user of the device to manage the calls (e.g., via buttons). The implementation of this interaction is device and switching function specific. The following are the call and call-related features/services that are controlled and observed through the logical element itself:

- Logical Device features/services which are used to indirectly control calls at the device such as Get/Set Forwarding, Get/Set Do Not Disturb and Get/Set Auto Answer.

Note that these features/services also include the associated events and monitoring of these events.

The logical element of a device is observed and/or controlled within the switching function through an assigned Device Identifier.

Note that if the device is a combination of logical and physical elements, the assigned Device Identifier is the same for both elements.

The following sections describe the attributes and components of the logical element. These attributes/component shall be controlled and observed in conjunction with the logical element (i.e., associated with the logical element's Device Identifier).

6.1.3.2.1 Appearance

An appearance is a receptor which is used to connect with at most a single call at the device. A logical element consists of one or more appearances. Appearances are also sometimes called *call appearances*. The number of appearances that a logical element can have is switching function and device specific. Changes in the number of addressable appearances for a logical element are reflected by the capabilities exchange services (13.1 beginning on page 80). Each appearance can be used independently. The following are the call and call-related features/services that are controlled and observed through an appearance for a particular call:

- Call Control features/services such as Make Call, Deflect Call, Answer Call.
- Call Associated features/services such as Associate Data, Generate DTMF, Generate Telephony Tones.
- Routing Services such as Route Request, Route Select and Route End.
- Media Stream Access such as Attach Media Service, and Detach Media Service.
- I/O Services such as Start Data Path and Send Data (when applied to logical elements of a device.)

Note that these features/services also include the associated events and monitoring of these events.

An appearance has several attributes. The following are those attributes:

1. *Addressability* - The addressability of an appearance refers to whether or not the switching function is explicitly representing the appearance to the computing function.
 - a. *Addressable* - An appearance is addressable if it can be explicitly referenced by the computing function at any time with or without the involvement in a call, through a CSTA static device identifier. Refer to Clause 10, "CSTA Device Identifier Formats" for a description of how addressable appearances are referenced.
 - b. *Non-addressable* - An appearance is non-addressable if it can only be referenced, when it is involved with a call, through a CSTA connection identifier. In this case, the logical element dynamically creates and destroys appearances based on the call activity, call capabilities, and features/services of the device. Once the appearance is created (i.e., associated with a call), the corresponding Connection Identifier shall be used to control and observe the appearance. For example, when a call is presented to the device, the logical element creates an appearance to handle the call.

2. *Appearance Type* - The type of appearance is based on its relationship with other devices. The type of appearance determines the functionality and behaviour associated with the logical element of the device. There are two types of appearances: Standard and Bridged Appearances.

Refer to Annex B for a complete description of the types of appearances, their associated behaviour, and Annex C for examples.

6.1.3.2.2 **Group (to be defined)**

6.1.3.3 **Device Configurations**

A *device configuration* describes the arrangement of the various elements and appearances that can be directly associated with a given device. Multiple device configurations may be formed from the possible combinations of physical elements, logical elements, and different appearance types.

Device configurations are described in terms of a specific device configuration for a particular device:

1. *Device's element combination* - This indicates whether the device has a physical element only, a logical element only or both a logical and physical element.
2. *Other devices using the physical element* - This indicates the list of devices (i.e., their logical elements) that are using the physical element of the base device.
3. *Other devices using the logical element* - This indicates the list of devices (i.e., their physical elements) that are using the logical element of the base device.
4. *The logical element's appearance addressability* - This is an attribute of the appearances of the logical element of the base device (if the logical element is present).
5. *The logical element's appearance type* - This is an attribute of the appearances of the logical element of the base device (if the logical element is present).
6. *The number of appearances associated with the logical element* - This is an attribute of the logical element of the base device (if the logical element is present).

As a set, these attributes describe the device configuration for a specific device. The following sections illustrate typical examples of device configurations that can exist in a switching sub-domain.

Note that in the following examples, where physical and logical elements form part of the same device, the application of a suffix number to the identifying letter identifies that they are parts of the same device (e.g. L1, P1 are a single device; L1, P2 are elements from different devices).

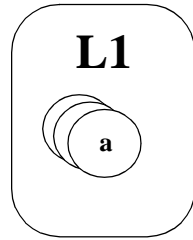
6.1.3.3.1 **Logical Element Only**

This device configuration has only one logical element (e.g., some Park devices). The following identify the attributes of this device configuration:

- *Device's element combination* - logical element only (L1)
- *Other devices using the physical element* - None
- *Other devices using the logical element* - None
- *The logical element's appearance addressability* - Non-addressable
- *The logical element's appearance type* - Selected-standard
- *The number of appearances associated with the logical element* - unlimited (switching function based limits)

Figure 6-6 is a diagram of a logical element only device configuration.

Figure 6-6 Logical Element Only Device Configuration



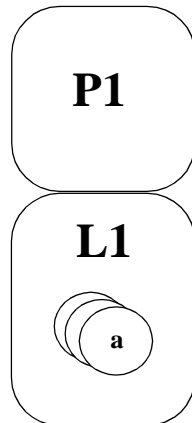
6.1.3.3.2 Single Physical and Logical Element

A *Single Physical and Logical Element* device configuration consists of a single physical element of the device associated with a single logical element of the device that contains non-addressable standard appearances. This device configuration could be used to model a basic telephone station device (e.g., a Plain Old Telephone Service (POTS) telephone or a featured telephone). The following identify the attributes of this device configuration:

- *Device's element combination* - both a logical and physical element (L1/P1)
- *Other devices using the physical element* - None
- *Other devices using the logical element* - None
- *The logical element's appearance addressability* - Non-addressable
- *The logical element's appearance type* - Selected-standard
- *The number of appearances associated with the logical element* - unlimited (switching function based limits)

Figure 6-7 is a diagram of this device configuration.

Figure 6-7 Single Physical and Logical Element Device Configuration (One Device)



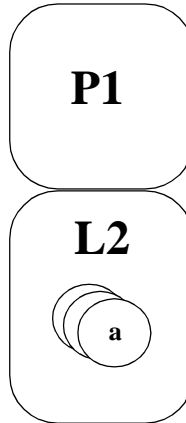
In this figure, the labels "L" and "P" represent the logical and physical elements of the device respectively.

Another variation of the single logical and physical element device configuration involves two different devices - one with a logical element only and one with a physical element only - that are associated with each other. From the perspective of the physical device element P1, the device configuration in this example can be represented as follows:

- *Device's element combination* - physical element only (P1)
- *Other devices using the physical element* - one device (L2)
- *Other devices using the logical element* - None
- *The logical element's appearance addressability* - N/A
- *The logical element's appearance type* - N/A
- *The number of appearances associated with the logical element* - N/A

Figure 6-8 is a diagram of this device configuration.

Figure 6-8 Single Physical and Logical Device Configuration (two devices)



6.1.3.3.3 Multiple Logical Elements

A *multiple logical elements* device configuration consists of a single physical element associated with multiple logical elements containing standard appearances. A multi-line telephone station could be modeled using this device configuration. The following identify the attributes of this device configuration:

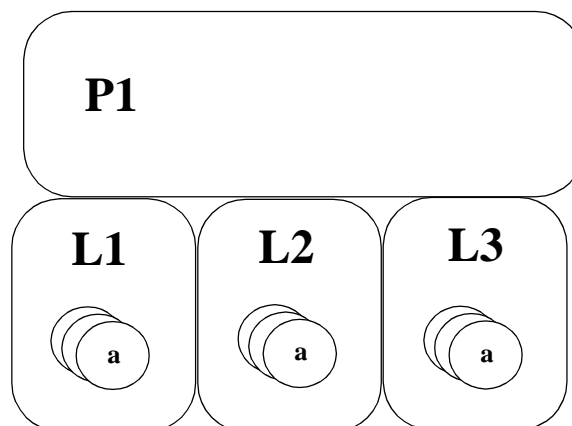
- *Device's element combination* - physical and logical element combination (L1/P1)
- *Other devices using the physical element* - two devices (L2,L3)
- *Other devices using the logical element* - None
- *The logical element's appearance addressability* - non-addressable
- *The logical element's appearance type* - Selected-standard
- *The number of appearances associated with the logical element* - unlimited (switching function based limits)

None of the logical elements in this device configuration need to be part of the same device as the physical element.

Multiple logical elements device configuration represents a single physical element (a telephone set) in a telephone system that supports only one appearance per logical element but has access to multiple calls simultaneously.

Figure 6-9 is a diagram of a multiple logical elements device configuration.

Figure 6-9 Multiple Logical Elements Device Configuration



6.1.3.3.4 Multiple Appearance

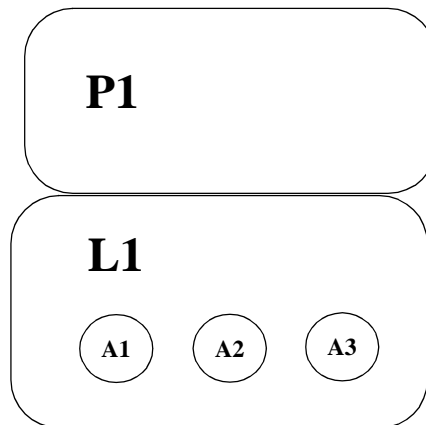
A *multiple appearance* device configuration consists of single physical element and a single logical element containing two or more addressable appearances. Multiple appearance device configurations are another way to represent a single telephone set that has access to multiple calls simultaneously. This approach could be used in a

telephone system that supports addressable standard appearances. This device configuration is sometimes called a *call appearance station*. The following identify the attributes of this device configuration:

- *Device's element combination* - physical and logical element combination (L1/P1)
- *Other devices using the physical element* - None
- *Other devices using the logical element* - None
- *The logical element's appearance addressability* - addressable
- *The logical element's appearance type* - Selected-standard
- *The number of appearances associated with the logical element* - 3 (A1, A2, A3)

Figure 6-10 is a diagram of a Multiple Appearance Device Configuration

Figure 6-10 Multiple Appearance Device Configuration



6.1.3.3.5 Bridged

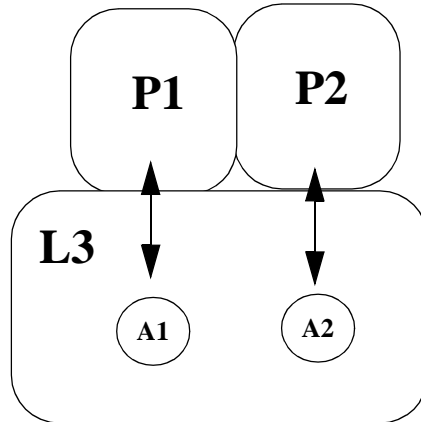
A *bridged* device configuration involves bridged appearances. The characteristics of a bridged device configuration depends upon whether the device configuration is for a physical or logical element.

In the example presented in Figure 6-11, the device configuration shown is for logical element L3 which has bridged appearances. The following identify the attributes of this device configuration:

- *Device's element combination* - logical element only (L3)
- *Other devices using the physical element* - None
- *Other devices using the logical element* - two devices (P1,P2)
- *The logical element's appearance addressability* - addressable
- *The logical element's appearance type* - Independent-shared-bridged
- *The number of appearances associated with the logical element* - 2 (A1 for P1 and A2 for P2)

Figure 6-11 is a diagram of a Bridged Device Configuration

Figure 6-11 Bridged Device Configuration

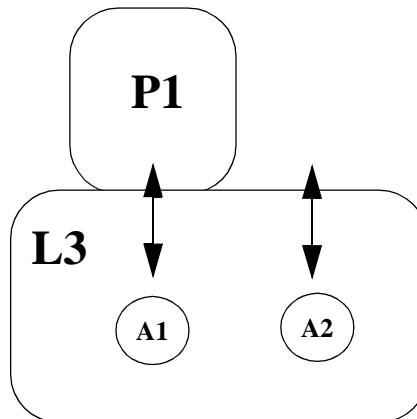


The device configuration for the physical element P1 in this example, is shown in Figure 6-12. In P1's device configuration there are only two device elements rather than three; one is the physical element (P1) and one is the logical element which has two addressable bridged appearances. The following identify the attributes of this device configuration:

- *Device's element combination* - physical element only (P1)
- *Other devices using the physical element* - one device (L3 using appearance A1)
- *Other devices using the logical element* - None
- *The logical element's appearance addressability* - N/A
- *The logical element's appearance type* - N/A
- *The number of appearances associated with the logical element* - N/A

Figure 6-12 is a diagram of a Bridged Device Configuration.

Figure 6-12 Bridged Device Configuration



6.1.3.3.6 Hybrid

A physical element associated with multiple logical elements that each have different types of appearances has a *hybrid* device configuration.

An arbitrary example of a hybrid device configuration is shown in Figure 6-13. This example consists of one physical element and three logical elements.

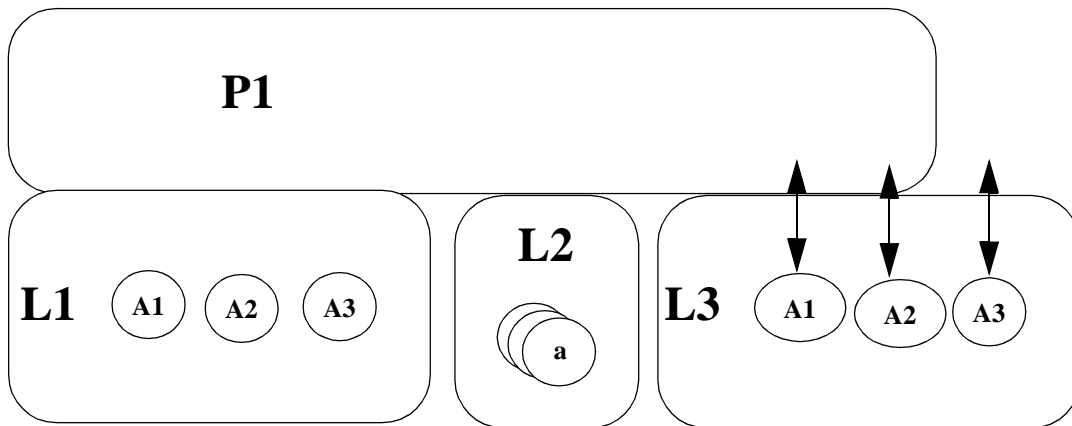
- device 1 has both a physical element (P1) and a logical element (L1) containing two addressable standard appearances
- device 2 has only a logical element (L2) with non-addressable standard appearances
- device 3 has only a logical element (L3) with three addressable bridged appearances

The following identify the attributes of this device configuration:

- *Device's element combination* - logical and physical element (L1/P1)
- *Other devices using the physical element* - two devices (L2, L3 using appearances A1 and A2)
- *Other devices using the logical element* - None
- *The logical element's appearance addressability* - addressable
- *The logical element's appearance type* - Selected-standard
- *The number of appearances associated with the logical element* - 3 (A1, A2, A3)

Figure 6-13 is a diagram of a Hybrid Device Configuration.

Figure 6-13 Hybrid Device Configuration



6.1.3.4 Device Categories

The device category of a particular device provides a generic indication of the device's behaviour and configuration. The computing function should use the device category along with other information provided by the capabilities exchange services to model a given device.

6.1.3.4.1 Station Device Category

This category of device can range from a basic "Plain Old Telephone Set" (POTS) device to a very complex feature telephone device. Station devices can be represented by any single device configuration type, or more commonly, as a hybrid of two or more different device configuration types. The physical component, if present, may have any combination of components. The logical element(s) may have any number of appearances appropriate for the type of device configuration.

6.1.3.4.2 Network Interface Device Category (to be defined)

6.1.3.4.3 ACD Device Category (to be defined)

6.1.3.4.4 ACD Group Device Category (to be defined)

6.1.3.4.5 Hunt Group Device Category (to be defined)

6.1.3.4.6 Park Device Category

A *park* device category is a device which is exclusively used by the switching function to park calls on behalf of other devices in the switching sub-domain. These calls, once parked, may be retrieved later by any device in the switching sub-domain. The number of calls that can be parked at one of these devices is switching function specific. These devices can be represented by either a single physical and logical elements device configuration or a logical element only device configuration. The visibility of these devices within the switching sub-domain is switching function specific.

Note that calls may be parked at devices other than a Park device.

6.1.3.4.7 Pick Group Device Category (to be defined)

6.1.3.4.8 Other Device Category (to be defined)

6.1.3.5 Named Device Types

Switching Function implementations may wish to indicate that a particular device is referred to as being of a particular device type. The following device types may be used for this purpose but the interpretation of a given device type is implementation specific.

- ACD
- ACD Group
- Button
- Button Group
- Conference Bridge
- Line
- Line Group
- Operator
- Operator Group
- Parking Device
- Station
- Station Group
- Trunk
- Trunk Group
- Other
- Other Group

6.1.4 Call

Calls are communication relationships between one or more CSTA Devices. A call's behaviour can be observed and manipulated across the CSTA service boundary (also called service boundary in this Standard). During some call phases (e.g., establishment and release) the call is not completely formed and there may be only a single CSTA Device involved (for example, the CSTA Device on whose behalf the call was initiated). In some call control operations, such as a conference and transfer, one CSTA Device in a call is replaced with another CSTA Device, or two calls are merged into a single call.

The CSTA Call attributes, which are described in detail in the following sub-clauses, shall be:

- Call Identifier
- Correlator Data
- User Data
- Media Call Characteristics
- Account Information
- Authorisation Code
- Charging Information

6.1.4.1 Call Identifier

A CSTA *Call Identifier* (also called *Call Identifier* in this Standard) is a reference associated with a call whereby the call is known to the switching, computing and special resource functions through the call's life. A Call Identifier shall be allocated to each call by the Switching Function, at the latest, when the call first becomes visible across a CSTA Service Boundary. It shall be unique within a switching sub-domain and shall be the same for all CSTA

Devices in the call. A Call Identifier can be assigned to a call before the call is fully established. For example, an incoming call may be assigned a Call Identifier when the called CSTA Device is alerting and before the call has been answered. A Call Identifier shall not only reference the entire call within the sub-domain but shall also infer a reference to that part of the call that is outside the sub-domain.

A call can pass through various stages involving many different CSTA Devices before it finally terminates. Some of these stages cause a call to change identifiers. Examples of services that cause this are Transfer and Conference. During the operation of these services, or as a result of manual intervention, the Call Identifier may change as a result of two calls being merged by the switching function but the call shall continue as a CSTA object. This merger results in the Call Identifiers for both old calls changing to a new identifier for the resulting calls in which the CSTA Devices are involved. The respective Conferenced or Transferred event specifies the transition from the old Call Identifiers to the new Call Identifiers indicating the old invalid Call Identifiers. Management of Call Identifiers is covered in section 6.1.8, “Management of Dynamically-Assigned Identifiers”.

6.1.4.2 Call State

The term *Call State* means the collective set of Connection states for all the Connections comprising a call. Call state is returned only by the Snapshot Device Service for CSTA Devices that have calls. Connection states are further described in 6.1.5, “Connection”, on page 24. Call states are described in more detail in 6.1.6, “Call State Definitions”, on page 27.

6.1.4.3 Correlator Data

Correlator Data is computing function specific data which has been attached to a call that a computing function is controlling or observing. This information, for example, might be a key to a database entry, an application command sequence, file name, etc. Once Correlator Data is associated with a call, it remains with the call for its entire duration (at least one CSTA Device is actively involved in the call), or until the computing function overwrites the data with new data. In order to remove data, the computing function shall overwrite the existing data with null data. Correlator Data enters the switching sub-domain in two ways:

1. A computing function provides Correlator Data on a Call Control service request
2. Correlator Data arrives from an external network connection with a call (for example, Correlator Data may be used as a key to pop a screen for the call). Correlator Data is delivered through an external network via the format described in Annex D.

Permitting a computing function to associate its own information with a call allows multiple computing functions to share data on calls which they are all controlling or observing. This feature is useful when calls are moving from one computing function to another in a distributed computer network or from one switching sub-domain to another. For more information on Correlator Data, refer to 12.2.9, “CorrelatorData”, on page 63.

Correlator Data is provided by the Computing Function and associated with the call for its entire duration or until overwritten with new data. This data survives Conference and Transfer and can be provided on various events. An application may remove the Correlator Data by overwriting existing data with null data.

When Correlator Data is associated with a call, call events that indicate that a CSTA Device has become part of a call (such as Delivered, Established and Queued events, for example) shall include the Correlator Data (if this parameter is supported in the event being reported). Subsequent call events also may contain the Correlator Data.

Note that “null Correlator Data” means Correlator Data information with zero length.

When a consultation call is transferred or conferenced and null or non-null Correlator Data is associated with either (or both) the primary or secondary call, the Correlator Data in the resulting call shall always be the same Correlator Data that was associated with the secondary call (even if only the primary call had non-null Correlator Data). In that case the Correlator Data (if any) associated with the primary call is discarded. If the secondary call contains no

Correlator Data, the Correlator Data associated with the resulting call is that which was associated with the primary call.

Table 6-1 Inheritance rules for Correlator Data in Conference and Transfer

Secondary Call	Primary Call		
	No Correlator Data	Null Correlator Data	Correlator Data 1
No Correlator Data	No Correlator Data	Null Correlator Data	Correlator Data 1
Null Correlator Data	Null Correlator Data	Null Correlator Data	Null Correlator Data
Correlator Data 2	Correlator Data 2	Correlator Data 2	Correlator Data 2

When Correlator Data is associated with a call via the Associate Data service, the Call Information event is provided by the switching function. If the data is changed by any other service, the switching function does not provide the Call Information event.

6.1.4.4 User Data

User Data is call-related computing function-to-computing function information that, unlike Correlator Data, is not associated with a call for the life of the call. The switching function receives User Data in two ways:

- A computing function sends User Data in a service request.
- User Data arrives from an external network connection with a call (for example, User Data may be used as a key to pop a screen for the call).

Both a computing function and the network may send User Data in two ways:

1. *With Call Control Activity* - Call control service requests (and network signalling for call control) permit User Data as an optional parameter. The switching function reflects the delivery of User Data in the first call control event that results from the switching function or network carrying out the call control activity. When the computing function provides User Data in a Call Control service request, the User Data is delivered to other parties if and only if the call control service successfully completes. If the switching function does not generate the call control event that corresponds to the call control activity because the computing function has set an event filter that filters out the relevant event, then the User Data is not propagated to subsequent events and the User Data information will be lost. Refer to the description of the individual Call Control services for more details on the events that will contain User Data for those services.
2. *Independent of Call Control Activity* - The computing function may use the Send User Data service to pass User Data at any time. Some networks provide an independent signalling mechanism for sending User Data. The switching function generates a Call Information event with the userData parameter containing the received User Data to reflect the delivery of the User Data. Independent of call control activity, this event is generated for all computing functions monitoring the call and all computing functions monitoring any CSTA Devices that have a connection to that call.

This standard defines a mechanism for delivering both User Data and Correlator Data through an external network at the same time. This mechanism is described in Annex D.

When a computing function uses a service to send User Data, the switching function sends that User Data only after the switching function sends a positive acknowledgement to the service request.

User Data is described further in 12.2.19, “UserData”, on page 69.

6.1.4.5 Other Call Related Information

There is additional information associated with calls such as:

- *Account Information* - A computing function or business-specific piece of information that is assigned to a call for accounting purposes. For more information, see 12.2.1, “AccountInfo”, on page 59.
- *Authorisation Code* - A code provided to the switching function that is used to check if a computing function user is authorised to perform a given service. For more information see 12.2.3, “AuthCode”, on page 60.

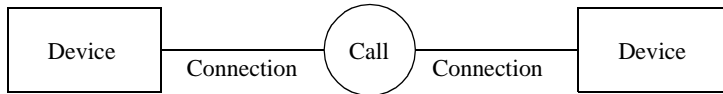
- *Charging Information* - An amount charged to a device for a call in which the device was involved. For more information, see 12.2.6, “ChargingInfo”, on page 61.

6.1.4.6 Media Call Characteristics (to be defined)

6.1.5 Connection

A *connection* is a relationship in a switching sub-domain between a CSTA Device, and a call in which that CSTA Device is involved. This connection relationship can be both observed and manipulated. Figure 6-14 illustrates the relationship between calls, devices, and connections.

Figure 6-14 Relationship between Calls, Devices, and Connections



Observation and manipulation of these connections are the basis for call control services (such as Clear Connection, Answer Call, etc.). Connections are CSTA Objects that have the following attributes:

1. *Connection Identifier* - Each connection that can be observed and/or controlled shall be referenced across the service boundary. To accomplish this, each connection is assigned a unique identifier by the switching function. This identifier is comprised of a Device Identifier and a Call Identifier. For a call, there are as many Connection Identifiers as there are associated devices, and for a device there are as many Connection Identifiers as there are associated calls. The Connection Identifier is unique within a sub-domain and over a single service boundary. It is provided by the switching function when either a new call is created or a new device becomes involved in a call. A Connection Identifier can change as a result of some operations (e.g., a transfer or conference) and in these cases the switching function presents the computing function with the appropriate information to transit from the old identifiers to the new. The Device Identifier used in the Connection Identifier may be either static or dynamically-assigned by the switching function.

As provided by the switching function to the computing function, a Connection Identifier will always include both a Device Identifier and a Call Identifier (unless otherwise noted in the specification of a particular CSTA event’s parameters). Computing functions wanting to correlate event reports which associate devices connected together in a call can use the Call Identifier to do this correlation. The definitions of a Connection Identifier and those identifiers that it comprises (Call and Device Identifiers) restrict computing functions from fabricating Connection Identifiers.

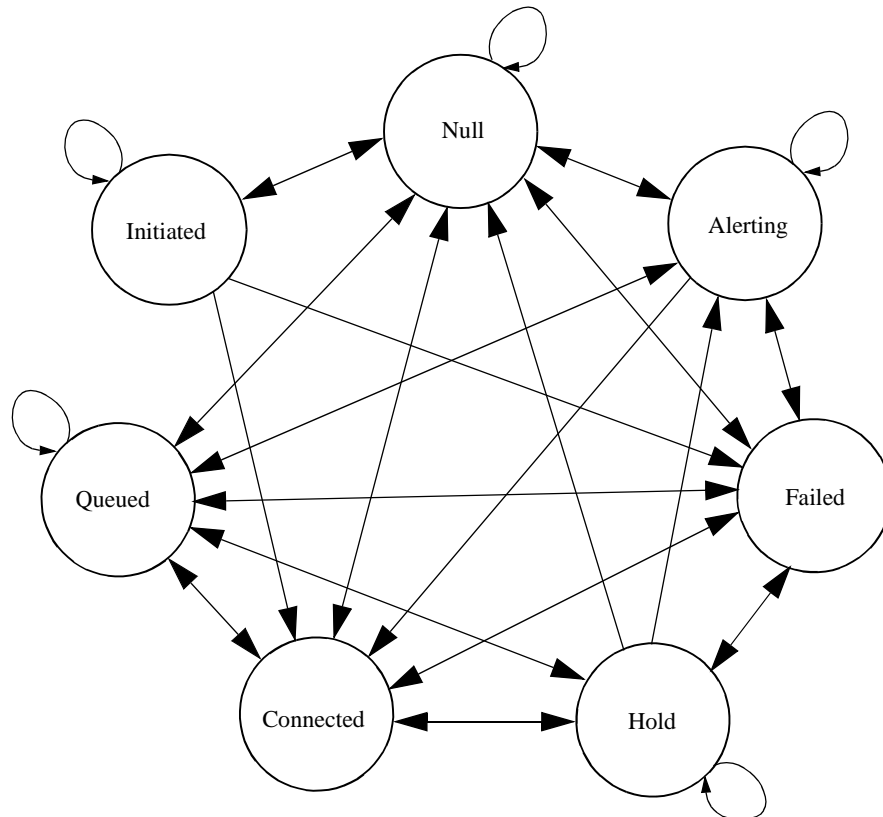
As provided by the computing function to the switching function, a Connection Identifier shall be one which was originally provided by the switching function. An exception to this rule is where either a deviceID only or a callID only Connection Identifier is used in a specific service (as indicated by the capability exchange services). If a Connection Identifier, provided by the computing function, includes only a Device Identifier, then that Device Identifier shall be a static Device Identifier. These conditions ensure that it is possible to use only a Device Identifier (without a Call Identifier) or a Call Identifier (without a Device Identifier) to provide a Connection Identifier in certain specified circumstances.

For additional details regarding Connection Identifiers, including Connection Identifier formats and specific functional requirements, see 12.3.8, “ConnectionID”, on page 74 of this Standard.

2. *Media Stream Flow Direction* - This attribute is the direction or directions in which the media stream can be transmitted on the given connection. The following are the types of directions that can be associated with a connection:
 - *Transmit* - Media stream data can only be transmitted on the connection by the associated device.
 - *Receive* - Media stream data can only be received on the connection by the associated device.
 - *Transmit and Receive* -Media stream data can be transmitted and received on the connection by the associated device.

3. *Media Stream Channels* - A channel is a path of communications between devices within a network. Channels are created to transmit/receive the media stream when the associated connection is created for the device in the call. The correlation of a channel to an actual media stream communications path/channel within the switching function is switching function specific. The switching function may represent a channel as a group of actual media stream communication paths. A device's connection represents a channel or set of channels on which the media stream associated with the call is to be transmitted and received. The number of channels per connection is switching function and device specific (the capability exchange services may be used to determine the value). A digital data connection can use one or more channels. In addition, there can be multiple media stream types associated with a given connection as well as the associated channels. The attachment of media services is to a connection and its associated channels as a whole. The switching function is then responsible for attaching the Media services to the appropriate channels.
4. *Connection State* - A connection state involves a single call/device relationship. When a call is present at a device, the connection representing that call at that device will transit through various stages. State transitions are observed by the computing function through event reports. The transition from one state to the next is caused by either a manual user stimulus or a CSTA service initiated across the service boundary. Connection states may also be reported by Snapshots on either calls or devices.

Figure 6-15 Connection State Model



The following are the connection state definitions:

- *Alerting* - A state in which an attempt is being made to connect a call to a device. There are three distinct modes where a connection may be in the alerting state:
 - *Offered* - In this mode, the call is in a pre-delivery state at the target device. The opportunity exists for a computing function to issue one of a set of supported services (e.g., Accept Call, Clear Connection (“reject”), Deflect Call) or an ISDN device to accept or reject the call. From the calling side perspective, the call is not delivered at the called device. As a consequence, delivery information such as Ringback indication and/or Network signalling is not provided. For example, the device makes no ringing sounds while in the Offered mode of the Alerting state. The Offered mode is indicated by an Offered event.

- *Ring*ing - In this mode, the call is being presented for the purpose of having the device connect to the call and the user is made aware that the call is being delivered at the device. The Ringing mode is indicated by a Delivered event with a cause code other than “Entering Distribution”. The actual device activity to notify the user (e.g., ringing) is reported through the physical device feature events.
- *Entering Distribution* - In this mode, a call is being delivered to a distribution device. The Entering Distribution mode is indicated by a Delivered event with a cause code of “Entering Distribution”.
- *Connected* - A state in which a device is actively participating in a call. This state includes logical participation in a call as well as physical participation.
- *Failed* - A state in which call progression has been aborted. This state generally results when a device tries to become Connected to a call or a call tries to become Connected to a device and the attempt fails. The Failed state can result from failure to connect the calling device and call, failure to connect the called device and call, failure to create the call, failure when the call ends and other reasons. Refer to 6.7.2, “Connection Failure”, on page 38 for more information.
- *Hold* - A state in which a device is inactively participating in a call. This state includes logical participation in a call while physical participation is suspended.
- *Initiated* - A state in which a device is requesting a service or in the process of dialling the necessary digit sequence to initiate a call to another device. The connection enters this state when the device goes off-hook (e.g., receiving dialtone) or the device is being prompted to go off-hook as a result of some service being initiated for the device.
- *Null* - A state in which there is no relationship between a call and device.
- *Queued* - A state in which call progression is suspended or made inactive while awaiting some form of action. Examples of situations in which a connection might transit to the Queued state include (among others) the following:
 - A call is parked at a device.
 - A call is queued at a distribution mechanism, waiting for an agent to become available.
 - A call is camped on to a device.
 - An appearance of a shared bridged device configuration is inactive with respect to the call.

Table A-1 on page 290 provides an example to illustrate each transition which is illustrated in Figure 6-15, “Connection State Model.”¹

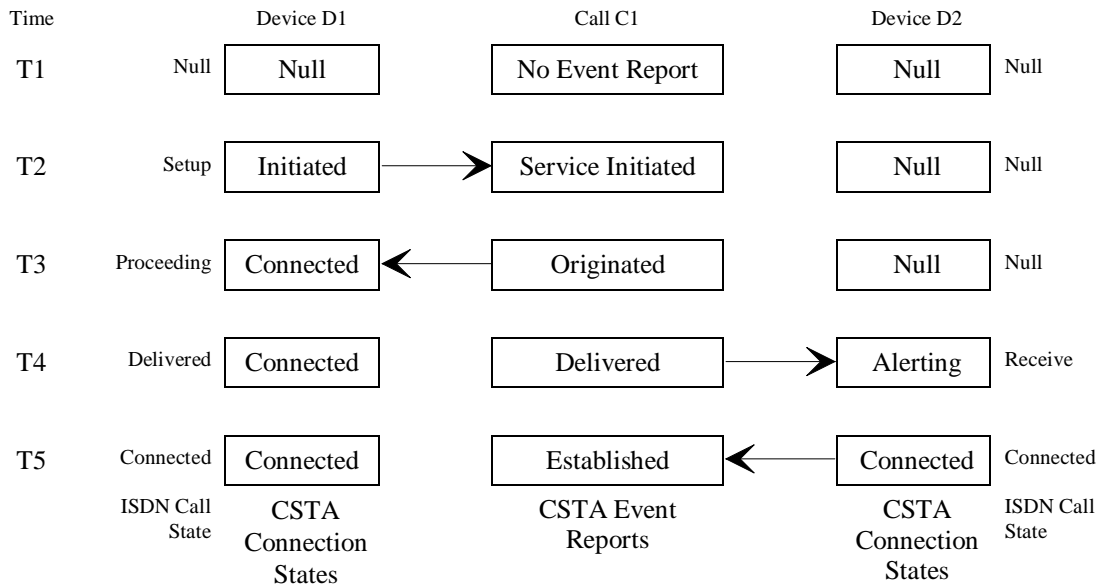
6.1.5.1 Call Event Reports

The Connection state model provides an abstract view of actual states and events that are communicated via underlying signalling systems. This abstract view is introduced to provide a language for describing CSTA Event Reports, states and Functional descriptions. Because of the topology of the Switching Function, the signals that report events and state changes have definite sources. Providing a telecommunications object (the Connection) that can be associated with the source of these signals helps when explaining the meaning of events and the operation of CSTA (and other) telecommunications services.

Note that on a typical ISDN access to a network there exists a distributed state machine. One part of this distributed state machine resides in the ISDN device. Another part resides on the other side of the ISDN access. There is another similar distributed-state access machine that resides across the ISDN network at a similar device. Using this concept, a call can be modeled as a collection of Connection state machines communicating with one another using signalling. When this communication occurs, a CSTA Event Report can be generated. In the following figure, this concept of communication between two state machines is illustrated for the case of establishing a simple call. Additionally, on each side of the figure the ISDN call states are indicated.

1. For an explanation of the Initial and Final State diagrams’ nomenclature used the Table A-1, refer to Clause 11, “Template Descriptions”, beginning on page 54.

Figure 6-16 Relationship of CSTA Call Event Reports



Notice in Figure 6-16 that the CSTA Event Reports are based on signalling interactions of the Switching Function. Many Connection events are of interest to CSTA applications. Typically, however, a CSTA application is interested in atomic telecommunications activities and these often involve many simultaneous Connection events. Generally, telecommunications operations embody changes to many Connections. These events can be summarized in a single Event Report. For instance, the Transfer, Conference and Clear Call Services all make changes to multiple Connections but are each represented by single Event Reports. The Connection state changes associated with each CSTA Event Report are defined in this Standard.

6.1.6 Call State Definitions

The state of a CSTA Call can be precisely expressed as the list of Connection states of all the devices involved in the call. This list is called the Compound Call State. The technique of listing the Connection states to describe the Call state can describe any call state that is possible in CSTA. However, most calls involve a small number of widely recognized states. CSTA defines those states in terms of their set of Connection states, but communicates them as atomic Call states - not as a list. These widely recognized states are called the Simple Call States.

For calls with one known Connection state, the single Connection state shall be provided as a Call state.

Note that since Null can be a known Connection state, for a nascent call it is possible to have a CSTA Call state with only one non-Null Connection (see Table 6-2).

For calls with more than two non-Null Connection states, the list of Connection states is provided as the call's state.

CSTA simplifies Call states by relating them (at times) to particular devices. These relationships are described by differentiating the call's Connection states. The Connection state associated with a particular device is called the local Connection state (for that device). Other Connection states are not differentiated from one another. Thus, CSTA Call state is defined for a device by the combination of Connection states as well as the order in which the Connection states are combined. For example, the Alerting-Connected Call state is not the same as Connected-Alerting. The first is defined as Received and the second is defined as Delivered. For calls with exactly two Connections, the CSTA Call state assigned to the combinations of Connection states are summarized in the following table. If there is no Simple Call state for a particular combination of Connection states, then a Compound

state shall be provided as the Call state. For Compound Call states, the first Connection state in the list shall be the local Connection state.

Table 6-2 Definition of CSTA Simple Call States

Local Connection State	Other Connection State	CSTA Simple Call State
Alerting	Connected	Received
Alerting	Hold	Received-On Hold
Connected	Alerting	Delivered
Connected	Connected	Established
Connected	Failed	Failed
Connected	Hold	Established-On Hold
Connected	Null	Originated/Terminated
Connected	Queued	Queued
Failed	Null	Blocked
Hold	Alerting	Delivered-Held
Hold	Connected	Established-Held
Hold	Failed	Failed-Held
Hold	Queued	Queued-Held
Initiated	Null	Pending
Null	Null	Null

NOTE

The Originated / Terminated state may occur both during call set-up and when the call ends. When a far-end party drops from a two-party call and the near-end end-point is not returned immediately to idle, then the Originated / Terminated state is entered for call tear-down. It is also possible to enter a blocked state when a call ends.

6.1.7 Referencing Devices, Elements, Appearances and Device Configurations

In services and events, devices, elements, appearances and device configurations are referenced using Device Identifiers or Connection Identifiers when a call is present at the device, element, or appearance. Table 6-3 indicates how these Device Identifier references are interpreted. The Connection Identifier references are described in 6.1.5, "Connection", and when a Connection Identifier is used it will refer to the specific device, element or appearance associated with the given call. The following symbols are used in the table:

- Logical** This indicates that the Device Identifier passed will be interpreted as reference to a specific logical element
- Physical** This indicates that the Device Identifier passed will be interpreted as reference to a specific physical element
- Device** This indicates that the Device Identifier passed will be interpreted as reference to the entire device configuration.
- Appearance** This indicates that the Device Identifier passed will be interpreted as reference to a specific addressable appearance of a logical element. In order for computing function to determine what type of referencing is supported for a given logical element, it shall use the capability exchange services (13.1 beginning on page 80).

Refer to 10.1, “Device Identifier Formats”, for the format of Device Identifiers.

Table 6-3 Device Identifier Interpretation

Service/ Event Categories	Identifier Represents	Additional Information
Call Control, Call Associated, Media Service and Routeing Services	Device	The device (configuration) itself selects which appearance is to be used.
	Appearance	The Device Identifier selects the specific appearance which is to be targeted by the service.
Call Control, Call Associated and Media Service Events, as well as switching function to computing function	Device	The Device Identifiers that are associated with an appearance identify only the associated device configuration rather than a specific appearance. Note that this information relates to the content of the event parameters, not what is supplied on the Monitor Start service.
	Appearance	The Device Identifier selects the specific appearances which are being reported in the event. Note that this information relates to the content of the event parameters, not what is supplied on the Monitor Start service
Logical Device Services	Logical	The Device Identifier refers to the particular logical element.
Logical Device Events	Logical	The content of the event parameters indicates the given logical element being used. Not that this information relates to the content of the event parameters, not what is supplied on the Monitor Start service.
Physical Device Services	Physical	The Device Identifier shall refer to the particular physical element.
Physical Device Events	Physical	The content of the event parameters indicates the given physical element being used. Note that this information relates to the content of the event parameters, not what is supplied on the Monitor Start service.
Device Maintenance Events	Device	The event parameter contains the Device Identifier of the device configuration.
Monitor Start Service (Call Control/ Associated, and Media Service events)	Device or Logical	The Device Identifier of the device configuration (Device) results in the observation of the entire configuration. The Device Identifier of the particular logical element (Logical) results in the observation of the entire logical element. Note that the event filter determines the types (logical or device) of the events required. The Switching Function interprets the Device Identifier to meet the requirements of the filter. This may result in the same Device Identifier being interpreted as both Device and Logical for different event categories.
	Appearance	The use of the Device Identifier results in the observation of the specific appearance.
Monitor Start Service (Logical Device events)	Logical	The use of the Device Identifier results in the observation of the particular logical element.
Monitor Start Service (Physical Device events)	Physical	The use of the Device Identifier results in the observation of the particular physical element.
<p>Logical = logical element’s Device Identifier</p> <p>Physical = physical element’s Device Identifier</p> <p>Appearance = addressable appearance (can be recognised from other forms of Device Identifiers by the suffix)</p> <p>Device=a Device Configuration formed by multiple devices, which is referenced by the Device Identifier</p>		

Table 6-3 Device Identifier Interpretation (continued)

Service/ Event Categories	Identifier Represents	Additional Information
Monitor Start Service (Device Maintenance events)	Device	The use of the Device Identifier results in the observation of the device configuration.
Snapshot Services	Logical	The Device Identifier refers to the particular logical element.
Logical = logical element's Device Identifier Physical = physical element's Device Identifier Appearance = addressable appearance (can be recognised from other forms of Device Identifiers by the suffix) Device=a Device Configuration formed by multiple devices, which is referenced by the Device Identifier		

6.1.8 Management of Dynamically-Assigned Identifiers

Management of dynamically-assigned Device Identifiers and Call Identifiers is provided through management of Connection Identifiers. This ensures that an identifier whose meaning is dependent on another identifier is always provided in the proper context (i.e., with the other identifier needed to resolve its meaning). For example if a Call Identifier is given relative to a device, then giving the Connection Identifier ensures that the Call Identifier is provided with its reference - the Device Identifier. Management of Connection Identifiers shall be provided as follows.

Connection Identifiers shall be provided when either a new Call is created or a new device becomes involved in a call. When a call is made a Connection Identifier shall be provided. A Connection Identifier shall be provided in Event Reports that pertain to a call. When a device becomes involved in a call, the Connection Identifier shall be provided in the Event Reports that occur at that device.

If a call changes its Call Identifier when a Conference or Transfer occurs, Connection Identifiers shall be provided to link the old Call Identifier to the new Call Identifier. Similarly, if a Device Identifier is changed, new Connection Identifiers shall be provided for the devices in the call.

Management of identifiers shall be provided via parameters included in Service acknowledgements and Event Reports.

Identifiers shall cease to be valid when their context vanishes. If a call ends, its Call Identifier is no longer valid to refer to that call. Similarly, if a device is removed from service or from a call, its dynamically-assigned Device Identifier shall become invalid.

Identifiers can be reused. Once an identifier has lost its context it may be re-used to identify another object. It is recommended that implementations not re-use identifiers immediately.

Individual Call and Device Identifiers are not guaranteed to be globally unique. CSTA requires that the combination of Call and Device Identifier be unique within a CSTA switching sub-domain. To accomplish this, either the Call Identifier, or the Device Identifier (or both) shall be unique. In many cases the Connection Identifier requires both the Call and Device Identifiers to uniquely refer to Connections in a call.

6.2 Special Resource Functions (to be defined)

6.2.1 Voice Unit (to be defined)

6.2.2 I/O Services (to be defined)

6.3 Call Detail Recording (CDR) Services (to be defined)

6.4 Capabilities Exchange (to be defined)

6.4.1 Switching Function Capabilities (to be defined)

6.4.2 Device Capabilities (to be defined)

6.4.3 Dynamic Feature Availability

A computing function can determine all possible CSTA services that can be applied to a connection given its state by using static information obtained through the Capability Exchange services. However, in certain implementations, there are situations where the set of services that can be applied to a connection varies depending upon how the connection got to a certain connection state and/or certain features active at a given device. In these cases, the static information provided in the Capability Exchange services may not reflect the actual set of services that are allowed.

If the Dynamic Feature Availability option is supported (as indicated through the Capability Exchange services), the actual set of CSTA services that can be applied to a connection at a given point is provided through the servicesPermitted parameter in every appropriate event.

Refer to 12.2.16, “ServicesPermitted”, on page 67 for a description on the use and restrictions of this parameter.

6.5 Switching Function Information Synchronization (to be defined)

6.5.1 Switching Function Level Information (to be defined)

6.5.2 Device Level Information (to be defined)

6.5.3 Call Level Information (to be defined)

6.6 Status Reporting Services

Note that this section describes the Status Reporting services between the Switching Function and the Computing Function.

6.6.1 System Status

System Status services provide a way for the computing function and switching function to exchange information about the overall status of the system within each function. For each service boundary in a CSTA environment, the computing function and switching function on each side maintain a status attribute. System status services are bi-directional, enabling the computing function to report its status to the switching function, or to request the status of the switching function, and vice-versa.

6.6.1.1 System Status Registration

Before the computing function can receive any system status service requests, it may be required to register with the switching function for system status services using the System Status Register service. The positive acknowledgement to this service contains the system status register identifier (sysStatRegisterID) that the computing function uses to identify service requests that arrive for this registration.

If the switching function supports the System Status Registration services, then the computing function shall use the System Status Register service to register for system status services before it can receive any system status service requests. The first (mandatory) System Status service request from the switching function issued during an initialization sequence is an exception to this rule, however. If the switching function does not support the System Status Registration services, then the computing function may receive system status service requests at any time. The capabilities exchange services can be used to determine if the switching function supports the System Status Registration services.

The type of system status service requests that apply to the registration can be chosen by the computing function when it issues the System Status Register service request. A status filter can also be specified such that only the status's of interest to the computing function will be reported by the switching function (i.e., if the bit for a status is set in the status filter, then that status is not reported). This filter can be changed using the Change System Status Filter at any time while the registration is active.

A system status registration can be cancelled using the System Status Register Cancel service. Once the switching function sends a positive acknowledgement to this service, it will no longer send system status service requests to the computing function. Additionally, the switching function can cancel a system status registration at any time by sending the computing function a System Status Register Abort service request.

While the system status services themselves are bi-directional, the System Status Registration services are not. These services are only issued by the computing function. The switching function does not register with the

computing function for system status services. The switching function is considered to be (implicitly) registered to receive system status service requests from the computing function at any time.

6.6.1.2 System Status Services

There are two System Status Services: System Status and Request System Status. The first service is used by the requesting function to report its status to the function receiving the service request. The second service is used by the requesting function to request (i.e., query) the status of the responding function.

The computing function can determine if the switching function uses the System Status service for periodic status reporting (i.e., heartbeats) using the capabilities exchange services. The Get Switching Function Capabilities service positive acknowledgement defines a parameter (systemStatusTimer) that is used to indicate whether periodic status reporting is used and if so, how often the computing function should expect the reports. The recovery action to be taken by the computing function in the event of a loss of heartbeats is implementation specific.

All System Status services use the following values to indicate system status:

- *Initializing* - The system is re-initializing or restarting. This status indicates that the system is temporarily unable to respond to any service requests. If provided, this status message shall be followed by an Enable status message that indicates that the initialization process is completed.
- *Enabled* - Request and responses are enabled, usually after a disruption or restart. This status indication shall be sent after an Initializing status indicator has been sent and may be sent under other conditions. This status indicates that there are no outstanding monitor requests.
- *Normal* - This value can be sent at any time to indicate that the status is normal. This status has no effect on other services.
- *Messages Lost* - This status indicates that a service request, response, or event report may have been lost.
- *Disabled* - This cause value indicates that active Monitor Start monitor requests have been disabled. Other requests and responses may also be disabled, but, unlike monitors, reject responses are provided for those.
- *Partially Disabled* - Some of the objects in the system can not be reached. Existing monitors on these objects will not provide events and computer requests targeting these objects will be rejected. This cause indicates to the receiving function that a degradation of service level may occur but not complete system disability. Automatic or manual actions may be taken to remedy the parts disabled.
- *Overload Imminent* - The system is about to reach an overload condition. The client should shed load to remedy the situation.
- *Overload Reached* - The system has reached an overload condition and may take action to shed load. The server may then take action to decrease message traffic. This may include stopping existing monitors or rejecting any new requests sent by the client.
- *Overload Relieved* - The system has determined that the overload condition has passed and normal application operation may resume.

Each system status service request may contain a system status registration identifier (sysStatRegisterID) to identify the associated system status registration (when system status registration is supported by the switching function). A system status service request from the computing function should never contain a system status registration identifier.

6.6.2 Monitoring

To track call control and other activity, and to receive notification of all changes in the switching Function, the computing function uses a feature called monitoring. By starting a monitor, the computing function indicates that it wants to be notified of specific changes that occur in the objects (call and device) and device attributes of a switching function. Examples include:

- Notification that a call has arrived at a device.
- Notification that a call has been answered.
- Notification that a feature such as “Forwarding” or “Do Not Disturb” has changed at a device.

Once a monitor is established, the switching function notifies the computing function of relevant activity by sending messages called *event reports*, or simply *events*.

For example, in the area of Call Control, events report the state transitions through which connections pass. In this way a computing function is able to determine what services are applicable to a given connection. For example, the Delivered event indicates when a connection state transits to the Alerting state.

The event categories are as follows:

- *Call Control* - These events report changes to information related to calls.
- *Call Associated* - These events report changes to information related to calls.
- *Media Stream* - These events report changes associated with attachment of a call to a media device.
- *Physical Device* - These events report changes to the components of a device's physical element.
- *Logical Device* - These events report changes to feature settings associated with a device's logical element(s).
- *Voice Unit* - These events report changes to Voice Unit messages.
- *Maintenance* - These events report changes regarding maintenance.
- *Private* - These events are switching function specific.

6.6.2.1 Starting and Stopping a Monitor

The Monitor Start service is used to establish a monitor. The computing function indicates the monitor object that it is interested in observing, the type of monitoring, the type of calls to monitor, and the list of events that it is interested in.

Once the Monitor Start service request has been validated by the switching function, the switching function provides a positive acknowledgement that includes a Monitor Cross Reference Identifier that uniquely identifies the monitor. The switching function also provides this identifier as a parameter in all events associated with this monitor. The computing function can use this identifier to correlate events to the particular Monitor Start service that established the monitor. (This identifier is also used in the Monitor Stop and Change Monitor Filter services.)

The Monitor Stop service is used to stop an established monitor. When a Monitor Stop service has been sent by the computing function, the switching function stops the monitor, releases the Monitor Cross Reference Identifier, and no longer provides events to the computing function.

The Monitor Stop service may also be sent from the switching function to the computing function when the switching function stops an existing monitor. This occurs when the monitor object is a call-object (Table 6-4), or when the switching function shall terminate a monitor due to load conditions, for example.

Refer to 15.1 beginning on page 94 for a complete description of the Monitor Start and Monitor Stop services.

6.6.2.2 Monitor Objects

The computing function indicates what it wants to monitor by specifying a monitor object parameter in the Monitor Start service request. There are two possible monitor objects: *call-object* and *device-object*.

The following table describes the monitor objects.

Table 6-4 Monitor Objects

Monitor Object	Description
call-object	Place a monitor on an existing call/connection. Only the specific call is monitored. A Monitor Stop service is sent by the switching function to indicate when the existing call is no longer monitored.
device-object	Place a monitor at the specified device.

6.6.2.3 Monitor Types

The computing function also indicates a monitor type when starting a monitor. There are two types of monitoring: *call-type* and *device-type*.

The following table describes the possible monitor types and their meanings.

Table 6-5 Monitor Types

Monitor Type	Description
call-type	<p>The call continues to be monitored as long as it remains in the switching sub-domain.</p> <p>For example, if a call that is being call-type monitored is transferred to another device in the switching sub-domain, the call will continue to be monitored. The computing function receives events for all devices in the call until the call ceases to exist or until it leaves the switching sub-domain. The Diverted event is an exception. The switching function (as indicated through the capabilities exchange services) may or may not be providing Diverted events to all devices in a call.</p> <p>For call-type monitors:</p> <ul style="list-style-type: none"> • When a device ceases to participate in a call, and the call is transferred or forwarded to another device, subsequent events at the new device are reported. The Monitor Cross Reference Identifier used in events at the new device will be the same one used before the call was forwarded or transferred. • If a call is being monitored using a call-type monitor and one of the devices consults to another device (i.e. a new call is created), then the computing function will not see events for the secondary call (new consultation call) until either the primary call is transferred to the consulted device, or until the two calls are conferenced together. • A call that is being monitored may have a new Call Identifier assigned to it after a conference or transfer. The switching function reports the new Call Identifier in a Conferenced or Transferred event.
device-type	The call does <i>not</i> continue to be monitored after the call leaves the device.

6.6.2.4 Relationship of Monitor Objects and Monitor Types

Monitor objects and monitor types are independent. A *monitor object* describes what the monitor is being placed on, while the *monitor type* describes if a call continues to be monitored after it leaves a device.

The following table describes the possible combinations of monitor objects and monitor types and what the resulting combinations represent.

Table 6-6 Monitor Object/Monitor Type Combination

Monitor Object	Monitor Type	Usage
call-object	call-type	<p>This combination is used to track an existing call, for as long as that call remains in the switching sub-domain.</p> <p>Monitor Stop service is sent by the switching function when the call ceases to exist in the switching sub-domain to indicate that the monitor is stopped and the associated Monitor Cross Reference Identifier is no longer valid.</p>
call-object	device-type	<p>This combination is used to track an existing call, while that call remains at the specified device.</p> <p>Monitor Stop service is sent by the switching function when the call leaves the device to indicate that the monitor is stopped and the associated Monitor Cross Reference Identifier is no longer valid.</p>
device-object	call-type	<p>This combination is used to track all calls that arrive (or are present) at the device, for as long as the calls remain in the switching sub-domain.</p> <p>The specified device object can be thought of as a trigger device where all calls that become involved with this device become monitored as long as the call remains in the switching sub-domain.</p> <p>Monitor Stop service is <i>not</i> sent by the switching function when a call ceases to exist or moves away from the monitored device, since the monitor is still in place at the device.</p>
device-object	device-type	<p>This combination is used to track all calls that arrive (or are present) at the device, for as long as the calls remains at the device.</p> <p>Monitor Stop service is <i>not</i> sent by the switching function when a call ceases to exist or moves away from the monitored device, since the monitor is still in place at the device.</p>

6.6.2.5 Monitoring in Relationship with Media Class

The computing function can also indicate the media class (voice, digital data, etc.) of calls to be monitored when starting a monitor.

Refer to the media class component of 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete set of possible values. The media class is independent of the monitor object and monitor type.

6.6.2.6 Reporting Connection State Changes

Once a call is monitored (irrespective of monitor type or monitor object), all connection state changes that are known by the switching function for that call are reported to the computing function (subject to the Monitor Filter—refer to 6.6.2.7).

For example, if device A is being monitored (with a device-type monitor) and a call is placed to device B (no monitor on B), then any connection state changes for either device A or B (such as when B answers the call) will be reported through device A’s monitor.

Monitoring is only guaranteed for devices in the switching sub-domain. Activity related to devices outside the switching sub-domain may be only partially available or completely unreported.

6.6.2.7 Monitor Filtering

The computing function can request that a set of events be filtered out (not sent) by the switching function. This information is specified in the monitorFilter parameter in the Monitor Start service request.

The monitorFilter parameter contains a list of filters that are grouped together into the following categories:

- Call Control events
- Call Associated events
- Media Stream events
- Physical Device Feature events
- Logical Device Feature events
- Maintenance events
- Voice Unit events
- Private events

The switching function indicates the actual list of events that will be sent by returning the monitorFilter parameter in the positive acknowledgement to the Monitor Start and Change Monitor Filter services.

The computing function can request that the filtered list of events for an existing monitor be changed by issuing the Change Monitor Filter service.

Some categories of events are not provided for call-type monitors. The capability exchange services indicate the categories of events that are supported by the switching function for call-type monitors.

6.6.3 Snapshot Services

Snapshot services are used by the computing function to determine information about a call or a device. These services may be used at any time, independently of, or in combination with existing monitors. For example, a computing function may snapshot a device prior to starting a monitor on the device, in order to obtain information on existing calls at the device.

6.7 Additional Services, Features & Behaviour

This section specifies standardized switching function features affecting calls at a given device that do not have an explicit service request associated with the invocation of the feature. These features are usually configured within the switching function or have a service request which sets up certain conditions at a device that causes a particular behaviour with respect to calls at the device. As a result, these features are only reflected through an event sequence from the switching function. The following sections explain these features and the event sequences associated with them.

6.7.1 Forwarding

The forwarding feature is a trigger at a device that will redirect incoming calls to another device based on a specific condition. The following are the types of conditions that would trigger the redirection, or forwarding of the incoming call:

1. *Immediate* - This condition indicates that if a call arrives at a device, it is immediately redirected to another device.
2. *Busy* - This condition indicates that if a call arrives at a device, and the device is busy with another call, then the incoming call will be redirected to another device.
3. *No Answer* - This condition indicates that if a call arrives at a device, and the call is not answered within a certain number of rings or within a specific amount of time, then the incoming call will be redirected to another device.
4. *Do Not Disturb (DND)* - This condition indicates that if a call arrives at a device, and the device has the Do Not Disturb feature active at the device, then the incoming call will be redirected to another device. Note that the Do Not Disturb feature does not necessarily imply that incoming calls are forwarded.
5. *Type of Call Origination* - This condition indicates that if a call arrives at a device, and the originating device is a specific class (i.e., external, such as a device that is outside the switching sub-domain, or internal, such as a device that is within the switching sub-domain), then the incoming call will be redirected to another device. This condition can be used in combination with the others to create a compound condition. For example, if busy with another call and the calling device is outside the switching sub-domain, then redirect the call to another device.

Switching functions may support one or both of the following levels of forwarding settings:

- switching function default settings
- User specified settings

Switching function default settings are a single set of forwarding-type/forward-destination combinations that can be activated and deactivated as a set. The set includes all of the CSTA forwarding-types defined and the forward-destinations for each type. Activation, deactivation, or changes to the forward-destinations are not normally possible by users.

User specified settings are individual forwarding-type/forward-destination combinations that can be activated or deactivated one at a time. User specified settings supersede switching function default settings during activation, deactivation, and when forwarding occurs.

A switching function that supports switching function default settings may also support user specified settings. Switching function default settings are used for forwarding to a standard destination such as voice mail or an attendant. User specified settings may be used to override the default settings to forward calls temporarily to another office, for example.

A user specified forwarding type supersedes the same switching function default forwarding type when forwarding occurs. For example, a user specified type of “No Answer” and its corresponding forward destination supersede a switching function default type of “No Answer”. Note that this rule may not apply to types that are not alike. For example, a user specified type of “No Answer” (a delayed type of forwarding) does not supersede a switching function default type of “Immediate”, although a user specified type of “Immediate” does supersede a switching function default type of “No Answer” (since “No Answer” is a delayed type of forwarding).

The forwarding feature has service requests and events to control and observe the activation and deactivation of the forwarding triggers at the device (i.e., Get Forward, Set Forward, Forwarding). These service requests and events are documented in Clause 22, “Logical Device Features”, beginning on page 279, and do not actually forward the incoming call when it arrives at the device, but instead sets up the trigger to cause the switching function to perform the redirecting of the call. The computing function should use the capabilities exchange services to determine which of these services and events the switching function supports.

The computing function should use the capabilities exchange services to determine which of the following levels of forwarding settings are supported by the switching function:

- Switching function default settings (set of forwarding types and forward destinations).
- User specified settings.
 - Default forwarding type.
 - Default forward destination.

Switching function default settings may be activated or deactivated manually at the device, or by providing neither the forwarding type nor forward destination (forward DN) in Set Forward service requests.

User specified settings may be activated or deactivated manually at the device or by providing the forwarding type and/or the forward destination (forward DN) in the Set Forward service request. If the forwarding type is not specified and the forward destination is specified, the switching function uses a default forwarding type. Likewise, if the forwarding type is specified and the forward destination is not specified, the switching function uses a default forward destination.

The computing function is informed that default settings are being activated in the Get Forward positive acknowledgement and the Forwarding event.

When the call is redirected as a result of the (Immediate) forwarding feature, there are two basic event sequence models to indicate that the call has been forwarded. The following are the event sequence model definitions (Note that the computing function should use the capabilities exchange services to determine which of model or models that the switching function supports.):

1. *Forwarding Is Triggered before the Call Is Delivered to the Device* - There is basically no event sequence associated with this condition. The only characteristic associated with this event sequence is:
 - The first event associated with the delivery of the call to the new device will have an appropriate forwarding event cause. If the RedirectionDeviceID parameter is available in this event, it will be provided based upon the definition of the Call Control event and 12.3.19, "RedirectionDeviceID", on page 77. Refer to 6.7.6, "Tracking a Diverted Call", on page 41 for additional information on event sequences for forwarded calls.

If the call is forwarded multiple times under the same condition (e.g., forwarded from device 1 to device 2 which is forwarded to device 3), then the information indicating that the call was forwarded will only be the information from the last device the call was forwarded from (e.g., device 2). As a result, the computing function will only see that the call has been forwarded one time.

2. *Forwarding Is Triggered after the Call Is Delivered to the Device* - The event sequence is a Diverted event followed by the first event associated with the delivery of the call to the new device. The characteristics associated with this event sequence are:
 - Depending on the capabilities of the switching function, a Delivered event may or may not flow as a result of presenting the to-be-forwarded call to the device from which it will be diverted.
 - The Diverted event will have an appropriate forwarding event cause. (Note that the reporting of this event is dependent on the capabilities of the switching function.)
 - The first event associated with the delivery of the call to the new device will have an appropriate forwarding event cause. If the RedirectionDeviceID parameter is available in this event, it will be provided based upon the definition of the Call Control event and 12.3.19, "RedirectionDeviceID", on page 77. Refer to 6.7.6, "Tracking a Diverted Call", on page 41 for additional information on event sequences for forwarded calls.

If the call is forwarded multiple times under the same condition (e.g., forwarded from device 1 to device 2 which is forwarded to device 3), then the information indicating that the call was forwarded will be available each time the call is forward (e.g., device 1, device 2). This is possible because the call is actually delivered to the device before it is forward to another.

If the call is forward multiple times with a mixture of forwarding conditions (i.e., event sequence types), then the information indicating that the call was forwarded will be a mixture of the event sequences depending on the order of the forwarding conditions.

6.7.2 Connection Failure

The information indicating connection failure can be reported through several different event sequences. The computing function should use the capabilities exchange services to determine which of these services and events the switching function supports. The following are the possible event sequences associated with connection failure:

1. *Negative Acknowledgement* - When the switching function supports service requests that perform connection creation process and the switching function detects a failure, the negative acknowledgement can be used to indicate the failure to complete the connection. The following are the service requests associated with connection creation process:
 - Consultation Call
 - Deflect Call
 - Dial Digits
 - Join Call
 - Make Call
 - Make Predictive Call
 - Pickup Call
 - Single Step Conference Call
 - Single Step Transfer Call

If the switching function uses the negative acknowledgement to indicate the connection failure, then the appropriate error code will be used to indicate the particular failure.

2. *Support of the Failed Event with an Associated Failed Connection* - When the switching function detects a connection failure, it places that connection into the failed state. This indicates that the call control services which can be performed with respect to the connection are limited. The following is the list of call control services that are applicable:
 - Clear Call
 - Clear Connection
 - Call Back Call-Related
 - Call Back Message Call-Related
 - Camp On Call
 - Deflect Call
 - Intrude Call

When a connection enters the Failed state, the event sequence provided is a Failed event. The characteristics associated with this event sequence are:

- The Failed event will have an appropriate failure event cause.
 - The failedConnection parameter in the Failed event will contain a “complete” Connection Identifier (i.e., a Connection Identifier that has both a Device Identifier and Call Identifier)
 - The Failed event will be reported to all active device-type monitors associated with the call, as well as all call monitors associated with the call.
3. *Support of the Failed Event without an Associated Failed Connection* - This case is similar to the “Support of the Failed Event with an Associated Failed Connection” state (case 2). The difference is that when the switching function detects a connection failure, it does not create a connection for the failed device but instead

indicates to the computing function that call control services, with respect to the connection, are limited. The following is the list of call control services that are applicable to the connection in the call under these conditions:

- Clear Call
- Clear Connection
- Call Back Call-Related
- Call Back Message Call-Related
- Camp On Call
- Deflect Call
- Intrude Call

When the failure is detected, the event sequence provided is a Failed event. The characteristics associated with this event sequence are:

- The Failed event will have an appropriate failure event cause.
- The failedConnection parameter in the Failed event will contain a “Call ID only” Connection Identifier. This indicates that there is not a valid connection for the failed device in the call but that the appropriate call control service can be performed (i.e., Call Back Call-Related, Intrude Call, etc.) on the call.
- The Failed event will only be reported to the active device and call monitors associated with the devices that were in the call prior to the failure (i.e., if a device-type monitor was on the failed device, then the event sequence is not reported).

If the Camp On Call or Intrude Call service request is performed on the call, then the connection associated with the failed device will be created (i.e., a valid connection).

4. *Support of the Failed Event with an Associated Failed Connection, not reported via monitors on the failing device* - This case is similar to the “Support of the Failed Event with an Associated Failed Connection” state (case 2). The difference is for which monitors the Failed event is being sent: The Failed event will only be reported to the active device and call monitors associated with the devices that were in the call prior to the failure (i.e. if a device-type monitor was on the failed device, then the event sequence is not reported). Apart from this, all aspects from case 2 apply also to this case.

6.7.3 Recall

The Recall feature is a trigger that is associated with a call after a specific call control feature has been executed. When this feature is executed, it redirects or presents the call either back to the device on who’s behalf the call control feature was executed or to a switching function administrated destination associated with the specific call control feature. There are several types of call control services which can have this feature associated with them. For example:

- Hold Call
- Transfer Call
- Single Step Transfer Call
- Deflect Call
- Park Call

The event sequence associated with this feature is the Diverted event (only if the device to whom the call is being redirected is not already in the call) and the first event associated with the delivery of the call to the new device or the device that performed the call control feature. The characteristics for this event sequence are:

- The Diverted event will have an appropriate recall event cause. This event is only reported when the device to whom the call is being redirected is not already in the call (i.e., a recall to a connection that is already in the call). (Note that the reporting of this event is dependent on the capabilities of the switching function.)

- The first event associated with the delivery of the call to the new device (i.e. Delivered), or the device that performed the call control feature will have an appropriate recall event cause (in either the Delivered, Held, Queued, etc.). If the lastRedirectionDevice parameter is available in this event, and the call was actually redirected to another device outside the current call, then it will be provided based upon the definition for this parameter. (Refer to the definition of the Call Control events and lastRedirectionDevice parameter for more details.) If the callingDevice parameter is available in this event, it may contain the same callingDevice information prior to the recall. This means that if the calling device is the Subject Device of the event, then the information in the callingDevice and corresponding SubjectDeviceID (e.g., Delivered event SubjectDeviceID = alertingDevice) parameters may be the same. If the switching function does not retain this information with the call, then the callingDevice parameter will contain a value of “Not Known”.

6.7.4 Call Back

The Call Back feature is a trigger which is set up within the switching function. The trigger is used to initiate a call between a particular pair of devices. The pair of devices is comprised of a calling device (i.e., the device on who's behalf the trigger is setup) and the called device (i.e., the device whom the calling device wants to initiate a call to when certain conditions associated with the called device are met). The type of conditions associated with the trigger is switching function specific. A common type of condition is the called device is no longer actively involved in a call(s). The trigger is activated for the calling device by either the Call Back Call-Related or Call Back Non-Call-Related services. The trigger is deactivated by one of the following: successful execution of the trigger, the Cancel Call Back service, or a switching function specific timeout period. Once the trigger is activated, the switching function waits for the particular condition associated with the Call Back feature to be met. Once met, the switching function initiates a call on behalf of the calling device to the called device. This is done by first prompting the calling device (if supported by the device) and then initiating the call to the called device.

The event sequence associated with execution of the Call Back trigger is the Service Initiated event (only if the calling device is prompted), Originated event and the events associated with the called device's involvement in the call. The characteristic for this event sequence is that both the Service Initiated (if supported) and Originated events will have an event cause of Call Back.

6.7.5 External Calls

A call is considered to be external when there is at least one device in the call that is outside the switching sub-domain. For more details on how the switching function represents these devices within the switching sub-domain, refer to 6.1.3.4.2, “Network Interface Device Category (to be defined)”, on page 20. The activities associated with external calls are broken down into two categories:

- *Incoming Calls* - A call is being initiated from a device outside the switching sub-domain.
- *Outgoing Calls* - A device inside the switching sub-domain is adding or initiating a call to a device outside the switching sub-domain.

These categories are represented by different event sequences. The characteristics associated with these event sequences are:

- Incoming Calls
 - A Service Initiated event is generated for the network interface device when the network interface device is allocated (e.g., seized) for the external incoming call. The initiatingDevice parameter will contain the information on which network interface device is being used for the call. Note that this event is only generated if the network interface device is monitored.
 - The Digits Dialed event is generated for the network interface device when a portion of the dialling sequence has been received over the network interface device. Note that this event is only generated if the network interface device is monitored.
 - The Originated event is generated for the network interface device when the external incoming call has originated from the network interface device. The NIDDevice parameter will contain the information on which network interface device is being used for the call. Note that this event is only generated if the network interface device is monitored.

- In all subsequent events (independent of whether or not the network interface device is observable), this incoming call is distinguished from an internal incoming call by the presence of the associatedCallingDevice (i.e., containing either a Device Identifier or a value of “Not Known”). The information in the associatedCallingDevice parameter is first associated with the call when it enters the switching sub-domain (i.e., Service Initiated or Originated events) and will be present until the calling device leaves the call.
- Outgoing Calls
 - When initiating a connection to a device outside the switching sub-domain and the switching function is associating the Network Interface Device with the outside device, a Network Reached event is reported. In addition, the Network Reached event is the first event that indicates the call is an external outgoing call. Subsequent events (if available) will contain the associatedCalledDevice parameter (i.e., the value from the networkInterfaceUsed parameter of the Network Reached event) and will be present until the call is cleared. In addition, until the Network Reached event is generated, the call is considered to be an internal call.
 - The event sequence after the Network Reached event that is associated with the device outside the switching sub-domain may be limited but the events that are reported will contain one of the event causes documented in the event definition. If the Network Reached event contained the networkCapability parameter, future Network Capabilities Changed events may be provided indicating a change in the signalling capability of the network and ultimately the types of events that can be provided.

6.7.6 Tracking a Diverted Call

When observing a call or a device in a call, and the call diverts from a device in the call, the computing function shall use the Diverted event to track the progress of the call as a result of the redirection.

If the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services), the computing function shall use parameters in the first event after the call has been diverted to properly track the progress of the call as a result of the redirection. The device identifiers are used to observe the movement of the call and the event cause is provided to indicate what caused the movement of the call. (Note that the call may have been diverted several times between the previous event (if one was generated) and the first event after the diversion. As a result, the computing function can only ascertain that either one or two redirections have occurred.)

6.7.7 Media Stream Access (to be defined)

6.7.8 Routeing Services

A switching function uses Routeing services when it needs the computing function to supply call destinations. This may be on a call-by-call basis or it may be non-call related. The computing function can use internal databases together with call information to determine a destination, or route, for each call. For example, the computing function might use the caller's number and information in a database to route incoming calls.

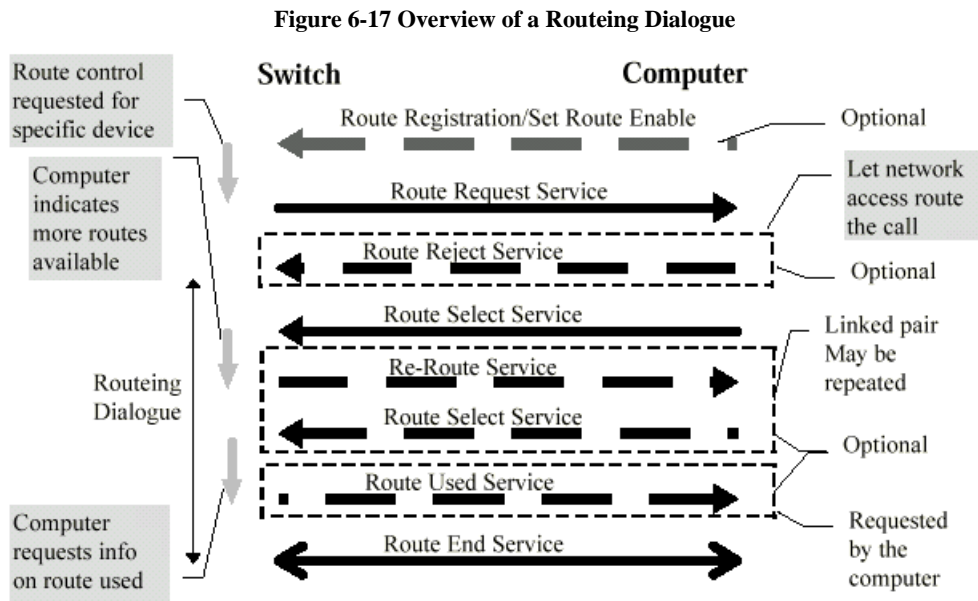
A switching function may support Routeing services for any type of call (e.g., external outgoing, external incoming, intra-switching sub-domain). Routeing services may require that the switching function be configured to direct calls to a device known as a *routeing device*. This device shall be addressable (i.e., visible within the switching sub-domain) with respect to Routeing services but may or may not be addressable with respect to other services (e.g., Call Control, Monitoring).

The routeing device may be a virtual device used only for routeing and thus may not be monitorable. The way a particular virtual routeing type device is used by a switching sub-domain is specific to each implementation. Examples include:

- a routeing device could be used to route all outgoing external calls from all devices within a given switching sub-domain
- a routeing device could be used to route all incoming external calls independent of the network interface device being used

- a routing device could be used to route all calls that are considered to be priority calls independent of their origin.

A switching function implementation will implement as many routing devices as it requires in order to reflect the different routing processes it supports.



Routing services are used within a sequential “routing dialogue” such as that represented in Figure 6-17. (Note that none of the routing services return positive acknowledgements. Negative acknowledgements, though provided by routing services when applicable, are not shown in the figure.)

A routing dialogue is typically initiated by the switching function when a call is directed to a device and particular conditions are met for that call at that device. The conditions at a device under which the switching function may initiate a routing dialogue are determined by its Route Mode and the Route Registration Service. Through these mechanisms the computing function may specify to the switching function that when calls encounter a particular device, the computing function should be consulted for a proposed route.

Routing services are linked within a routing dialogue by the routing cross reference identifier (routingCrossRefID). A routing cross reference identifier is provided by the Switching function as part of the Route Request Service used to initiate a routing dialogue. This routing cross reference identifier is quoted by each subsequent invocation of a routing service in the routing dialogue.

Route Requests generated by the switching function may be call-related or non-call related.

6.7.8.1 Route Registration and Route Mode

Table 6-7 below specifies the conditions that must be satisfied before a given switching function initiates a routing dialogue for a given routing device by generating a Route Request. The switching function’s behaviour is governed by its support for the routing registration services and support for the Route Mode attribute. Registration has no affect on routeMode, and enabling/disabling routeMode has no affect on registration. In order to use

routing services for a given routing device, a computing function must satisfy the conditions specified in Table 6-7 by invoking the appropriate services.

Table 6-7 Routing Behaviour

	Registration Not Supported	Registration Supported
Route Mode Supported	<ul style="list-style-type: none"> • Registration not required • RouteMode must be enabled • Switching Function must initiate routing dialogue if a call of any media class arrives 	<ul style="list-style-type: none"> • Registration required • RouteMode must be enabled • Switching Function must initiate routing dialogue if a call of any media class arrives that matches the media class requested
Route Mode Not Supported	<ul style="list-style-type: none"> • RouteMode implicitly enabled • Registration not required • Switching Function may initiate routing dialogue at its discretion 	<ul style="list-style-type: none"> • RouteMode implicitly enabled • Registration required (specific or all) • Switching Function must initiate routing dialogue if a call arrives that matches the media class requested

The positive acknowledgement to the Route Register service contains the route register request identifier (routeRegisterReqID) that the computing function uses to identify service requests that arrive for this registration.

If the switching function supports the Route Registration services, then the computing function shall use these services to register as a routing server before it can route calls. If the switching function does not support the Route Registration services, then the computing function may receive route service requests for any routing device at any time.

The computing function may either register as the routing server for a specific routing device or, if supported by the switching function, as a routing server for all routing devices within the switching sub-domain.

A route registration can be cancelled using the Route Register Cancel service. Once this service is positively acknowledged, the switching function will no longer send route service requests to the computing function. Additionally, the switching function can cancel a route registration at any time by sending the computing function a Route Register Abort service request.

Routing services for a particular device may be suspended without cancelling route registration by disabling its Route Mode. This does not effect route registration and route requests for the given device will resume when its Route Mode is enabled.

The capabilities exchange services can be used to determine if the switching function supports the Route Registration services and if so, if the capability to register for all routing devices is supported. The capabilities exchange services can also be used to determine if the switching function supports the Route Mode attribute.

6.7.8.2 Call Routing

An example of a routing process may involve the following sequence of steps:

1. The switching function receives a call at the routing device. The routing device may be any device within the switching sub-domain.
2. When the call arrives at the routing device, the switching function creates a routing dialogue for the call. The switching function allocates a routing cross reference identifier (routingCrossRefID) that references this routing dialogue.
3. The switching function sends the Route Request service to the computing function (that registered as the routing server) for the routing device or as the routing server for all routing devices within the switching sub-domain. This service request contains the routing cross reference identifier, the route registration request identifier (if supported), and call information such as the Connection Identifier for the call, and calling and called numbers.
4. The computing function decides whether to reject the Routing service request for this call, provide a route for the call, or end the routing dialogue. If the computing function decides to reject the call, it sends the

switching function a Route Reject service request. If the computing function decides to provide a route for the call, it sends the switching function a Route Select service request containing the destination for the call. The computing function may include an optional flag in the Route Select service request (i.e., routeUsed) instructing the switching function to inform it of the call's final destination. The final destination may be different than the computing function-provided destination when switching function features such as call forwarding redirect the call. If the computing function decides to end the routing dialogue, it sends the switching function a Route End service request. In this case, the computing function does not provide a destination for the call and the switching function uses an alternate mechanism (not defined) to route the call.

5. If the switching function receives a Route Reject service request, then it returns the call to the network for alternate routing, and sends the computing function a Route End service request to indicate that the routing dialogue is ended. If the switching function receives a Route Select service request, it attempts to route the call to the computing function-provided destination. If the destination is valid, the switching function routes the call to that destination and sends the computing function a Route End service request to terminate the routing dialogue. If the computing function-provided destination is not valid (e.g., invalid directory number, destination busy), then the switching function may send a Re-Route service request to the computing function to request a route to an alternate destination. If the switching function receives a Route End service request, it terminates the routing dialogue.
6. If the computing function receives a Re-Route service request it can select a different destination for the call and send the switching function another Route Select service request. Depending on the switching function implementation, the re-routing service request exchange can repeat until the computing function provides an acceptable route. The computing function will find out about a successful route when the switching function sends a Route End service request or if the computing function included the routeUsed flag in its last Route Select service request.

Either the switching function or the computing function may send a Route End service request at any time to end the routing process and terminate the routing dialogue. This releases the routing cross reference identifier for use in the future. This service request indicates, for example, that the computing function does not want to route the call, or the switching function (usually in the absence of a Route Select service request) routed the call using some default mechanism within the switching function.

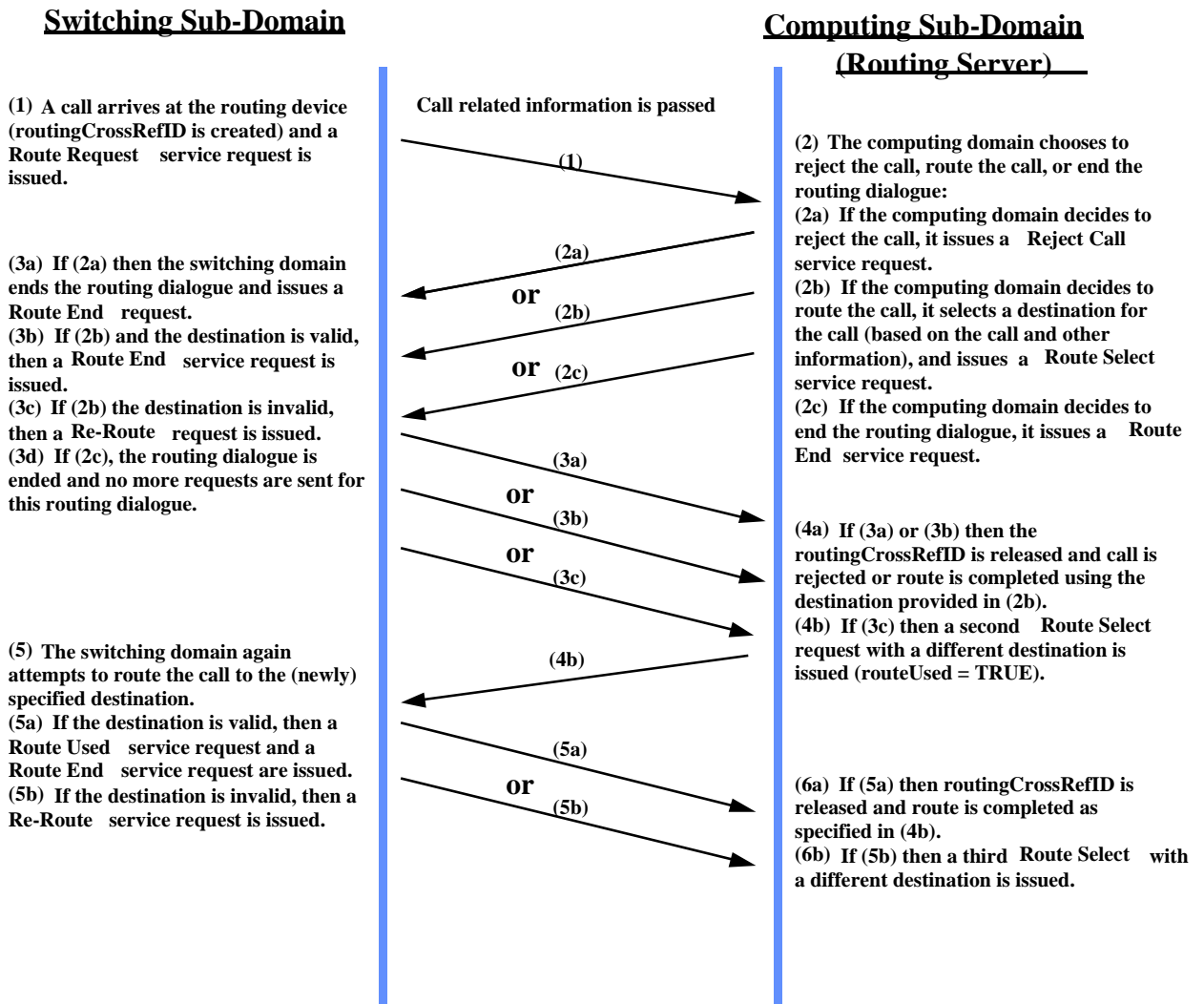
Note that a conflict may arise in this dialogue if the computing function invokes the Route End Service, for example to indicate that no more alternative routes are available, but still wants to receive a route used report via the Route Used Service invoked by the switching function. Avoidance or resolution of this conflict is the responsibility of the computing and/or switching function implementation(s).

A call that is not successfully routed does not necessarily mean that the call is cleared or not answered. Most switching function implementations will have a default mechanism for handling a call at a routing device when the computing function has failed to provide an acceptable destination for the call. The switching function shall send a Route End service request to the computing function when it terminates the routing dialogue, unless the routing dialogue was terminated by a Route End service request from the computing function first.

The minimum set of services a switching function shall provide if it supports routing are: Route Request, Route Select, and Route End (from the switching function). Other routing services may be provided in any combination in addition to this minimum set.

Figure 6-18 illustrates the typical Routing procedure.

Figure 6-18 Routeing Procedure



6.7.8.3 Route Register Request ID and the Routing Cross Reference ID

The routeing services use two identifiers to refer to different software objects in the switching sub-domain. The route register request identifier (routeRegisterReqID) identifies a routeing registration for which the computing function (acting as a routeing server) will receive Routeing service requests. This identifier may be associated with a particular routeing device within the switching sub-domain or it may indicate that the computing function is the routeing server for all routeing devices within the switching sub-domain. When the computing function uses the Route Register service to register for routeing services, it receives a routeRegisterReqID in the positive acknowledgement from the switching function. The routeRegisterReqID is only valid until the routeing session is ended by the computing function or switching function.

Within a routeing registration (routeRegisterReqID) the switching function may initiate many routeing dialogues (shown in Figure 6-18) to route multiple calls. A switching function uses a routeing cross reference identifier (routeingCrossRefID) to refer to each routeing dialogue. The computing function receives a routeingCrossRefID in each Route Request service request. The Route Request service initiates a routeing dialogue. The routeingCrossRefID is only valid for the duration of the routeing dialogue pertaining to a specific call.

The routeing cross reference identifier (routeingCrossRefID) is unique within the routeing registration (routeRegisterReqID). Some switching functions may provide the additional benefit of a unique routeing cross reference identifier across the entire switching sub-domain. This is also the case if routeing registration is not

supported by the switching function. Routeing registration identifiers (routeRegisterReqIDs) are unique across a given CSTA service boundary.

6.7.8.4 Monitoring of Routeing Device

Some switching function implementations may support monitoring of routeing devices. For those computing functions that have an active monitor on the routeing device, any activity at the device (for instance call control activity) shall generate the relevant event sequence as specified throughout this specification.

6.7.8.5 Routeing Services with respect to Media Class

A routeing device can support the routeing of calls of any combination of media class (i.e., voice or digital data or both). Refer to the media class component of 12.2.4, "MediaCallCharacteristics", on page 60 for the complete set of possible values.

Once the routeing session is visible to the computing function through the Route Request service, the media characteristics of the call will be identified and associated with the routeing cross reference identifier.

6.7.9 Device Maintenance

Device Maintenance events indicate changes in the maintenance state of a device. These events indicate if a device has been taken out of service (can no longer accept calls or be manipulated by the computing function), or if a device has been placed back in service.

6.7.10 Prompting

Some CSTA services (Make Call, Call Back, Pickup, Join Call, for example) may require to prompt the user of the targeted device in order to take that device off-hook. The implementation of a prompting mechanism is switching function specific (display flashing, ring pattern, lamp blinking, etc.).

For CSTA services that specify prompting (except the Make Call service), the switching function shall support (as indicated by the capability exchange services) one of the two possible prompting modes:

- prompting is a pre-condition to a service - in this mode prompting occurs before the execution of the CSTA service. The Service Initiated event that indicates prompting shall flow before any other service specific events and shall contain connection identifier that is not associated with the CSTA service. After the device goes offhook, a Connection Cleared event associated with the prompt is generated and the CSTA service that initiated the prompt is executed.
- prompting is part of a service - in this mode, prompting is part of the execution of the service. The Service Initiated event that indicates prompting is part of the completion criteria for the service and the connection identifier used in the Service Initiated event is associated with the CSTA service.

For information on event sequences with respect to prompting in the context of specific services, refer to the Monitoring Event Sequences associated with a CSTA service.

7 Association Establishment (to be defined)

8 Security Service (to be defined)

9 Generic Service Requirements

9.1 Service Request

This standard defines a set of CSTA operations that can be used to control and observe objects within a switching and/or special resource function. The CSTA operations are defined as "Services" in which one function requests, across the service boundary, that the other function perform a given CSTA operation. Services are defined for the CSTA service boundaries between the computing function, switching function, and special resource function. Services are defined in terms of what they accomplish (i.e. functionality), not how they should be implemented.

When one function sends a service request to the other function to perform a service with a given set of parameter values, it is called a *service request*. Each service defined in this Standard falls into one of following categories based upon the direction of the service request:

- *Switching Function Service* - Switching function services are services where the computing function is the client (i.e., service requestor) and the switching function is the server. An example of a switching function service is the Make Call service.
- *Computing Function Service* - Computing function services are services where the switching function is the client (i.e., service requestor) and the computing function is the server. An example of a computing function service is the Route Request service.
- *Special Resource Function Service* - Special Resource function services are services where the computing function is the client (i.e., service requestor) and the special resource function is the server. An example of a special resource function service is the Play Message service.
- *Bi-directional Service* - Bi-directional services are services where either the switching/special resource function or the computing function can be the client (i.e., service requestor). An example of a bi-directional service is the System Status service.

Some switching/special resource functions implementations support registration mechanisms that allow the computing function to indicate that it would like to receive service requests in a certain category (e.g., routing, system status, escape) from the switching/special resource functions. (If the switching/special resource function indicates that it supports the computing function services in a particular category but does not support the registration mechanism, the computing function shall be prepared to handle the requests without previous registration.)

If the server detects that a service request is invalid, a negative acknowledgement shall be generated.

Every service request and service response defined in this Standard allows the inclusion of non-standardized, Private Data, that shall be informational in nature. Refer to 9.4, “Vendor Specific Extensions (to be defined)”, on page 48, for more information.

9.2 Service Response (Acknowledgements)

The other part of a service is the acknowledgement to the service request. This acknowledgement is used by the requesting function to verify that the other function has received the service request and that some level of processing has been performed with respect to the service. There are two types of acknowledgements: *positive acknowledgements* and *negative acknowledgements*, for a given service, as well as two types of positive acknowledgement models which a given service can adhere to. These definitions are documented in the following sections.

Note that there are some services defined in this Standard that do not provide a positive acknowledgement. For these services, if the service request is invalid, a negative acknowledgement shall be generated.

9.2.1 Positive Acknowledgement Models

All acknowledgements to each service request defined in this Standard shall follow the principles outlined by one of two models defined below. The computing function learns which model a switching function supports for each service through the capability exchange services described in Clause 13, “Capability Exchange Services”, beginning on page 80.

9.2.1.1 Atomic Model

Switching functions that indicate support of the atomic acknowledgement model designate that the particular service request can be accomplished in a single logical step. This acknowledgement model reflects whether or not the service request has meet the completion conditions as documented by each individual service.

An atomic positive acknowledgement indicates that not only were the parameters on the service request valid, but the switching function has successfully completed the service requested as defined in that service’s “Service Completion Conditions” section. The condition of the call(s) and/or connection states of the device(s) associated with the service request have transitioned to that service’s Operational Model After state.

9.2.1.2 Multi-Step Model

Switching functions that indicate support of the multi-step acknowledgement model designate that the particular service request is accomplished as its name implies, in multiple logical steps. This acknowledgement model reflects

whether or not the parameters passed on the service were valid but does not guarantee anything as far as the completion conditions is concerned for the service.

A multi-step positive acknowledgement guarantees only that the parameters passed on the service request were accepted by the switching function to be valid. This positive acknowledgement does not determine if the service request's completion criteria are met. (However, depending on the switching function, the positive acknowledgement may indicate, in certain situations, the service request's completion conditions.) Therefore the computing function shall monitor for events associated with the particular service request, affected call(s) or device(s) to verify completion. A computing function shall also be prepared to handle the Service Completion Failure event and/or the Failed or Connection Cleared events after receiving the positive acknowledgement. The Service Completion Failure event will only be reported to the computing function which issues the service request *and* has a device-type monitor on the device which has or had connection(s) that were used in the particular request. Each of these events are provided by the switching function to indicate that the completion conditions for the service was not met.

If, through the event flow, a failure is detected, it is up to the computing function to apply the appropriate recovery to return the call(s) and/or device(s) back to the original conditions (if needed). Finally, a computing function should not issue subsequent service requests for a device until a previous multi-step service request's completion conditions has been satisfied. Doing so may result in unpredictable results generated by the switching function.

9.2.2 Negative Acknowledgement

A negative acknowledgement indicates that the service request has failed and the condition of the call(s) and/or connection states of the device(s) associated with the service request have not changed as a result of the failure (i.e., they remain as they were in the service's Operational Model Before state).

9.3 Diagnostic Error Definitions (to be defined)

9.3.1 Operation errors (to be defined)

9.3.2 Security errors (to be defined)

9.3.3 State incompatibility errors (to be defined)

9.3.4 System resource availability errors (to be defined)

9.3.5 Subscribed resource availability errors (to be defined)

9.3.6 Performance management errors (to be defined)

9.3.7 CSTA Private Data Information Errors (to be defined)

9.3.8 Unspecified errors (to be defined)

9.4 Vendor Specific Extensions (to be defined)

9.4.1 Private Data (to be defined)

9.4.2 Private Data Version Negotiation (to be defined)

9.4.3 Defined Services and Events (to be defined)

9.4.4 Escape Services and Private Event (to be defined)

9.5 General Services and Event Functional Requirements

The following sections discuss functional requirements that are applicable to the services and events specified in this Standard.

9.5.1 Services

1. If a service is performed manually from a device, computing functions that have device-type or call-type (for device or call) monitors on this device receive the same event sequence as reported when performing the service through the service boundary (i.e., computing function-initiated). Refer to the appropriate service's "Monitoring Event Sequence" sections for details.

Depending on the particular switching function, additional events may also be reported as part of manual invocation services. For example: ²

2. These events are only reported if the computing function has the appropriate monitors started for the device (i.e., correct filters and type of monitor) and those monitors are supported by the switching function.

- A Held event, if the device already has an active call.
 - A Service Initiated (event cause of NewCall) event for the device because a new call is needed to execute the service manually. This is followed by a Connection Cleared (event cause of Normal Clearing) event for the device when the service has been executed.
 - Logical and Physical device events that are associated with the execution of the service. These events may appear any time during the execution of the service.
2. If a service request is invoked after a device has manually gone off-hook (Service Initiated event), an implementation may either accept the service or it may reject the service. If it accepts the service, (unless otherwise specified for a particular service or event), the connection that has gone off-hook will be cleared and the computing function will receive a Connection Cleared event, followed by service specific events.
 3. Other than for calls in the Initiated state, a service only affects connections that are specified by its service description. If, prior to its completion, the execution of the requested service would cause the switching function to affect any other connections, then the service shall be rejected with a negative acknowledgement.
 4. If the switching function permits the passing of Connection Identifiers without Call Identifiers, then the Device Identifiers they contain shall be within the switching sub-domain. In addition, if DeviceIDs only are passed in the Connection Identifiers, then:
 - If only one call exists at the specified device and the service request supports a single ConnectionID, its connection shall be in one of the initial states specified by the service or the service will be rejected.
 - If more than one call is in an initial state defined by the service, the service request will be rejected.
 - If two calls exist at the specified device and the service request supports two ConnectionIDs in the service request, the DeviceIDs within the ConnectionIDs shall be identical or the service will be rejected.
 - If two calls exist at the specified device and the service request supports two ConnectionIDs in the service request, both calls shall be a valid combination of initial states specified for that service or the service will be rejected.
 - If more than two calls exist at the specified device, the service will be rejected unless full and valid ConnectionIDs are specified.
 5. If the device that is the subject of a service request is not capable of performing the service, a negative acknowledgement with an appropriate error code will be provided.
 6. For optional parameters in service requests the following requirements apply:
 - a. If an optional parameter is supported by the switching function but is not supplied in a service request, the switching function uses the specified default value associated with that parameter unless otherwise specified.
 - b. If an optional parameter is not supported by the switching function, the switching function uses its administered value unless otherwise specified. (In addition, if the non-supported parameter is passed in the service request, see Services Requirement #7).
 7. The switching function may either reject service requests that contain optional parameters that it does not support, or, it may accept the service request and ignore the unsupported optional parameters. However, the switching function shall handle unsupported optional parameters the same way for all service requests. The switching function indicates how it handles unsupported optional parameters via the capabilities exchange services.
 8. When setting a value for a Physical or Logical Device Feature (specifically the “Set” features described in Clause 21, “Physical Device Features”, on page 276 and Clause 22, “Logical Device Features”, on page 279), the switching function shall return a positive acknowledgement when the feature is already set to the requested value specified in the service request. (Since the service request, in this case, did not result in a change of feature status, a feature event will not be generated.)

9. It is the switching function's responsibility to verify that connections in a call are in their proper initial states prior to accepting a service request. Acceptable states are documented in each service request's description.

9.5.2 Events

1. For the same telephony situation, the event generated for a call-type monitor will be the same as the event generated for a device-type monitor, except that the localConnectionInfo and the servicesPermitted parameters described in the Call Control event descriptions are not provided for events generated for call-type monitors.
2. If the computing function has call-type monitoring in effect, the event seen for that monitor will be the same event as the one seen for the subject device from a device-type monitor.
3. If the Device Identifier portion of a Connection Identifier is a static Device Identifier, then that portion of the Connection Identifier and the Device Identifier parameters in an event will not necessarily be the same. For example, the switching function may have a static internal representation of a device which will be used in the Connection Identifier, but the actual diallable representation for the same device may be different and may be used in one of the Device Identifier parameters in the same event. This requirement is in addition to and does not supersede the definition for the Connection Identifier or Device Identifier parameters described in 12.3.8, "ConnectionID", on page 74 and 12.3.9, "DeviceID", on page 75.
4. The set of state transitions (refer to Figure 6-15, "Connection State Model" on page 25) supports the services and features documented in this Standard.

10 CSTA Device Identifier Formats

This clause describes the formats that may be used for Device Identifiers, their usage, and examples.

10.1 Device Identifier Formats

The possible types of Device Identifiers formats are:

- *Diallable Digits* - this format is a sequence of characters to be dialled to reach a device. The sequence of characters may contain diallable digits and/or special characters that specify to the switching function how digits should be dialled (";" indicates that a pause should be inserted into the dialling sequence, for example). This format must be used when special dialling characters are required or when it is necessary to provide partial or incomplete dialling sequences.
- *Switching Function Representation* - this format is a sequence of characters that is used to reference devices within a switching sub-domain. In addition to specifying the directory number of the device, it also provides the ability to specify call appearance, agent identifier, subaddress, name, etc.
- *Device Number* - this format is a non-diallable, integer representation of a Device Identifier. This format of Device Identifier can be used to reference switching sub-domain devices that may not be typically associated with a diallable number such as trunks, line cards, etc.

In this section, the following example will be reflected. The called number is a subscriber in the US (country code 1) in San Jose (area code 408). The local number is 996 1010. The extension is 321. The name of the subscriber is "John Smith".

10.1.1 Diallable Digits

Generic Format: DD

A first character of the Device Identifier string which is not "N", "\", or "O" indicates that the Device Identifier uses the Diallable Digits format. This format may contain from 0 (null Device Identifier) to 64 characters. DD is a string of dialling commands/digits. The following is the list of the complete set of permitted dialling commands/digits and their definitions:

- 0-9** These characters represents the number digits on a telephone keypad.
- *** This represents the "*" character, typically found on a telephone keypad.
- #** This represents the "#" character, typically found on a telephone keypad.

A-D	These characters represent DTMF digits.
!	The exclamation mark indicates that a hookflash is to be inserted into the dial string.
P	The character P followed by a string of digits indicates that the string of digits is to be pulse dialed.
T	The character T followed by a string of digits indicates that the string of digits is to be tone dialed.
,	The comma character indicates that dialling is to be paused. The length of the pause is provided by the switching function through the capabilities exchange services. Multiple commas can be used to create a long pause.
W	The character W followed by a string of digits indicates that the string of digits is to be dialed only after dial tone has been detected by the switching function.
@	The “at” symbol indicates that the switching function shall wait for “Quiet Answer” before dialling the rest of the string. This means that the switching function shall wait for remote ringing indication, followed by 5 seconds of silence.
\$	This dollar sign indicates that the switching function shall wait for the billing signal (i.e., credit card prompt tone) before continuing.
;	The semi-colon character indicates that the digit string is incomplete and more digits will be dialed using the Dial Digits service. This character may only be used in a Diallable String Device Identifier.

Examples:

- If the number is called from France (country prefix 00³), the string is “00,14089961010W321”.
- If the number is called from a switch in New York (dial 9 to get outside line), the string is “9,14089961010W321”.
- If the number is called from San Jose, the string is “9961010W321”.
- If the number is called from inside the subscriber’s PBX, the string is “321”.

Functional Requirements:

1. The switching function shall accept, as a minimum, digits 0-9 of this format when the computing function wants to make a call.
2. The diallable digits format shall be used to represent a device’s dialling sequence. A device’s dialling sequence is a string of outband digits used to initiate a call with another device. When placing a call from a device to another device, there are basically two ways a device’s dialling sequence can be used:
 - a. The entire sequence of digits is dialed to reach the destination. This is the most common way to place a call.
 - b. The dialling sequence is broken up into a number of stages in order to execute and complete the call. This is called “multi-stage” dialling in this Standard. This type of dialling is needed in cases where the switching function prompts the device for more digits (by sending dialtone again or some other tone).

Note that switching functions support different combinations of dialling sequences.

10.1.2 Switching Function Representation

Generic Format: N<DN!SA&CA/EXT%AID>NM (*in this order*)

The syntax of the generic format is broken down as follows:

3. The country prefix is the sequence of digits that needs to be dialed to make an international call (011 when calling from the US). It is always followed by the called country code.

- N** The “N” character at the beginning of the Device Identifier string (which is 2 to 64 characters in length) indicates that the Device Identifier uses the Switching Function Representation format. At least one of the following components needs to be present in this format:
- < >** The angled brackets characters encompass the string when a name (NM) string representing the person associated with the device is provided after the “>” character. If the character “<” is not the first character in the string after the N then the string will not have a name string associated with it.
- DN** The first string of characters represents the Directory Number (DN) associated with the given device. The Directory Number shall contain characters selected from the following set: “0” through “9”, “*”, “#”, DTMF digits “A” through “D”. The Directory Number may use any of the following notations:
- Implicit TON (Type Of Number)
example: “0014089961010”⁴
(refer to ECMA-155)
 - PublicTON - unknown
(refer to ECMA-155)
 - PublicTON - international number
example: “14089961010”
(refer to ITU-T E.160)
 - PublicTON - national
example: “4089961010”
(refer to ITU-T E.160)
 - PublicTON - subscriber
example: “9961010”
(refer to ITU-T E.160)
 - PublicTON - abbreviated
example: “17”
(refer to ITU-T E.131)
 - PrivateTON - unknown
(refer to ECMA-155)
 - PrivateTON - level 3 regional
example: “41396557321”
(refer to ECMA-155)
 - PrivateTON - level 2 regional
example: “96557321”
(refer to ECMA-155)
 - PrivateTON - level 1 regional
example: “557321”
(refer to ECMA-155)
 - PrivateTON - local
example: “321”
(refer to ECMA-155)
 - PrivateTON - abbreviated
example: “2”
(refer to ECMA-155)

4. This example is a caller in France dialling the country prefix (00), the USA country code (1), the trunk code (408) and the subscriber number.

- Other (other numbering plans)
 - Generic (the notation is unknown)
- !
- This exclamation mark character represents the start of a Sub-Address (SA) string. If the “!” character is not present, then there will be no sub-address associated with this Device Identifier string. The termination character for the sub-address string will be the next key character found in the string or null.
- &
- The ampersand symbol represents the start of a Call Appearance (CA) string. It is added to the logical element’s device identifier to uniquely identify an addressable standard appearance. The value of the string is switching function specific. The valid characters for the call appearance string are 0-9. The termination character for the call appearance string will be the next key character found in the string or null. Refer to 6.1.3.2.1, “Appearance”.
- /
- The slash symbol represents the start of a physical element extension (EXT) string. It is added to the logical element’s device identifier to uniquely identify a bridged appearance. Its value is the physical element’s device identifier that is associated with the appearance. The termination character for the physical element extension string will be the next key character found in the string or null. Refer to 6.1.3.2.1, “Appearance”.
- %
- The percent sign represents the start of an Agent ID (AID) string. This string represents an ACD agent identifier associated with a device. This string may be present when the computing function wants to focus a service at a specific agent identifier that is associated with a device or when the switching function generates an event that is associated with a particular device and agent. The valid characters for the agent identifier string are A-Z and 0-9. If the “%” character is not present then there will be no agent identifier associated with this Device Identifier string. The termination character for the agent identifier string will be the next key character found in the string or null.
- NM
- The name string (NM) represents the person associated with the device. This string can be used for selecting a Device Identifier associated with a user or for logging and informational purposes. The name string may contain any character.

Example:

- If the Device Identifier is PublicTON International, then the string can be “N14089961010”.
- If the Device Identifier is PublicTON Subscriber, then the string can be “N<9961010>John Smith”.

Functional Requirements:

1. This format shall always contain at least a directory number string or an agent ID string.
2. The interpretation of additional digits beyond those that are required to reach a destination are switching function specific.
3. When there is more than one bridged appearance associated with a single physical element (see 6.1.3.3.6, “Hybrid”, on page 19 for an example) there are two methods for representing these appearances: One is to have a unique call appearance (CA) and physical element extension (EXT) combination for each appearance where EXT is used to represent the given physical element and CA is used to represent multiple appearances associated with the same physical element. The other is to have a single EXT for each appearance, independently of their association with the physical element. In either case, the resulting Device Identifier is unique for the given appearance.

10.1.3 Device Number

Generic Format:

The Device Number format represents a Device Identifier using an integer. The integer shall be maximum size of four octets.

10.2 Functional Requirements

1. If the switching function detects a problem with a Device Identifier, the service will be rejected with a negative acknowledgement.
2. The switching function may use any format in service acknowledgements and events.
3. For Device Identifiers in service requests, the computing function should check the deviceIDFormat parameter in a capabilities exchange service to determine:
 - Which formats are supported.
 - For the Switching Function Representation format, which notations are supported.
 - For the Diallable Digits format, which special characters are supported.
4. When providing a null Device Identifier, the Diallable Digits Format is used.

11 Template Descriptions

This Clause explains the template formats used to describe the CSTA services, events, and parameter types defined in this Standard.

11.1 Service Template

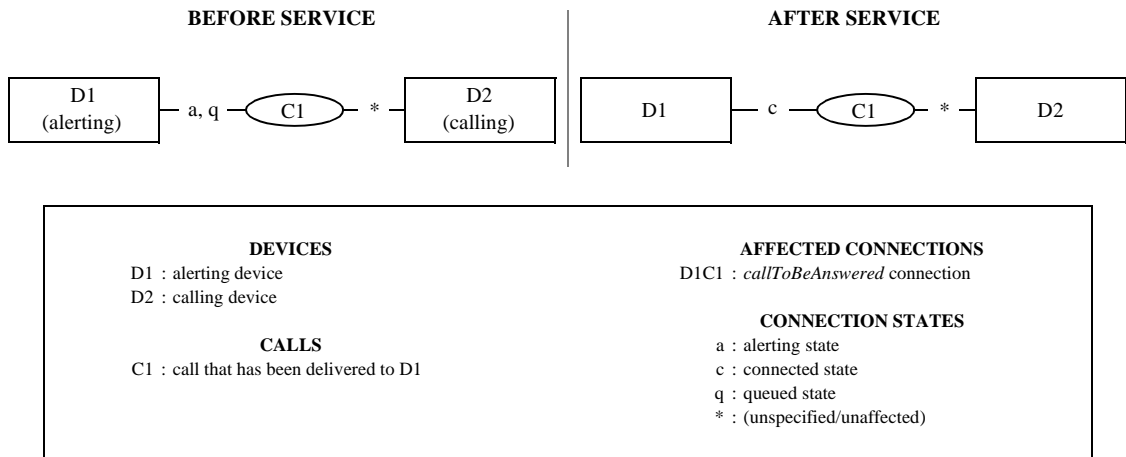
The following sections describe the Services template components.

11.1.1 Service Description

This is textual description of the service that may be followed by a figure. The figure is included when a service affects a connections state(s). The figure defines the role of devices and connections from a before/after service execution perspective. Note that this figure indicates the successful completion of the service but does not indicate the service completion criteria (see the Monitoring Event Sequences for the service completion criteria).

In order to describe the nomenclature used in the figures in the templates, the figure from the CSTA Answer Call service follows:

Figure 11-1 Example of a Figure in a CSTA Service (Answer Call)



In the figures, small boxes (labeled Dx) are used to represent devices, lines represent connections, ovals (labeled Cx) represent calls, and dotted lines represent connections with partial connection identifiers (see 6.7.2). The legend (large box) associates names with devices and calls. Names in italics refer to parameter names used in the service.

The connections are labeled with the set of possible connection states. In some cases the following symbols are used in place of a specific connection state:

“*” indicates that the connection state is not specified and it is not affected by the service

“!” indicates that the connection state is unspecified but may be affected by the service

“#” indicates that the connection state is not specified but the connection state is inherited from a connection that used to exist (for example, when a connection at a device changes its call identifier).

“@” indicates any non-Null connection state.

11.1.2 Service Request

This section contains a table with the possible parameters in the service request. Associated with each parameter are:

- Parameter Name (“Parameter Name”) - This is used to reference the parameter from other parts of the template and to distinguish the parameter from other parameters with the same parameter type. An example of a parameter name is connectionToBeCleared.
- Parameter Type (“Type”)- This is the parameter type as defined in Clause 12, “Parameter Types”, on page 57. In most cases a parameter type references a parameter type defined in either 12.2, “Defined Parameter Types”, on page 58 (CorrelatorData, for example) or 12.3, “Identifier Parameter Types”, on page 71 (ConnectionID, for example). In other cases the parameter type may be a Boolean, Value, Enumerated, etc. Refer to 12.1, “Definitions”, on page 57 for more information.
- Parameter Optionality (“M/O/C”) - Indicates whether the parameter must be included (M for mandatory), if the parameter is optional (O), or if the parameter is conditional (C). If a parameter is conditional, then there are specific requirements when the parameter must be supported. These requirements are described in the parameter description column.
- Parameter Description (“Description”) - This is a brief description of the parameter in the context of the service. A description of the parameter in the context of its parameter type can be found with its parameter type description in Clause 12, “Parameter Types”, on page 57.

11.1.3 Service Response

This section includes:

- a description of the type of acknowledgement model that can be used with the service (see 9.2.1, “Positive Acknowledgement Models”, on page 47).
- a table that contains all of the parameters in the positive acknowledgement. The format of the table is the same as in the service request (see 11.1.2).
- a reference to the negative acknowledgement error codes.

11.1.4 Operational Model

The operational model consists of:

Connection State Transitions - This is a table with all possible connections affected by the service. Associated with each connection is the:

- Connection Name (“Connection”) - This is used to reference the connection from other parts of the template including the figure in the service description.
- Initial State (“Initial State (Required)”) - This is the set of allowed initial states (connection states before the service is executed). An implementation shall support one or more of the specified initial states associated with a service (as indicated in the capability exchange services).
- Final State (“Final State”) - This is the set of allowed final states (connection states after the service is executed). An implementation shall support one or more of these states. In many cases there are statements following the connection state transition table that further describe or clarify the information in the table.

Monitoring Event Sequences - For services that affect connections, this section includes tables that describes the event sequence generated for device-type and call-type monitors. Unless otherwise specified, the events (and associated causes) in this table are required as part of the service completion criteria. Each table contains:

- Monitored Device or Monitored Call (“Monitored Device” or “Monitored Call”) - For the device-type monitoring table, this indicates the monitored device. For the call-type monitoring table, this indicates the monitored call. The names can be used to reference back to the figure in the service description.

- Connection Name (“Connection”) - This column indicates the connection that is the subject of the event.
- Event (“Event”) - This is the name of the event generated as the result of the service.
- Event Cause (“Event Cause”) - This is the set of possible cause codes associated with the event. In many cases there are statements following the connection state transition table that further describes or clarifies the information in the table

Functional Requirements - The functional requirements contain additional requirements associated with the service.

11.2 Event Template

The following sections describe the Event Template components.

11.2.1 Event Description

This is textual description of the event followed by an optional figure. The figure is included when an event indicates a change in one or more connections. The figure defines the role of devices and connections from a before/after perspective. The nomenclature used in the figures is described in 11.1.2, “Service Request”.

11.2.2 Event Parameters

This section consists of a table that contains all of the parameters in the event. The format of the table is the same as in the service request table described in 11.1.2, “Service Request”.

11.2.3 Event Causes

This section consists of a table that contains all of the possible cause codes that can be included with the event. Associated with each cause code are:

- Event Cause (“Event Cause”) - This is the event cause name.
- Event Description (“Description”) - This is a description of the event cause in the context of the event.
- Associated Features (“Associated Features”) - This is the complete set of possible features associated with the cause code. A feature may either correspond to a CSTA service or it may be associated with switch features specified in 6.7, “Additional Services, Features & Behaviour”, on page 35.

11.2.4 Functional Requirements

Functional requirements contain additional requirements associated with the event.

11.3 Parameter Type Template

The following sections describe the Parameter Template components:

11.3.1 Parameter Type Description

This contains a description of the parameter type.

11.3.2 Format

This section specifies the format of the parameter type. For example, it could list the possible values associated with a parameter type (enumerated list).

For parameter types that are a type of device identifier, the format contains the allowed statuses associated with the parameter type (e.g. “Not Known”).

11.3.3 Functional Requirements

Functional requirements contain additional requirements associated with the parameter type.

12 Parameter Types

12.1 Definitions

This section describes the parameter types for all parameters described in the services and events sections of this specification. There are five sets of parameter types:

1. *Basic parameter types* are simple types that are not necessarily specific to these specifications. The basic parameter types used in these specifications are:
 - *Boolean* - Either TRUE or FALSE.
 - *Value* - Integer value with a length of 4 bytes always.
 - *Characters* - Character string of varying lengths, as specified in specific services or events.
2. *Meta parameter types* refer to constructions that combine one or more parameter types. The meta parameter types used in these specifications are:
 - *Bitmap* - Multiple values may be set in a specified set.
 - *Enumerated* - One value only may be set in a specified set.
 - *Structure* - A combination of different types combined into one parameter type, as defined in a specific service or event.
 - *List* - List of a single specified parameter type or structure
3. *Defined parameter types* are specific to these specifications. These are briefly defined in Table 12-1 on page 58, and further defined in the pages indicated.
4. *Identifier parameter types* are specific to these specifications. These are briefly defined in Table 12-2 on page 71, and further defined in the pages indicated.
5. *Capability bitmap parameter types* are bitmaps included in the Get Physical Device Information, Get Logical Device Information and Get Switching Function Capabilities services.

12.2 Defined Parameter Types

Defined parameter types specific to these specifications are summarized in the following table.

Table 12-1 Defined Parameter Types Summary

Defined Parameter Type	Description	Pg.
12.2.1 AccountInfo	Contains computing sub-domain/business specific code that is to be applied or has been applied to a call for accounting purposes.	59
12.2.2 AuditoryApparatusList	Specifies either a specific auditory apparatus or all auditory apparatuses associated with a physical device.	59
12.2.3 AuthCode	Contains an authorization code that the switching function understands and will use to check to see if the user of the computing sub-domain is authorized to perform the given service.	60
12.2.5 CallCharacteristics	Specifies the high level characteristics of the call.	61
12.2.6 ChargingInfo	Specifies the total value of charging or currency units.	61
12.2.7 ConnectionInformation	Specifies the connection information associated with the subject connection.	62
12.2.8 ConnectionList	Specifies the list of devices/connections that are known to the switching function, and which remain in the call after a conference or transfer.	62
12.2.9 CorrelatorData	Contains computing sub-domain-specific data that has been or will be attached to a call that the computing sub-domain is controlling or monitoring.	63
12.2.10 DisplayList	Specifies the attributes of either a specific display or all displays associated with a physical device.	64
12.2.11 ErrorValue	Contains hierarchical error codes.	65
12.2.12 EventCause (to be defined)	Provides additional information on why the event was generated.	66
12.2.13 LocalConnectionState	Describes the connection state of the device associated with the Monitor Cross Reference ID.	66
12.2.4 MediaCallCharacteristics	Specifies the media class (Voice, Digital Data, etc.) and media characteristics of the call.	60
12.2.14 CSTAPrivateData (to be defined)	Provides a mechanism for providing non-standard parameters in events.	67
12.2.15 CSTASecurityData (to be defined)	Specifies the security attributes associated with the message.	67
12.2.16 ServicesPermitted	Specifies the set of services that the switching function permits to be applied to a connection.	67
12.2.18 SimpleCallState	Provides the simple call state.	68
12.2.17 SystemStatus	Indicates the reason for the System Status service request.	68
12.2.19 UserData	Contains device-to-device or computing sub-domain-to-computing sub-domain data.	69

12.2.1 AccountInfo

The AccountInfo parameter type contains a computing sub-domain/business specific code that is to be applied or has been applied to a call for accounting purposes.

Format

This parameter type is a character string with a maximum length of 32.

Functional Requirements

1. The management of the account code data is done by the switching function. To understand how the switching function maintains this information with the call, you need to consult the switching function specific documentation.
2. The computing sub-domain will only be notified that the account code data has been added or changed through the Call Information event. This event will be generated when the user enters the account code data manually or after a service has added or changed the data.
3. The way to clear the data on the call is to pass a null string of data on one of the above mentioned services. (The actual parameter is passed, but the content is a null string.)
4. The switching function may choose to filter this information by not providing it in events for security reasons.
5. When this information is being attached to a call through a service request, its association shall be completed prior to any state transitions resulting from the request. Thus any state transition events which contain this parameter shall contain the information passed on the request.

12.2.2 AuditoryApparatusList

The AuditoryApparatusList parameter type specifies either a specific auditory apparatus or all auditory apparatuses associated with a physical device.

Format

This parameter type is a list of auditory apparatus entries. These entries are comprised of the following components:

1. auditoryApparatus (M)
2. auditoryApparatusType (M) - Specifies whether or not this auditory apparatus is a speakerphone, handset, headset or other type. Auditory apparatus type may have one of the following values:
 - speakerphone - Designates a logical hookswitch that is associated with a speakerphone.
 - handset - Designates a physical hookswitch associated with a typical telephone handset that is operated (i.e., “opened” and “closed”) by manually lifting the handset from, and replacing it in, a handset cradle.
 - headset - Designates a logical hookswitch associated with a headset.
 - speaker-only phone - Designates a hookswitch associated with a “phone” that has only a speaker.
 - other.
3. speaker (M) - For this auditory apparatus element, the following items are provided:
 - present - Indicates whether a speaker element is present in this auditory apparatus.
 - volumeSettable - Indicates whether the volume can be set by the switching function.
 - volumeReadable - Indicates whether the volume setting of this speaker can be read by the switching function.
 - muteSettable - Indicates whether the mute function for this speaker can be set by the switching function.
 - muteReadable - Indicates whether the current mute function setting can be read by the switching function.

4. microphone (M) - For this auditory apparatus element, the following items are provided:
 - present - Indicates whether a microphone element is present in this auditory apparatus.
 - gainSettable - Indicates whether the gain of this microphone can be set by the switching function.
 - gainReadable - Indicates whether the gain setting of this microphone can be read by the switching function
 - muteSettable - Indicates whether the mute function for this microphone can be set by the switching function.
 - muteReadable - Indicates whether the current mute function setting can be read by the switching function.
5. hookswitch (M) - For this auditory apparatus element, the following items are provided:
 - hookswitchSettable - Indicates whether the switching function can set the status of this hookswitch.
 - hookswitchOnHook - Indicates whether this auditory apparatus's hookswitch is currently on-hook or off-hook.
6. hookswitchID (M) - Indicates the ID of the hookswitch used by this auditory apparatus. (Note that the same hookswitch may be used by multiple auditory apparatuses.)

12.2.3 AuthCode

The AuthCode parameter type contains an authorization code that the switching function understands and will use to check if the computing function is authorized to perform a given service.

Format

This parameter type is a character string with a maximum length of 32.

Functional Requirements

1. If the switching function requires this parameter, and either the authorization code supplied is not valid or the parameter type is not supplied, then the service will be rejected with a negative acknowledgement.
2. The switching function may choose to filter this information by not providing it in events for security reasons.
3. When this information is being attached to a call through a service request, its association shall be completed prior to any state transitions resulting from the request. Thus any state transition events which contain this parameter shall contain the information passed on the request.

12.2.4 MediaCallCharacteristics

The MediaCallCharacteristics parameter type specifies the media (voice, digital data, etc.) characteristics of the call.

Format

This parameter type is comprised of the following:

1. mediaClass (M) Bit List - Specifies the media class (voice, digital data, etc.).
2. connectionRate (O) Value - The digital data connection rate of the call. The contents of this parameter is switching function specific (the capability exchange services may be used to obtain the list of possible values that are supported by the switching function). A value of zero (0) indicates that the type of media stream associated with the connection is digital data but the connection rate is unknown.
3. bitRate (O) Enumerated - The digital data bit rate of the call. If this parameter is not present, the bit rate of the call is a constant bit. The following is the complete set of possible values:
 - Constant (Default) - A bit rate which ensures a dedicated bandwidth and a constant rate of media stream delivery.
 - Variable - A bit rate which may variable during the life of the call.

4. delayTolerance (O) Value - The digital data delay tolerance of the call. This parameter specifies the maximum amount of media stream delivery delay that will be tolerated for the call. If the bit rate is constant, then this value will indicate the actual amount of media stream delivery delay for the life of the call. Where as if the bit rate is variable, it will be the maximum delay allowed during the life of the call. The contents of this parameter is switching function specific, use the capability exchange services to obtain the list of possible values that are supported by the switching function. If this parameter is not present, the delay tolerance of the call is not known.
5. switchingSubDomainCCIEType (O) Enumerated - The type of switching sub-domain private call control information elements that are present in the switchingSubDomainInformationElements parameter. If this parameter is not present, there are no information elements associated with the call and the switchingSubDomainInformationElements parameter should be ignored. The following is the complete set of possible values:
 - ISDN
 - ATM (B-ISDN)
 - ISO-Ethernet (TDM part only)
 - RSVP
 - Other (switching sub-domain specific)
6. switchingSubDomainInformationElements (C) Characters - These parameters contain the private information elements that are available from the switching sub-domain (as specified by switchingSubDomainCCIEType) which represents a specific set of information elements. The format, meaning and behaviour of these information elements are specific to the given switching function. This parameter is only present and mandatory when the switchingSubDomainCCIEType parameter is present.

12.2.5 CallCharacteristics

The CallCharacteristics parameter type describes, when included on an event, the high level characteristics associated with a call.

When this parameter is included on a switching function service request, it indicates the requested set of high level characteristics that should be associated with the call.

Format

This parameter type is a bitmap. Multiple bits may be set. The complete set of possible values in the bitmap is:

- ACD call. This bit is set to indicate an ACD call. Once the call is no longer associated with the ACD, this bit is no longer set. See Functional Requirement #1.
- Priority call - This bit is set to indicate a priority call.
- Maintenance call - This bit is set to indicate a maintenance call.

Functional Requirements

1. There are many conditions when a switching function may classify a call as an ACD call and when an ACD call becomes a non-ACD call. The specific conditions are switching function dependent.

12.2.6 ChargingInfo

The ChargingInfo parameter type represents a cumulative value of charging or currency units charged to a device for a call in which the device was involved. This information can represent an intermediate (during the call) or final total (when the device leaves the call).

Format

This parameter consists of the following components:

- numberUnits (M) - This component consists of one of the following:
 - numberOfChargingUnits - indicates a cumulative number of charging units. This component consists of a sequence that may be repeated to report different types of charging units. The sequence consists of:
 - chargingUnits (M) - the number of charging units.
 - typeOfUnits (O) - the type of units. This may be included to differentiate among these types. Its definition is network-dependent.
 - numberOfCurrencyUnits - indicates a cumulative value of currency units. If this component is provided, then an additional multiplier parameter (chargingMultiplier) shall be included that applies to the currency unit.
- Type of Charging Information (M) - This can have one of the following values:
 - Sub-total - indicates that the information is an intermediate value.
 - Total - indicates that the charging information is complete.
- chargingMultiplier (C) - indicates the currency unit multiplier. It shall be provided if the numberOfCurrencyUnits is provided, otherwise it shall not be provided. The complete set of possible values is:
 - .001
 - .01
 - .1
 - 1
 - 10
 - 100
 - 1000

12.2.7 ConnectionInformation

The ConnectionInformation parameter type specifies the connection information associated with the subject connection (i.e., the connection that is the focus of the event or positive acknowledgement being reported).

Format

This parameter type is comprised of the following parameters:

1. flowDirection (O) - Specifies the direction of flow that is associated with the subject connection. If this parameter is not present, the connection's flow direction is unknown. The complete set of possible values is:
 - Transmit - Media stream data is only capable of being transmitted on the connection by the associated device.
 - Receive - Media stream data is only capable of being received on the connection by the associated device.
 - Transmit and Receive - Media stream data is capable of being transmitted and received on the connection by the associated device.
2. numberOfChannels (O) - Specifies the number of media stream channels that are associated with the subject connection. If this parameter is not present, the number of channels associated with the connection is one.

12.2.8 ConnectionList

The ConnectionList parameter type provides the linkage mechanism between a device's old connection ID and new connection ID resulting from the conference or transfer.

Format

This parameter includes the following components for every device or connection being reported:

- new ConnectionID (C) - The CallID portion of this ConnectionID refers to the resulting call. This component is optional for the transferringDevice in the Transferred event, otherwise it is mandatory.
- old ConnectionID (C) - The CallID portion of this ConnectionID refers to the original call. This component is mandatory if the switching function previously reported the CallID, otherwise it is optional.
- endPoint DeviceID (O) - For internal calls, this is the representation of the device inside the switching sub-domain. For external calls (incoming or outgoing), this is the representation of the externally located device (if known by the switching function). This component is a character string. The maximum length supported by the switching function is provided via the capabilities exchange services. It may be:
 - of any device identifier format
 - of the following statuses: “Provided” or “Not Known”
- associatedNID (C) - For external calls (incoming and/or outgoing), this component specifies the Network Interface Device (e.g., trunk, CO line) within the switching sub-domain that is associated with the externally located device. In that case the component endPoint DeviceID (if provided) shall represent the externally located device. The associatedNID component is mandatory in case of external calls and shall be omitted when the device is located inside the switching sub-domain. This component may be:
 - of any device identifier format
 - of the following statuses: “Provided” or “Not Known”
- resulting ConnectionInformation (O) - This component contains the flow direction and channel characteristics associated with the resulting connection.

Functional Requirements

1. This list should be used by the computing function to associate devices which remain in a call, as a result of a Conference or Transfer, with the connection IDs that are used to manipulate them.

12.2.9 CorrelatorData

The CorrelatorData parameter type contains computing sub-domain specific data that has been or will be attached to a call that the computing function is controlling or monitoring. This allows the computing function to associate its own information with a call and, as a result, share it with other computing functions. For example, this information might be a key to a database entry, a computing function command sequence, file name, etc. This feature is useful when calls are moving from one computing function to another in a distributed computer network or from one switching sub-domain to another.

See 6.1.4.3, “Correlator Data”, on page 22 for specific rules on the use of Correlator Data.

This specification defines a mechanism for delivering both user data and correlator data through an external ISDN network at the same time. This mechanism is described in Annex D.

Format

This parameter type is an octet string. The maximum length supported by the switching function is provided via the capabilities exchange services, but is limited to 32.

Functional Requirements

1. The correlator data will stay with the call as long as the call exists. This means that the correlator data will be presented to the computing function on events that have this parameter and that the switching sub-domain supports.
2. The correlator data can be changed during the life of the call by any of the services that has the parameter or by using the Associate Data service.

3. The way to clear the data on the call is to pass a null string of data on one of the above mentioned services. (The actual parameter is passed but the content is a null string.) If correlator data is cleared, then the switching function notifies the computing function by sending a null string.
4. See 6.1.4.3, "Correlator Data", on page 22 for a description of how Correlator Data is inherited by calls during a conference or transfer.
5. If the computing function issues the Consultation Call service without correlator data, initially the secondary call will not have correlator data associated with it, as it does not inherit any correlator data that may be associated with the primary call. If the computing function issues the Consultation Call service with correlator data, this data is for the secondary call only and does not affect any correlator data that may be currently associated with the primary call.
6. When correlator data is associated with a call, for all Call Control events listed in 17.2 on page 192 *except* for the Bridged, Call Cleared, Connection Cleared, Held, and the Retrieved events (i.e. call events that may indicate that a device becomes part of a call) shall include the correlator data (if supported). Correlator data can optionally be included with the four event exceptions listed above.
7. When this data is being attached to a call through a service request, its association shall be completed prior to any state transitions resulting from the request. Thus any state transition events which contain this parameter shall contain the information passed on the request.
8. If a computing function issues a Snapshot Call service after a service request has been issued with this information, but prior to the switching function making any state transitions, it is switching function specific as to what will be returned in the positive acknowledgement with regards to this parameter.

12.2.10 DisplayList

The DisplayList parameter type specifies the attributes of either a specific display or all displays associated with a physical element of a device.

Format

This parameter type is a list of display entries. These entries are comprised of the following components:

1. displayID (C) - This parameter indicates the display to which the other provided information applies. When there is one display this parameter may be omitted. When there is more than one display this parameter shall be present.
2. logicalRows (M) - The number of rows on the logical display.
3. logicalColumns (M) - The number of columns on the logical display.
4. physicalRows (C) - The number of rows on the physical display. When the number of physicalRows is equal to the number of logicalRows this parameter shall be omitted, otherwise it shall be present.
5. physicalColumns (C) - The number of columns on the physical display. When the number of physicalColumns is equal to the number of logicalColumns this parameter shall be omitted, otherwise it shall be present.
6. physicalBaseRowNumber (C) - The row number of the physical base, i.e. the logical row that appears at the first row of the physical display. When the number of physicalRows is equal to the number of logicalRows this parameter shall be omitted, otherwise it shall be present.
7. physicalBaseColumnNumber (C) - The column number of the physical base, i.e. the logical column that appears at the first column of the physical display. When the number of physicalColumns is equal to the number of logicalColumns this parameter shall be omitted, otherwise it shall be present.
8. characterSet (O) - Specifies the character set which is being used to represent the text on the display. The complete set of possible values is:
 - ASCII (default)

- Unicode
 - Proprietary
9. contentsOfDisplay (M) - Specifies the text on display as a string of characters consisting of the text on each row of the display (including spaces) concatenated together.

12.2.11 ErrorValue

The ErrorValue parameter type defines error codes.

Format

This parameter contains the following values:

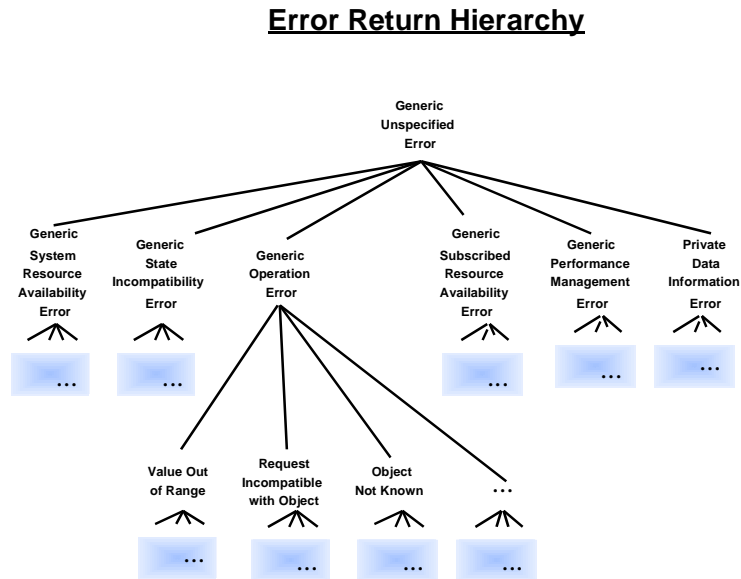
- *Category* - Generic, System Resource Availability, State Incompatibility, Operation, Subscribed Resource Availability, Performance Management, or Private Data.
- *Value* - A value describing the error. The following text describes the various categories and values within each category.

Values

Negative acknowledgements to service requests contain error codes. This standard uses the following definitions across all error codes in all services to ensure a uniform meaning for error codes.

- Error codes reflect why the switching function could not carry out the service request on the specified call, device, or connection and do not reflect the status of any other call, device or connection.
- Error codes reflect why the switching function could not perform the request at the time that it attempted to execute the request. Thus a switching function will return the same error code in the same circumstance regardless of the past history of any object involved in the request.
- This specification does not require that service parameters be checked in any order. Thus, where there are multiple errors in parameters (or multiple errors apply to a single parameter), the computing function may receive any of the applicable errors.
- There is a hierarchy of error return values. The errors range from one very high level error that spans all errors (GENERIC_UNSPECIFIED) to detailed errors that mean only one specific thing. The diagram below shows the hierarchy. The errors become more detailed toward the bottom of the diagram.

Figure 12-1 ErrorValue Hierarchy



12.2.12 EventCause (to be defined)

12.2.13 LocalConnectionState

The LocalConnectionState parameter type describes the connection state of the device associated with the Monitor Cross Reference ID.

This parameter type is only applicable for events generated by device-type monitors.

Format

This parameter shall contain one of the following connection states:

- Alerting
- Connected
- Fail
- Hold
- Initiated
- Null
- Queued

Refer to 6.1.5, “Connection”, for detailed descriptions of the connection states.

Example

The following is a scenario that illustrates the usage of this parameter.

Consider the case of a two device call where device one has called device two, and device two is ringing. If both devices are monitored, then the switching function generates two separate (Delivered) events to indicate that the call has been delivered. While the subject device is identical for both Delivered events, the localConnectionInfo parameters are different.

- Both events contain the same subject device, device two in this case, since that is the device in the call being alerted.
- The connection state for device one is connected (most likely listening to ringback). This is reported in the localConnectionInfo parameter of the Delivered event for device one.
- The connection state for device two is alerting. This is reported in the localConnectionInfo parameter of the Delivered event for device two.

12.2.14 CSTAPrivateData (to be defined)

12.2.15 CSTASecurityData (to be defined)

12.2.16 ServicesPermitted

The ServicesPermitted parameter type specifies the set of services that the switching function permits to be applied to a connection.

The servicesPermitted parameter (when provided in a call event) is similar to the localConnectionInfo parameter in that it applies to the services permitted for the connection at the monitored device.

This parameter type is only applicable for events generated by device-type monitors.

Format

This parameter type is a bitmap where each bit represents a service that can be applied to a connection. When a bit is set, the corresponding service is permitted. The following is the list of bits (multiple bits may be set in this parameter):

- Accept Call
- Alternate Call
- Answer Call
- Call Back Call-Related
- Call Back Message Call-Related
- Camp On
- Clear Call
- Clear Connection
- Conference Call
- Consultation Call
- Deflect Call
- Dial Digits
- Directed Pickup Call
- Group Pickup Call
- Hold Call
- Intrude Call
- Join Call
- Make Call
- Make Predictive Call
- Park Call
- Reconnect Call
- Retrieve Call

- Single Step Conference Call
- Single Step Transfer Call
- Transfer Call

Functional Requirements

1. This parameter indicates which of a subset of CSTA services are permitted.
2. When the `servicesPermitted` parameter is provided in an event, it applies to the connection at the monitored device. This may or may not be the same as the subject device.
3. If there are multiple connections at a device, the information reported in the `servicesPermitted` parameter may not accurately reflect all possible service restrictions and interactions between multiple connections at a device.
4. There may be situations in a switching function that cause a service to fail after being presented as permitted in the `servicesPermitted` parameter. This may be due to dynamic system and/or resource conditions that may cause service availability restrictions. The switching function shall provide the appropriate error code in the negative acknowledgement to the failed service request.

12.2.17 SystemStatus

The `SystemStatus` parameter type indicates the reason for the System Status service request.

Format

This parameter type is an enumeration. The following values are possible:

- *Disabled* - Existing Monitor Requests have been disabled. Other requests and acknowledgements also may be disabled, but negative acknowledgements should always be provided.
- *Partially Disabled* - Some of the objects in the system can not be reached. Existing monitors on these objects will not provide events and computer requests targeting these objects will be rejected. This cause indicates to the receiving function that a degradation of service level may occur but not complete system disability. Automatic or manual actions may be taken to remedy the parts disabled.
- *Enabled* - Requests and acknowledgements have been enabled. This usually occurs after a disruption or restart. This status cause is always sent after an *Initializing* cause has been sent and may be sent under other conditions. This status indicates that there are no outstanding monitors (existing monitors and their associated monitor cross reference identifiers are no longer valid).
- *Initializing* - The system is initializing or restarting. This status indicates that a system is temporarily unable to respond to any requests. If provided, this status message is followed by an *Enable* status message to indicate that the initialization process has completed.
- *Messages Lost* - Requests and/or acknowledgements, including event reports, may have been lost.
- *Normal* - May be sent at any time and indicates that the status is normal. This status has no effect on other Services.
- *Overload Imminent* - The receiver is requested to take initiative to shed load.
- *Overload Reached* - The requester may take initiative to shed load. This cause may be followed by *Stop Monitor* requests sent to the client and by rejections to additional service requests.
- *Overload Relieved* - The overload condition has passed.

12.2.18 SimpleCallState

The `SimpleCallState` parameter type indicates the main call states in simplified encoding. The semantics are identical to the sequence of connection states but they are represented by an item from an enumerated list.

Format

This parameter type is an enumeration. The following values are possible:

- callNull
- callPending
- callOriginated
- callDelivered
- callDeliveredHeld
- callReceived
- callEstablished
- callEstablishedHeld
- callReceivedOnHold
- callEstablishedOnHold
- callQueued
- callQueuedHeld
- callFailed
- callFailedHeld
- callBlocked

12.2.19 UserData

The UserData parameter type contains computing sub-domain to computing sub-domain data. Note that the capabilities exchange services return the maximum length of the user data for a switching function.

User Data is described further in 6.1.4.4, “User Data”, on page 23. Also, refer to 12.2.9, “CorrelatorData”, on page 63.

This specification defines a mechanism for delivering both user data and correlator data through an external ISDN network at the same time. This mechanism is described in Annex D.

Format

This parameter type is an octet string. The maximum length supported by the switching function is provided via the capabilities exchange services, but is limited to 256 octets.

Functional Requirements

1. The ability to send User Data, the timing of when user data can be sent, and the size of user data, is dependent upon the switching function’s capabilities and the underlying network (such as ISDN).
2. Unlike correlator data, User Data is not attached to a call for the life of the call. User Data that has been associated with a primary or secondary call does not get retained with a resulting conference or transferred call.
3. The switching function reflects the delivery of User Data in the call control events that result from the switching function or network carrying out the call control activity with which the User Data was associated. When the switching function receives user data independent of call activity (i.e., Send User Information service), the User Data is provided in the Call Information event.
4. User data addresses a specific user in a call (e.g. the initially called device). The delivery and propagation of the user data to other devices inside the switching sub-domain in regards to features that apply to the call (e.g. forwarding, do not disturb) is switching function dependent.

5. When this data is being attached to a call through a service request, its association shall be completed prior to any state transitions resulting from the request. Thus any state transition events which contain this parameter shall contain the information passed on the request.
6. If a computing function issues a Snapshot Call service after a service request has been issued with this information, but prior to the switching function making any state transitions, it is switching function specific as to what will be returned in the positive acknowledgement with regards to this parameter.

12.3 Identifier Parameter Types

Identifier parameter types specific to these specifications are summarized in the following table.

Table 12-2 Identifier Parameter Types Summary

Defined Parameter Type	Description	Pg.
12.3.1 AgentID	Identifies an ACD agent.	72
12.3.2 AssociatedCalledDeviceID	Describes the switching function's internal representation of the originally called device in a call.	72
12.3.3 AssociatedCallingDeviceID	Describes the switching function's internal representation of the calling device in the call when the calling device is outside the switching sub-domain (i.e., trunk number).	72
12.3.4 AuditoryApparatusID	Indicates the auditory apparatus containing the speaker whose volume has changed.	73
12.3.5 ButtonID	Specifies the button identifier on a device.	73
12.3.6 CalledDeviceID	Specifies the device to be called via a service. This parameter describes the originally called device associated with a call.	73
12.3.7 CallingDeviceID	Describes the calling device associated with the call.	73
12.3.8 ConnectionID	Describes a device's connection in a given call.	74
12.3.9 DeviceID	Identifies or represents a device in the switching function.	75
12.3.10 DisplayID	Specifies the display identifier on a device.	76
12.3.11 EscapeRegisterID	Used to identify and escape service registration.	76
12.3.12 HookswitchID	Used to specify the hookswitch to query at a specified device.	76
12.3.13 LampID	Specifies the lamp identifier.	76
12.3.14 MediaServiceInstanceID (to be defined)	Identifies a particular media access service instance (e.g., specific media access server or subsystem)	76
12.3.15 MediaStreamID (to be defined)	Specifies a media stream identifier that can be used to access an attached media service.	76
12.3.16 MonitorCrossRefID	Specifies an identifier that is used to correlate an event to an established monitor.	76
12.3.17 NetworkCalledDeviceID	Specifies the called device information provided by the network for external incoming calls.	76
12.3.18 NetworkCallingDeviceID	Specifies the calling device information provided by the network for external incoming calls.	77
12.3.19 RedirectionDeviceID	Describes the last device known by the switching function from which the current call was routed.	77
12.3.20 RingerID	Specifies the ringer identifier associated with a physical device	78
12.3.21 RouteingCrossRefID	References the routeing dialogues initiated by the switching function within a routeing registration.	78
12.3.22 RouteRegisterReqID	Identifies a routeing registration for which the computing function (acting as a routeing server) will receive routeing requests.	78
12.3.23 ServiceCrossRefID	Specifies an identifier that is used to correlate one service request to another service request.	79
12.3.24 SubjectDeviceID	Describes the device where a telephony event occurred or was invoked.	79
12.3.25 SysStatRegisterID	Used to identify system status registration.	79

12.3.1 AgentID

The AgentID parameter type identifies an ACD agent.

Format

This parameter type is an octet string. The maximum length supported by the switching function is provided via the capabilities exchange services.

12.3.2 AssociatedCalledDeviceID

For outgoing external calls, the AssociatedCalledDeviceID parameter type specifies the Network Interface Device (e.g., trunk, CO Line) within the switching sub-domain that is associated with the originally called device. This parameter is mandatory on all events dealing with external outgoing calls.

For incoming external calls, this parameter specifies a device within the switching sub-domain that is associated with the originally called device (such as a switching function internal representation of DNIS, for example). This parameter is optional on all events dealing with incoming external calls.

Format and Status

This device identifier type may be:

- Of any format.
- Of the following statuses: “Provided”, “Not Known”

Functional Requirements

1. A device identifier of this type will only be present when the switching sub-domain is using a network interface device for an external call; that is, the call is an External Outgoing or External Incoming call.
2. A device identifier of this type is not used to provide DNIS (Dialled Number Identification Service) or DID (Direct Inward Dialing) digit information or a string of digits that represents the called device. (This information is provided in the corresponding CalledDeviceID parameter.)
3. A device identifier of this type is set to “Not Known” when the switching function does not know the Network Interface Device associated with the original called device.
4. A device identifier of this type will never contain the value “Not Required” or “Not Specified”.

12.3.3 AssociatedCallingDeviceID

The AssociatedCallingDeviceID parameter type specifies the Network Interface Device (e.g., trunk, CO line) within the switching sub-domain that is associated with the calling device in the call if the call is an external incoming call. This parameter shall be included on all external incoming calls.

Format and Status

This device identifier type may be:

- Of any format.
- Of the following statuses: “Provided”, “Not Known”

Functional Requirements

1. A device identifier of this type will only be present when the switching function is using a network interface device for an inbound call; that is, the call is an external incoming call.
2. A device identifier of this type is not used to provide ANI (Automatic Number Identification), CLID (CallerID) or SID (Station Identification) digit information or a string of digits that represents the calling device. (This information is provided in the corresponding CallingDeviceID parameter.)

3. A device identifier of this type will never contain the value “Not Required” or “Not Specified”.
4. If a call is created that contains multiple AssociatedCallingDeviceIDs (i.e., a conference call calling back to a device), the AssociatedCallingDeviceID status shall be “Not Known”.

12.3.4 AuditoryApparatusID

The AuditoryApparatusID parameter type specifies a particular auditory apparatus associated with the device.

Format

This parameter type is an octet string with a maximum length of four.

12.3.5 ButtonID

The ButtonID parameter type specifies the button identifier on a device.

Format

This parameter type is an octet string with the maximum length of four.

Table 12-3 Reserved Button ID Assignments

Button ID	Button Label
0-9	Keypad Digits: “0” through “9”
10	Keypad Symbol: “*”
11	Keypad Symbol: “#”

12.3.6 CalledDeviceID

A device identifier of the CalledDeviceID type describes the originally called device associated with a call.

Format and Status

This parameter type is a character string. The maximum length supported by the switching function is provided via the capabilities exchange services.

This device identifier type may be:

- Of any device identifier format.
- Of the following statuses: “Provided” and “Not Known”.

Functional Requirements

1. A device identifier of this type contains the originally called device in the call. For External Incoming calls, a device identifier of this type will contain DNIS (Dialed Number Identification Service) or DID (Direct Inward Dialing).
2. This parameter will never contain the value “Not Required” or “Not Specified”.
3. When two calls are being joined through a conference or transfer, the CalledDeviceID information for the resulting call shall be taken from the secondary call.
4. This parameter type is different from the NetworkCalledDeviceID parameter type in that the CalledDeviceID information may change if the call is transferred and/or conferenced whereby the NetworkCalledDeviceID information does not change as long as the NID associated with the original calling device remains in the call. Also, the NetworkCalledDeviceID is limited to information passed over a Network Interface Device.

12.3.7 CallingDeviceID

The CallingDeviceID parameter type specifies the calling device associated with the call.

Format and Status

This parameter type is a character string. The maximum length supported by the switching function is provided via the capabilities exchange services.

This device identifier type may be:

- Of any device identifier format.
- Of the following statuses: “Provided” and “Not Known”.

Functional Requirements

1. A device identifier of this type contains the calling device in the call. For External Incoming calls, a device identifier of this type will contain ANI (Automatic Number Identification), CLID (CallerID) or SID (Station Identification) digit information or a string of digits that represents the calling device.
2. This parameter will never contain the value “Not Required” or “Not Specified”.
3. If more than one device is the calling device in a call (i.e., a conference call calling back to a device), the CallingDeviceID status will be “Not Known”.
4. This parameter type is different from the NetworkCallingDeviceID parameter type in that the CallingDeviceID information may change if the call is transferred and/or conferenced whereby the NetworkCallingDeviceID information does not change as long as the NID associated with the original calling device remains in the call. Also, the NetworkCallingDeviceID is limited to information passed over a Network Interface Device.

12.3.8 ConnectionID

The ConnectionID parameter type describes a device’s connection in a given call. (Connection Identifiers are also discussed in 6.1.5, “Connection”, on page 24.)

Format

The ConnectionID is always comprised of the following parameters (except in special cases which are described below):

1. callID (M) - An identifier used by the switching function to represent a valid call. The maximum length of this ID is eight octets. These IDs are created by the switching function and are globally unique among all calls within the switching sub-domain.
2. deviceID (M) - An identifier which is used to represent a device in the switching sub-domain. This identifier can be either one of the two following values:
 - *Static* - This type of identifier is defined in 6.1.3, “Device”, on page 6.
 - *Dynamic* - This type of identifier is one that is created by the switching function for a device when it enters into a call and shall remain constant for the life of the device’s participation in the call (i.e., the creation of a connection identifier for the device). As soon as the device leaves the call, the identifier becomes invalid. The use of a dynamic identifier by a switching function is determined when the switching function does not have a static identifier for the device or the identifier can not uniquely identify the device in a call. This type of identifier is an octet string, with a maximum length of 32. It is never a diallable number and can never be used outside the context of the connection identifier. This type of identifier is not directly related to a device element but is strictly used to make the connection identifier unique. Refer to 6.1.8, “Management of Dynamically-Assigned Identifiers”, on page 30, for more information.

Functional Requirements

1. The computing function shall not fabricate its own Connection IDs. This will lead to unpredictable results.

2. The Connection IDs in events and service acknowledgements are always allocated by the switching function.
3. Computing functions can extract Device IDs from Connection IDs and use them on services that have Device ID parameters only if the Device ID extracted is a static Device ID that the switching function accepts. Otherwise, the Device ID cannot be used.
4. Computing functions shall extract Call IDs from Connection IDs, provided by the switching function, to correlate event reports associated with devices that are connected together in a call.
5. The computing function will always receive an event to indicate the termination of a Connection ID if the appropriate monitor is started. Refer to the individual services and events to better understand the meaning of individual events with respect to connection states.
6. If the computing function issues a service with a Connection ID that cannot be controlled by the switching function, the service will be rejected with a negative acknowledgement.
7. Connection IDs used as parameters can only have three formats:
 - a. A *complete* Connection ID (i.e., call ID and device ID). This extracted from either events received by the computing function or positive acknowledgements received as a result of services issued.

When supplied as a parameter, the Connection ID will be validated by the switching function with respect to the service being issued. If this Connection ID is not valid, the service request will be rejected with a negative acknowledgement.
 - b. A *DeviceID only* Connection ID. If a service has more than one Connection ID parameter, the switching function supports this type of Connection ID, and the computing function wants to use this type of Connection ID, then all Connection ID parameters in the service shall be of this type.

If this type of Connection ID is used as the Connection ID parameter for a service, then rules documented in the services sections will determine whether it is accepted or not by the switching function. If this type of Connection ID is not accepted, then the service will be rejected with a negative acknowledgement.
 - c. A *Call ID only* Connection ID. In events, this format can only be used for the Call Cleared and Failed event. If this format is used for any service other than Clear Call, Monitor Start, or Snapshot Call, it will be rejected with a negative acknowledgement.
8. If a call changes its Call ID when a Conference or Transfer occurs, Connection IDs shall be provided to link the old Call IDs to the new Call IDs. When this occurs, the event will contain a list of originally known Connection IDs of devices that are still in the call along with the new replacement Connection IDs. When the new Connection IDs are created in such cases, new dynamic Device IDs may also be used to create the Connection IDs.
9. Connection IDs that come from the switching function (events and positive acknowledgement to services) will always contain both the Call ID and Device ID portions (see item 7a above) except for the Call Cleared and Failed events that may also contain only a valid call ID in the connection ID (see item 7c above).
10. The computing function should never assume the reuse of callIDs, although some switching functions may reuse one or the other.

12.3.9 DeviceID

The DeviceID parameter type identifies or represents a device.

Format

The maximum length supported by the switching function is provided via the capabilities exchange services.

This device identifier type may be:

- Of any device identifier format. (See Clause 10, “CSTA Device Identifier Formats”, on page 50)

Functional Requirements

1. For more details on DeviceID parameter types, refer to 6.1.3, “Device”.
2. For information on DeviceID in Connection Identifiers, refer to 12.3.8, “ConnectionID”, on page 74 and 6.1.5, “Connection”, on page 24.

12.3.10 DisplayID

The DisplayID parameter type specifies a particular display associated with the device.

Format

This parameter type is a string with the maximum length of four characters.

12.3.11 EscapeRegisterID

The EscapeRegisterID parameter type is used to identify an escape service registration.

Format

This parameter type is an octet string with the maximum length of four.

12.3.12 HookswitchID

The HookswitchID parameter type is used to specify the hookswitch to query at a specified device. If not provided, the default is to get the status of each hookswitch at the specified device.

Format

This parameter type is an octet string with the maximum length of four.

12.3.13 LampID

The LampID parameter type specifies the lamp identifier.

Format

This parameter type is an octet string with the maximum length of four.

12.3.14 MediaServiceInstanceID (to be defined)

12.3.15 MediaStreamID (to be defined)

12.3.16 MonitorCrossRefID

The MonitorCrossRefID parameter type specifies an identifier that is used to correlate an event to an established monitor. When a monitor is established using the Monitor Start service, a monitorCrossReferenceID parameter is returned as part of the positive acknowledgement message. This monitorCrossReferenceID parameter is included in every event for that specific monitor.

Format

This parameter type is an octet string with a maximum length of four.

Functional Requirements

1. This parameter is allocated by the switching function.
2. The switching function is responsible for providing unique monitorCrossReferenceID parameters over a specific service boundary.

12.3.17 NetworkCalledDeviceID

For external incoming calls, this parameter specifies the called device information that was provided by the Network over a Network Interface Device. For example, this may contain DNIS (Dialed Number Identification Service) or DID (Direct Inward Dialing) digit information or a string of digits that represents the called device.

This information is established when the call is first created and stays with the call as long as the Network Interface Device (NID) associated with the original calling device remains in the call, even if the call is transferred from the original called device, for example.

Format and Status

This parameter type is a character string. The maximum length supported by the switching function is provided via the capabilities exchange services.

This device identifier type may be:

- Of any device identifier format.
- Of the following statuses: “Provided” and “Not Known”.

Functional Requirements

1. This parameter will never contain the value “Not Required” or “Not Specified”.
2. This parameter type is different from the CalledDeviceID parameter type in that the CalledDeviceID information may change if the call is transferred and/or conferenced whereby the NetworkCalledDeviceID information does not change as long as the NID associated with the original calling device remains in the call. Also, the NetworkCalledDeviceID is limited to information passed over a Network Interface Device.

12.3.18 NetworkCallingDeviceID

For external incoming calls, this parameter specifies the calling device information that was provided by the Network over a Network Interface Device. For example, this may contain ANI (Automatic Number Identification), CLID (CallerID) or SID (Station Identification) digit information or a string of digits that represents the calling device.

This information is established when the call is first created and stays with the call as long as the Network Interface Device (NID) associated with the original calling device remains in the call, even if the call is transferred from the original called device, for example.

Format and Status

This parameter type is a character string. The maximum length supported by the switching function is provided via the capabilities exchange services.

This device identifier type may be:

- Of any device identifier format.
- Of the following statuses: “Provided” and “Not Known”.

Functional Requirements

1. This parameter will never contain the value “Not Required” or “Not Specified”.
2. This parameter type is different from the CallingDeviceID parameter type in that the CallingDeviceID information may change if the call is transferred and/or conferenced whereby the NetworkCallingDeviceID information does not change as long as the NID associated with the original calling device remains in the call. Also, the NetworkCallingDeviceID is limited to information passed over a Network Interface Device.

12.3.19 RedirectionDeviceID

The RedirectionDeviceID parameter type describes the last device known by the switching function from which the current call was routed. “Routed” includes forwarded from, diverted from, or redirected from.

Format and Status

The maximum length supported by the switching function is provided via the capabilities exchange services.

This device identifier type may be:

- Of any device identifier format. See Clause 10, “CSTA Device Identifier Formats”, on page 50.
- Of the following statuses:
 - “Provided” - indicates that the DeviceID of the last redirection device is provided
 - “Not Known” - indicates that the call has been redirected but the switching function cannot provide the DeviceID
 - “Not Required” - indicates that the current call has never been redirected during the existence of the call
 - “Not Specified” - indicates that the switching function cannot determine whether or not the call has ever been redirected

Functional Requirements

1. The information in a device identifier of this type will stay with the call until the call is established. If the call is routed multiple time before it is established, then the information in this parameter will be updated to the last known device from which the call was routed. If the call was redirected from a device, but the device identifier is unknown, “Not Known” shall be used. Depending on the capabilities of the switching function, the last known device for the call might be reflected by only one device identifier and in actuality the call might have been routed several times before arriving at the final destination.

Note that in the case of Immediate Forwarding, where forwarding is triggered *before* the call is delivered to a device, the lastRedirectionDevice in the event associated with the delivery of the call to a new device (after it was immediately forwarded) shall contain “Not Known”. Refer to 6.7.1, “Forwarding”, on page 36 for more information on this behaviour.

12.3.20 RingerID

Specifies the ringer identifier associated with a physical element.

A device can be associated with one or more ringers.

Format

This parameter type is an octet string with a maximum length of four.

12.3.21 RouteingCrossRefID

The routeingCrossRefID parameter type identifies each routeing dialogue. The computing function receives a routeingCrossRefID in each Route Request service request. The Route Request service initiates a routeing dialogue. The routeingCrossRefID is only valid for the duration of the routeing dialogue pertaining to a specific call.

The routeingCrossRefID is unique within the routeing registration (routeRegisterReqID). Some switching functions may provide the additional benefit of a unique routeing cross reference identifier across the entire switching sub-domain. This is also the case if routeing registration is not supported by the switching function.

Format

This parameter type is an octet string with a maximum length of four.

12.3.22 RouteRegisterReqID

The RouteRegisterReqID parameter type identifies a routeing registration for which the computing function (acting as a routeing server) will receive routeing requests. This identifier may be associated with a particular routeing device within the switching sub-domain or it may indicate that the computing sub-domain is the routeing server for all routeing devices within the switching sub-domain. When the computing function uses the Route Register service to register for routeing services, it receives a routeRegisterReqID in the positive acknowledgement sent by the

switching function. The routeRegisterReqID is only valid until the routing registration is ended by the computing function or switching function.

routeRegisterReqID parameters are unique across a given CSTA service boundary.

Format

This parameter type is a string with a maximum length of four.

12.3.23 ServiceCrossRefID

The ServiceCrossRefID parameter type specifies an identifier that is used to correlate one service request to another service request.

For example, a service may be specified to request information from a switching function using an asynchronous mechanism. In this case there would be a service request from the computing function requesting information. The switching function would return a ServiceCrossRefID in the positive acknowledgement to this request. The switching function would subsequently send messages in the form of Service Requests to the computing function that would contain the same ServiceCrossRefID that could be used to correlate the service request with the original service request.

Format

This parameter type is an octet string with the maximum length of four.

Functional Requirements

1. This parameter is allocated by the switching function.
2. The switching function is responsible for providing unique ServiceCrossRefIDs over a specific CSTA service boundary.

12.3.24 SubjectDeviceID

The SubjectDeviceID parameter type represents a device which is the focus of the action associated with the event being reported.

Format and Status

The maximum length supported by the switching function is provided via the capabilities exchange services.

This device identifier type may be:

- Of any device identifier format. See Clause 10, "CSTA Device Identifier Formats", on page 50.
- Of the following statuses: "Provided" and "Not Known".

12.3.25 SysStatRegisterID

The SysStatRegisterID parameter type is used to identify system status registration.

Format and Status

This parameter type is an octet string with the maximum length of four.

12.4 Capability Bitmaps Parameter Types (to be defined)

13 Capability Exchange Services

13.1 Services

Table 13-1 Capability Exchange Services Summary

Capability Exchange Service	Description	Pg.
13.1.1 Get Logical Device Information (to be defined)	Used by the computing function to obtain the current set of logical device information for a given device identifier.	80
13.1.2 Get Physical Device Information (to be defined)	Used by the computing function to obtain the current set of physical device information for a given device identifier.	80
13.1.3 Get Switching Function Capabilities (to be defined)	Used by the computing function to obtain the current set of capabilities for the entire switching function.	80
13.1.4 Get Switching Function Devices (to be defined)	Used by the switching function to obtain the devices that can be controlled and observed.	80
13.1.5 Switching Function Devices (to be defined)	Provides the actual list of devices that can be controlled and observed.	80

- 13.1.1 **Get Logical Device Information (to be defined)** C → S
- 13.1.2 **Get Physical Device Information (to be defined)** C → S
- 13.1.3 **Get Switching Function Capabilities (to be defined)** C → S
- 13.1.4 **Get Switching Function Devices (to be defined)** C → S
- 13.1.5 **Switching Function Devices (to be defined)** S → C

14 System Services

This clause consists of:

- System Registration services
- System services

NOTE

This clause describes System Services between the Switching Function and the Computing Function.

14.1 Registration Services

Table 14-1 System Registration Services Summary

System Registration Service	Description	Pg.
14.1.1 Change System Status Filter	Changes the system status filter options for a current system registration.	82
14.1.2 System Register	Registers the computing function for system services with the switching function.	84
14.1.3 System Register Abort	Indicates that the switching function has terminated a system registration.	87
14.1.4 System Register Cancel	Unregisters the computing function for system services with the switching function.	88

14.1.1 Change System Status Filter

C → S

The Change System Status Filter service is used by the computing function to change the filter options for a current system registration.

14.1.1.1 Service Request

Table 14-2 Change System Filter—Service Request

Parameter Name	Type	M/O/C	Description
sysStatRegisterID	SysStatRegisterID	M	Specifies the system registration identifier for which the status filter should be changed.
requestedStatusFilter	Bitmap	M	Specifies the requested System Status Types to be filtered (not sent) by the switching function. This parameter is a bitmap of the following values: <ul style="list-style-type: none"> • Initializing • Enabled • Normal • Messages Lost • Disabled • Partially Disabled • Overload Imminent • Overload Reached • Overload Relieved Multiple bits may be set.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.1.1.2 Service Response

This service follows the atomic acknowledgement model for this service request.

14.1.1.2.1 Positive Acknowledgement

Table 14-3 Change System Status Filter—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
actualStatusFilter	Bitmap	M	Specifies the actual set of System Status Types that will be filtered (not sent) by the switching function. This parameter is a bitmap with the same set of possible values as in the service request. The actualStatusFilter may differ from the requestedStatusFilter parameter in the service request (See Functional Requirement #2).
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.1.1.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

14.1.1.3 Operational Model

14.1.1.3.1 Connection State Transitions

There are no connection state changes due to this service.

14.1.1.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.1.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.1.3.4 Functional Requirements

1. The requestedStatusFilter parameter allows the computing function to choose the System Status Types for which no System Status service requests should be issued by the switching function. This parameter only applies to the System Status service. If the System Status service has not been requested for this system registration, then the switching function shall send a negative acknowledgement to the Change System Status Filter service request.
2. An implementation that does not support all System Status Types will nevertheless accept the Change System Status Filter service even if the requested filter cannot be provided. The service acknowledgement indicates the actual set of System Status Types that will be filtered. This means that the actual set of filtered types returned in the positive acknowledgement may include additional types to be filtered (or fewer types generated by the switching function) than those requested in the service request.

14.1.2 System Register

C → S

The System Register service is used by the computing function to register to receive system services from the switching function. The computing function may be required to register for system services before it can receive any system service requests from the switching function.

14.1.2.1 Service Request

Table 14-4 System Register—Service Request

Parameter Name	Type	M/O/C	Description
requestTypes	Bitmap	M	Specifies the system services that are being registered. This parameter is a bitmap of the following values: <ul style="list-style-type: none"> • System Status • Request System Status Multiple bits may be set.
requestedStatusFilter	Bitmap	C	Specifies the requested set of System Status Types to be filtered (not sent) by the switching function. This parameter is a bitmap of the following values: <ul style="list-style-type: none"> • Initializing • Enabled • Normal • Messages Lost • Disabled • Partially Disabled • Overload Imminent • Overload Reached • Overload Relieved Multiple bits may be set. This parameter is mandatory if the requestTypes parameter includes System Status, otherwise it shall not be provided.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.1.2.2 Service Response

This service follows the atomic acknowledgement model for this service request.

14.1.2.2.1 Positive Acknowledgement

Table 14-5 System Register—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
sysStatRegisterID	SysStatRegisterID	M	Specifies the system registration identifier for this registration.
actualStatusFilter	Bitmap	C	Specifies the actual set of System Status Types that will be filtered (not sent) by the switching function. This parameter is a bitmap with the same set of possible values as in the service request. The actual types filtered may differ from what was requested in the service request (See Functional Requirement #5). If the requestType parameter in the service request includes System Status, then this parameter is mandatory, otherwise it shall not be provided.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.1.2.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

14.1.2.3 Operational Model

14.1.2.3.1 Connection State Transitions

There are no connection state changes due to this service.

14.1.2.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.2.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.2.3.4 Functional Requirements

1. The sysStatRegisterID parameter returned in the positive acknowledgement is used to identify the registration over which system services will be sent. The sysStatRegisterID is also used when cancelling the system registration.
2. The number of simultaneous system registrations allowed is switching function dependent. When the limit is reached, subsequent System Register service requests shall result in negative acknowledgements from the switching function.
3. The requestTypes parameter specifies which system services are to be issued by the switching function to the registering computing function.
4. The requestedStatusFilter parameter allows the computing function to choose the System Status Types for which no System Status service requests should be issued by the switching function. If the System Status service is not being requested (i.e., in the requestTypes parameter), then the actualStatusFilter parameter does not apply and shall not be provided. The actual system status filter is provided in the actualStatusFilter parameter in the positive acknowledgement.
5. An implementation that does not support all System Status Types will nevertheless accept the System Register service even if the requestedStatusFilter cannot be provided. The service acknowledgement indicates the actual set of System Status Types that will be filtered. This means that the actual set of filtered types returned in the positive acknowledgement may include additional types to be filtered (or fewer types generated by the switching function) than what was requested in the service request.
6. If explicit registration is not supported, all system services (e.g., System Status, Request System Status) and System Status Types (e.g., Initializing, Enabled) supported by the switching function shall be provided to the computing functions. Note that the computing function *shall* be prepared to respond to a System Status service

request from the switching function in such cases (because it has no way of specifying that it should *not* receive such requests).

7. Note that if a computing function registers for system services it shall support the ability to respond to any switching function System Status service requests it may receive. In particular, for implicit registrations, a CSTA-conformant computing function shall always be able to support such requests from the switching function.

14.1.3 System Register Abort

S → C

The System Register Abort service is used by the switching function to asynchronously cancel an active system registration. This service invalidates a current systems status registration. There is no positive acknowledgement defined for this service.

14.1.3.1 Service Request

Table 14-6 System Register Abort—Service Request

Parameter Name	Type	M/O/C	Description
sysStatRegisterID	SysStatRegisterID	M	Specifies the system registration identifier for the system registration that was aborted.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.1.3.2 Service Response

There are no service completion conditions for this service.

14.1.3.2.1 Positive Acknowledgement

There is no positive acknowledgement defined for this service.

14.1.3.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

14.1.3.3 Operational Model

14.1.3.3.1 Connection State Transitions

There are no connection state changes due to this service.

14.1.3.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.3.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.3.3.4 Functional Requirements

1. The switching function may issue this service at any time when it can no longer maintain the system registration.
2. The computing function may send a negative acknowledgement to this service request, but no positive acknowledgement is defined.

14.1.4 System Register Cancel

C → S

The System Register Cancel service is used to cancel a previous system registration. This request terminates the system registration and the computing function receives no further system service requests for that system registration once it receives the positive acknowledgement to the System Register Cancel request.

14.1.4.1 Service Request

Table 14-7 System Register Cancel—Service Request

Parameter Name	Type	M/O/C	Description
sysStatRegisterID	SysStatRegisterID	M	Specifies the system registration identifier for which the system registration is to be cancelled.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.1.4.2 Service Response

This service follows the atomic acknowledgement model for this service request.

14.1.4.2.1 Positive Acknowledgement

Table 14-8 System Register Cancel—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.1.4.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

14.1.4.3 Operational Model

14.1.4.3.1 Connection State Transitions

There are no connection state changes due to this service.

14.1.4.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.4.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.1.4.3.4 Functional Requirements

1. The computing function shall continue to process outstanding system requests from the switching function until it receives a positive acknowledgement for the System Register Cancel service request. The switching function shall not send any further system requests for a registration once it has sent the positive acknowledgement.

14.2 Services

Table 14-9 System Services Summary

System Service	Description	Pg.
14.2.1 Request System Status	Request to query the system status of the function receiving the request (bi-directional).	90
14.2.2 System Status	Request that reports the status of the function issuing the request to the function receiving the request (bi-directional). The indicated status may or may not have changed since the last System Status request was issued.	92
14.2.3 Switching Function Capabilities Changed (to be defined)	Request that reports that information reported in the Get Switching Function service has changed.	93
14.2.4 Switching Function Devices Changed (to be defined)	Request that reports that information reported in the Switching Function Devices service has changed.	93

14.2.1 Request System Status

C ↔ S

The Request System Status service is used by the computing function or switching function to obtain (i.e., query) the system status of its peer function.

14.2.1.1 Service Request

Table 14-10 Request System Status—Service Request

Parameter Name	Type	M/O/C	Description
sysStatRegisterID	SysStatRegisterID	C	Specifies the system registration identifier associated with the system registration for this request. This parameter is mandatory if the switching function is issuing the request and supports system registration, and shall not be provided otherwise.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.2.1.2 Service Response

This service follows the atomic acknowledgement model for this service request.

14.2.1.2.1 Positive Acknowledgement

Table 14-11 Request System Status—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
systemStatus	Enumerated	M	Specifies the status of the function issuing the service request. The complete set of possible values is: <ul style="list-style-type: none"> • Initializing • Enabled • Normal • Messages Lost • Disabled • Partially Disabled • Overload Imminent • Overload Reached • Overload Relieved See 12.2.17, “SystemStatus”, on page 68 for a description of these values.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.2.1.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

14.2.1.3 Operational Model

14.2.1.3.1 Connection State Transitions

There are no connection state changes due to this service.

14.2.1.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.2.1.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.2.1.3.4 Functional Requirements

1. The `systemStatus` parameter in the positive acknowledgement provides the requesting function with information regarding the state of the overall system of the responding function. This information is important for proper system operation and should be processed accordingly. If the responding function has informed the requesting function that an overload condition is imminent then the requesting function should attempt to decrease the overall traffic to the responding function.

14.2.2 System Status

C ↔ S

The System Status service is used by the computing function or switching function to report its system status to its peer function. The indicated status may or may not have changed since the last System Status request was issued. This service can also be used to implement a heartbeat mechanism between the two functions.

14.2.2.1 Service Request

Table 14-12 System Status—Service Request

Parameter Name	Type	M/O/C	Description
sysStatRegisterID	SysStatRegisterID	C	Specifies the system registration identifier associated with the system registration for this request. This parameter is mandatory if the switching function is issuing the request and supports system registration, and shall not be provided otherwise.
systemStatus	Enumerated	M	Specifies the status of the function issuing the service request. The complete set of possible values is: <ul style="list-style-type: none"> • Initializing • Enabled • Normal • Messages Lost • Disabled • Partially Disabled • Overload Imminent • Overload Reached • Overload Relieved See 12.2.17, “SystemStatus”, on page 68 for a description of these values.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.2.2.2 Service Response

This service follows the atomic acknowledgement model for this service request.

14.2.2.2.1 Positive Acknowledgement

Table 14-13 System Status—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

14.2.2.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

14.2.2.3 Operational Model

14.2.2.3.1 Connection State Transitions

There are no connection state changes due to this service.

14.2.2.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.2.2.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

14.2.2.3.4 Functional Requirements

1. The `systemStatus` parameter in the request provides the responding function with information regarding the state of the overall system of the requesting function. This information is important for proper system operation and should be processed accordingly. If the requesting function has informed the other function that an overload condition is imminent then the responding function should attempt to decrease the overall traffic to the requesting function.
2. The computing function can determine if the switching function uses the System Status service for periodic status reporting (i.e., heartbeats) using the capabilities exchange services. The Get Switching Function Capabilities service positive acknowledgement defines a parameter (`systemStatusTimer`) that is used to indicate whether periodic status reporting is used and if so, how often the computing function should expect the reports. The recovery action to be taken by the computing function in the event of a loss of heartbeats is implementation specific.

14.2.3 Switching Function Capabilities Changed (to be defined)

S → C

14.2.4 Switching Function Devices Changed (to be defined)

S → C

15 Monitoring Services

NOTE

This clause describes Monitoring Services between the Switching Function and the Computing Function.

15.1 Services

Table 15-1 Monitoring Services Summary

Monitoring Service	Description	Pg.
15.1.1 Change Monitor Filter	Modifies the event filter for an existing monitor.	95
15.1.2 Monitor Start	Initiates an event monitor on a specified device or call.	97
15.1.3 Monitor Stop	Terminates an existing monitor.	102

15.1.1 Change Monitor Filter

C → S

The Change Monitor Filter service is used to modify the set of event reports that are filtered out (not sent) over an existing monitor.

The new set of filtered out (not sent) event reports may be listed in the service acknowledgement.

15.1.1.1 Service Request

Table 15-2 Change Monitor Filter—Service Request

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	MonitorCrossRefID	M	This indicates the monitor for which to change the filter.
requestedFilterList	Bitmap	M	This parameter specifies the requested set of events to be filtered out (not sent) by the server. It is a bitmap of all events defined in this Standard.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

15.1.1.2 Service Response

This service follows the atomic acknowledgement model for this service request.

15.1.1.2.1 Positive Acknowledgement

Table 15-3 Change Monitor Filter—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
actualFilterList	Bitmap	C	This parameter specifies the actual set of events that will be filtered out (not sent) by the server. It is a bitmap of all events defined in this Standard. The actual events filtered may differ from the requestedFilterList parameter on the service request (See Functional Requirement #1). This parameter is optional if the actualFilterList is the same as the requestedFilterList parameter on the service request, otherwise it is mandatory.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

15.1.1.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

15.1.1.3 Operational Model

15.1.1.3.1 Connection State Transitions

There are no connection state changes due to this service.

15.1.1.3.2 Monitoring Requirements

- Once a request has been acknowledged, a new set of events will be filtered out (not sent) by the server.

15.1.1.3.3 Functional Requirements

- An implementation that does not support all event reports, or that does not support filtering will nevertheless accept the Change Monitor Filter service even if the requested filter cannot be provided. In this case, the service acknowledgement indicates the actual set of events that will be filtered out (not sent). This means that the actual set of filtered events returned in the positive acknowledgement may include additional events to be filtered out (or fewer monitored events supported for the monitor) than those requested in the service request. For example, an implementation that does not support event filtering responds to the Change Monitor Filter service with a filter that shows provided events as unfiltered and unimplemented events as filtered out.

Similarly, an implementation that does not support, for example, Delivered events, shall always respond with a filter indicating that Delivered events will not be reported.

15.1.2 Monitor Start

C → S

The Monitor Start service initiates event reports (otherwise known as events) for a call, device, or for one or more calls involving a device.

The server starts a monitor, allocates a Monitor Cross Reference Identifier that uniquely identifies the monitor, and then positively acknowledges the request. All activities satisfying the filter provided (for example: call, feature, agent, private) trigger events which are delivered as a stream of event reports to the server. Each event contains the Monitor Cross Reference Identifier that correlates the event back to the Monitor Start service that established the monitor.

These event reports cease after the switching function terminates the monitor. Service termination can result from a client request (15.1.3, “Monitor Stop” on page -102) or it can be initiated by the server. The switching function shall terminate the monitor if the monitorObject ceases to exist, or if the monitorObject leaves the switching sub-domain. There may be other conditions that cause the server to terminate the monitor.

Once the monitor is terminated, the monitor cross reference ID is no longer valid.

Please refer to 6.6.2, “Monitoring”, on page 32 for an overview of monitoring and related concepts such as monitor objects, monitor types, monitor call types, and monitor filters.

15.1.2.1 Service Request

Table 15-4 Monitor Start—Service Request

Parameter Name	Type	M/O/C	Description
monitorObject	MonitorObject	M	Specifies the monitor object of a call or device to be monitored. The complete set of possible values is: <ul style="list-style-type: none"> • call (i.e., connection ID) • device (i.e., device ID) See Functional Requirement #5.
requestedMonitorFilter	Bitmap	O	This parameter specifies the requested set of events to be filtered out (not sent) by the switching function. It is a bitmap of all events defined in this standard. <p>If this parameter is not provided (or if the parameter is not supported by the switching function), then it shall mean that no filtering of events is requested (all events are requested).</p>
monitorType	Enumerated	O	Specifies the type of monitor requested. The complete set of possible values is: <ul style="list-style-type: none"> • call-type • device-type <p>If this parameter is not provided (or if the parameter is not supported by the switching function), then the monitor type shall be selected by the switching function (as indicated by the capabilities exchange services).</p>
requestedMonitorMediaClass	Bitmap	O	Specifies the media classes (voice, digital data, etc.) of calls that are being requested to be monitored for the monitorObject. <p>Refer to the mediaClass component in 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete set of possible values. Note that multiple bits may be set.</p> <p>If this parameter is not provided (or if the parameter is not supported by the switching function), it is switching function dependent which media classes of calls are monitored.</p>
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.

Table 15-4 Monitor Start—Service Request (continued)

Parameter Name	Type	M/ O/C	Description
privateData	CSTAPrivateData	O	Specifies non-standardized information.

15.1.2.2 Service Response

This service follows the atomic acknowledgement model for this service request.

15.1.2.2.1 Positive Acknowledgement

Table 15-5 Monitor Start—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	MonitorCrossRefID	M	This indicates a value that is unique within the association for the duration of the monitor and that can be used to relate subsequent events to the monitor request that initiated them. It shall also allow correlating Monitor Stop and subsequent Change Monitor Filter services with the original Monitor Start service on which they act.
actualMonitorFilter	Bitmap	C	<p>This parameter specifies the actual set of events that will be filtered (not sent) by the switching function. It is a bitmap of all events defined in this standard. The actual events filtered out may differ from the filterList parameter on the service request (See Functional Requirement #1).</p> <p>If this parameter is supported by the switching function, it may be omitted if the requested and actual monitor filters are the same, otherwise it shall be provided.</p> <p>If the parameter is not supported by the switching function, then the switching function does not filter events and all events supported (as indicated by the capability exchange services) shall be sent for this monitor.</p>
actualMonitorMediaClass	Bitmap	C	<p>This parameter specifies the actual media classes of calls that are monitored by the switching function for this monitor.</p> <p>The actual media classes of calls monitored may be the same or a subset of what was requested on the service request.</p> <p>If this parameter is supported by the switching function, it may be omitted if the requested and actual monitor media class parameters are the same otherwise it shall be provided.</p> <p>If the parameter is not supported by the switching function, then the switching function does not filter call types for specific monitors. The capability exchange services indicates the media classes of calls that can be monitored.</p>
monitorExistingCalls	Boolean	O	<p>Indicates whether or not the computing function will receive event reports regarding calls that are currently existing at the device at which the monitor was started. The complete set of possible values is:</p> <ul style="list-style-type: none"> • TRUE - Indicates event reports will be provided for calls that are at the device at the time of the acknowledgement [Default]. • FALSE - Indicates event reports will not be provided for calls that are at the device at the time of the acknowledgement. <p>This parameter is applicable to monitors that have devices as their object. For such monitors, if this parameter is not present (or the parameter is not supported), it means that the switching function always provides event reports for calls that are currently present at the device when the monitor was started.</p>
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

15.1.2.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

15.1.2.3 Operational Model

15.1.2.3.1 Connection State Transitions

There are no connection state changes due to this service.

15.1.2.3.2 Monitoring Requirements

1. For call related events, events are provided for all devices associated with the call or device being monitored. For non-call related events, events are provided for the monitored device.
2. Once a call is monitored (irrespective of the monitor type or monitor object), all connection state changes that are known by the switching function are reported (subject to the Monitor Filter).
 - For example, if device A is being monitored (with a device-type monitor) and a call is placed to device B (no monitor on B), then any connection state changes for either device A or B (such as when B receives the Delivered event and answers the call) will be reported through device A’s monitor.
 - Monitoring is only guaranteed for devices in the switching sub-domain. Activity related to devices outside the switching sub-domain may be only partially available or completely unreported.
3. Since some devices do not support all events, when a device enters a call that is being monitored with a call-type monitor, there may be a reduced set of event reporting associated with that monitor.
4. Physical Device, Logical Device, and Maintenance events are only reported for device-type monitors.

15.1.2.3.3 Functional Requirements

1. An implementation that does not support all event reports, or that does not support filtering will nevertheless accept the Monitor Start service even if the requested filter cannot be provided. In this case (if the actualMonitorFilter parameter is supported), the service acknowledgement indicates the actual set of events that will be filtered out. This means that the actual set of filtered events returned in the positive acknowledgement may include additional events to be filtered out (or less monitored events supported for the monitor) than those requested in the service request. For example, an implementation that does not support event filtering responds to the Monitor Start service with a filter that shows provided events as unfiltered and unimplemented events as filtered out. Similarly, an implementation that does not support, for example, Delivered events, shall respond with a filter indicating that Delivered events will not be reported.
2. If the switching function does not support the requestedMonitorFilter parameter (and the switching function is ignoring unsupported parameters, as defined by the capability exchange services), the computing function may receive events that it had requested to be filtered out.
3. When monitoring device configurations, refer to 6.1.7, “Referencing Devices, Elements, Appearances and Device Configurations”.
4. Events that occurred prior to the Monitor Start positive acknowledgement are not reported.
5. When a call-type monitor is requested and a connection identifier is being provided, the connection identifier may consist of only the callID. This is an exception to the general rule that deviceIDs are always provided in connectionIDs (refer to section 12.3.8, “ConnectionID”).
6. The Service Completion Failure event is *only* reported to device-type monitors on the device which has or had connection(s) that were used in the particular service request which is associated with this event (i.e., this event is not reported to any call-type monitors or device-type monitors for other devices in the call(s) associated with the service request).
7. If a computing function starts a monitor on a device, then issues another Monitor Start on the same device, the switching function will start a new monitor and will create a new Monitor Cross Reference Identifier. The new Monitor Start service request can be the same as the original or it may include a different Monitor Type or a different Monitor Filter.

8. The event reporting of the Failed event may vary depending on the given switching function. For the conditions under which the event reporting is limited, see section 6.7, “Additional Services, Features & Behaviour” and the Failed event section.
9. If filtering of the individual Private Events is desired, then the CSTA Private Data Information (privateData parameter) shall be used.

15.1.3 Monitor Stop

C ↔ S

The Monitor Stop service is used to cancel a previously initiated Monitor Start service.

The Monitor Stop service can be issued by a function to terminate or signal the termination of a corresponding Monitor Start service.

A positive acknowledgement to the service request indicates that the Cross Reference ID used by the Monitor Start service has become invalid.

15.1.3.1 Service Request

Table 15-6 Monitor Stop—Service Request

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	MonitorCrossRefID	M	This indicates the Cross Reference Identifier provided in the original Monitor Start service positive acknowledgement.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

15.1.3.2 Service Response

This service follows the atomic acknowledgement model for this service request.

15.1.3.2.1 Positive Acknowledgement

Table 15-7 Monitor Stop—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

15.1.3.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

15.1.3.3 Operational Model

15.1.3.3.1 Connection State Transitions

There are no connection state changes due to this service.

15.1.3.3.2 Monitoring Requirements

1. Once a request has been acknowledged, event reports are no longer sent.

15.1.3.3.3 Functional Requirements

1. This service can be sent to the function that requested a monitor to report that the monitor has been terminated.
2. The switching function may issue a Monitor Stop service when it can no longer provide information. Examples of when this may occur are:
 - For monitors on calls that have ended (call monitoring).
 - For load management reasons.
 - If the monitor object leaves the switching sub-domain (via configuration, for example).

16 Snapshot Services

NOTE

This clause describes Snapshot Services between the Switching Function and the Computing Function.

16.1 Services

Table 16-1 Snapshot Services Summary

Snapshot Service	Description	Pg.
16.1.1 Snapshot Call	Provides information about the devices participating in a specified call.	104
16.1.2 Snapshot Device	Provides information on the status of calls at a specific device.	107
16.1.3 Snapshot CallData	Provides Snapshot Call Information in segmented messages.	110
16.1.4 Snapshot DeviceData	Provides Snapshot Device Information in segmented messages.	112

16.1.1 Snapshot Call

C → S

The Snapshot Call service provides information about the devices participating in a specified call. The information provided includes device identifiers, their connections in the call, and local connection states of the devices in the call as well as call related information.

Information that applies to the entire call shall be provided in the Snapshot Call positive response.

Information that is specific to each endpoint in the call (snapshotData parameter) shall be provided using one of two possible mechanisms (as indicated by the capability exchange services): either in the Snapshot Call positive acknowledgement or in one or more messages using the Snapshot CallData Service. Note that both mechanisms cannot be used at the same time.

If the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), then for each connection in the call, this service provides the list of permitted call control services.

16.1.1.1 Service Request

Table 16-2 Snapshot Call—Service Request

Parameter Name	Type	M/O/C	Description
snapshotObject	ConnectionID	M	This indicates the connectionID of the call to be snapshot. See Functional Requirement #2.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

16.1.1.2 Service Response

This service follows the atomic acknowledgement model for this service request.

16.1.1.2.1 Positive Acknowledgement

Table 16-3 Snapshot Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
serviceCrossRefID	ServiceCrossRefID	C	Specifies the reference used to associate subsequent Snapshot CallData services to this service request. This parameter is mandatory if the switching function is providing the snapshot information using the Snapshot CallData service, otherwise it shall not be provided.
snapshotData	List of Structures	C	Specifies information for each endpoint in a call. This parameter is mandatory if the switching function is providing all of the response information in this message. This parameter shall not be provided if the switching function is providing the snapshot information using the Snapshot CallData Service. The complete set of possible information is: <ul style="list-style-type: none"> • deviceOnCall (M) - Of a device involved with the endpoint. • connectionIdentifier (C) - For the endpoint. This is mandatory if the endpoint is in the switching sub-domain, otherwise it is optional. • localConnectionState (O) - For the endpoint. • servicesPermitted (C) - This is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange service). • mediaServiceInformationList (O) - Specifies information about the media services that are attached (bound to) the connection in the call. The complete set of possible values is: <ul style="list-style-type: none"> • mediaServiceType (M) - A media service type (i.e., type is characters) that has been bound to the connection. • mediaServiceInstance (O) - A media service instance (i.e., type is MediaServiceInstance-ID) associated with the media service bound to the connection. • mediaStreamID (M) - The media stream identifier (i.e., type is MediaStreamID) for the media service binding. • connectionInformation - ConnectionInformation (O) - The connection information associated with the callIdentifier connection.
mediaCallCharacteristics	MediaCallCharacteristics	O	This specifies the media class and data characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete set of possible values.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
callingDevice	CallingDeviceID	O	Specifies the calling device.
calledDevice	CalledDeviceID	O	Specifies the called device.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.

Table 16-3 Snapshot Call—Positive Acknowledgement (continued)

Parameter Name	Type	M/O/C	Description
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

16.1.1.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

16.1.1.2.3 Operational Model

16.1.1.2.4 Connection State Transitions

There are no connection state changes due to this service.

16.1.1.2.5 Monitoring Requirements

This service does not affect existing monitors.

16.1.1.2.6 Functional Requirements

1. The Snapshot Call service is intended to provide information about devices in calls that makes further monitoring more meaningful. For example, if the computing function started working with a call, the event reports needed to provide synchronization may not occur for some time. To facilitate operations before an event report is available to synchronize the monitor, it is necessary to be able to query the current state of devices. Snapshot Call service provides this function.
2. The connection ID provided in the service may consist of only a callID portion. This is an exception to the rule of always providing deviceIDs in connectionIDs of service requests as described in 12.3.8, “ConnectionID”, on page 74.
3. If the switching function is providing the snapshot response information in multiple messages, it shall support the Snapshot CallData service. In this case, the computing function can associate subsequent Snapshot CallData services to the original Snapshot Call service by means of the serviceCrossRefID parameter.

16.1.2 Snapshot Device

S → C

The Snapshot Device service provides information about calls associated with a given device. The information provided identifies each call the device is participating in and the local connection state of the device in that call.

The switching function shall provide the response information using one of two possible mechanisms (as indicated by the capability exchange services): either in the Snapshot Device positive acknowledgement or in one or more messages using the Snapshot DeviceData Service. Note that both mechanisms cannot be used at the same time.

If the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), then for each connection at the device, this service provides the list of permitted call control services.

16.1.2.1 Service Request

Table 16-4 Snapshot Device—Service Request

Parameter Name	Type	M/O/C	Description
snapshotObject	DeviceID	M	This indicates the deviceID of the device to be snapshot.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

16.1.2.2 Service Response

This service follows the atomic acknowledgement model for this service request.

16.1.2.2.1 Positive Acknowledgement

Table 16-5 Snapshot Device—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
serviceCrossRefID	ServiceCrossRefID	C	Specifies the reference used to associate subsequent Snapshot DeviceData services to this service request. This parameter is mandatory if the switching function is providing the snapshot device information using the Snapshot DeviceData service, otherwise it shall not be provided.
snapshotData	List of Structures	C	Specifies information for each call at a device. This parameter is mandatory if the switching function is providing all of the response information in this message. This parameter shall not be provided if the switching function is providing the response information using the Snapshot DeviceData Service. This complete set of information is: <ul style="list-style-type: none"> • connectionIdentifier (M) • localCallState (M). This is one of the following possible values: <ul style="list-style-type: none"> • compound call state - This consists of a sequence of local connection states. • simpleCallState - See 12.2.18, “SimpleCallState”, on page 68 for the list of possible values • unknown • servicesPermitted (C) - This is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services). • mediaServiceInformationList (O) - Specifies information about the media services that are attached (bound to) the connection in the call. The complete set of possible elements include: <ul style="list-style-type: none"> • mediaStreamID (M) - The media stream identifier (i.e., type is MediaStreamID) for the media service binding. • connectionInformation - ConnectionInformation (O) - The connection information associated with the callIdentifier connection • MediaCallCharacteristics (O) - specifies the media class and data characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the list of possible values.
security	CSTASecurityData	O	Specifies the timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

16.1.2.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

16.1.2.3 Operational Model

16.1.2.3.1 Connection State Transitions

There are no connection state changes due to this service.

16.1.2.3.2 Monitoring Requirements

This service does not affect existing monitors.

16.1.2.3.3 Functional Requirements

1. The Snapshot Device service is intended to provide information about devices to make further monitoring more meaningful. For example, when a computing function starts working with a device, the event reports

needed to provide synchronization may not occur for some time. To facilitate operations before event reports synchronize the monitor, it is necessary to be able to query the current state of devices. Snapshot Device service provides that function.

2. If the switching function is providing the snapshot device response information in multiple messages, it shall support the Snapshot DeviceData service. In this case, the computing function can associate subsequent Snapshot DeviceData services to the original Snapshot Device service by means of the serviceCrossRefID parameter.

16.1.3 Snapshot CallData

S → C

This service is generated as a result of the Snapshot Call service. It is used when the switching function is providing snapshot call information in multiple messages (otherwise the switching function provides the snapshot call information in the Snapshot Call positive acknowledgement).

The information provided includes information about the endpoints in the call (information about the entire call is provided in the Snapshot Call positive response).

The switching function may generate a sequence of Snapshot CallData services, individually referred to as segments, in response to a single Snapshot Call service request.

16.1.3.1 Service Request

Table 16-6 Snapshot CallData—Service Request

Parameter Name	Type	M/O/C	Description
serviceCrossRefID	ServiceCrossRefID	M	Specifies the reference used to associate the Snapshot CallData service messages to the Snapshot Call service request.
segmentID	Value	O	Specifies the segment number of this message. Each successive segment number in the sequence increments the segmentID by one.
lastSegment	Boolean	M	Specifies if this segment is the last one associated with the serviceCrossRefID. The complete set of possible values is: <ul style="list-style-type: none"> • TRUE - Indicates that this is the last segment • FALSE - Indicates that this is not the last segment in the sequence.
snapshotData	List of Structures	M	Specifies information for each endpoint in a call. The complete set of possible information is: <ul style="list-style-type: none"> • deviceOnCall (M) - Of a device involved with the endpoint. • connectionIdentifier (C) - For the endpoint. This is mandatory if the endpoint is in the switching sub-domain, otherwise it is optional. • localConnectionState (O) - For the endpoint. • servicesPermitted (C) - This is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange service). • mediaServiceInformationList (O) - Specifies information about the media services that are attached (bound to) the connection in the call. The complete set of possible values is: <ul style="list-style-type: none"> • mediaServiceType (M) - A media service type (i.e., type is characters) that has been bound to the connection. • mediaServiceInstance (O) - A media service instance (i.e., type is MediaServiceInstance-ID) associated with the media service bound to the connection. • mediaStreamID (M) - The media stream identifier (i.e., type is MediaStreamID) for the media service binding. • connectionInformation - ConnectionInformation (O) - The connection information associated with the callIdentifier connection.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

16.1.3.2 Service Response

There are no service request completion conditions associated with this service.

16.1.3.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service request.

16.1.3.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, "ErrorValue", on page 65.

16.1.3.3 Operational Model

16.1.3.3.1 Connection State Transitions

There are no connection state changes due to this service.

16.1.3.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

16.1.3.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

16.1.3.3.4 Functional Requirements

1. Due to the nature of the switching function configuration, the switching function may buffer the Snapshot CallData messages to the computing function. The time between these messages may vary significantly between various implementations.
2. The switching function may include information for one or more endpoints in each segment.
3. The information reported in the sequence of segments generated from the Snapshot Call service request represents the state of the call at the time the Snapshot Call service is positively acknowledged.

16.1.4 Snapshot DeviceData

S → C

This service is generated as a result of the Snapshot Device service. It is used when the switching function is providing snapshot device response information in multiple messages (otherwise the switching function provides the snapshot device response in the Snapshot Device positive acknowledgement).

This includes information about calls associated with a given device. The information provided identifies each call the device is participating in and the local connection state of the device in that call.

The switching function may generate a sequence of Snapshot DeviceData services, individually referred to as segments, in response to a single Snapshot Device service request.

16.1.4.1 Service Request

Table 16-7 Snapshot DeviceData—Service Request

Parameter Name	Type	M/O/C	Description
serviceCrossRefID	ServiceCrossRefID	M	Specifies the reference used to associate the Snapshot DeviceData service messages to the Snapshot Device service request.
segmentID	Value	O	Specifies the segment number of this message. Each successive segment number in the sequence increments the segmentID by one.
lastSegment	Boolean	M	Specifies if this segment is the last one associated with the serviceCrossRefID. The complete set of possible values is: <ul style="list-style-type: none"> • TRUE - Indicates that this is the last segment • FALSE - Indicates that this is not the last segment in the sequence.
snapshotData	List of Structures	M	Specifies information for each call at a device. This complete set of information is: <ul style="list-style-type: none"> • connectionIdentifier (M) • localCallState (M). This is one of the following possible values: <ul style="list-style-type: none"> • compound call state - This consists of a sequence of local connection states. • simpleCallState - See 12.2.18, “SimpleCallState”, on page 68 for the list of possible values • unknown • servicesPermitted (C) - This is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services). • mediaServiceInformationList (O) - Specifies information about the media services that are attached (bound to) the connection in the call. The complete set of possible elements include: <ul style="list-style-type: none"> • mediaStreamID (M) - The media stream identifier (i.e., type is MediaStreamID) for the media service binding. • connectionInformation - ConnectionInformation (O) - The connection information associated with the callIdentifier connection • MediaCallCharacteristics (O) - specifies the media class and data characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the list of possible values.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

16.1.4.2 Service Response

There are no service request completion conditions associated with this service.

16.1.4.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service request.

16.1.4.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, "ErrorValue", on page 65.

16.1.4.3 Operational Model

16.1.4.3.1 Connection State Transitions

There are no connection state changes due to this service.

16.1.4.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

16.1.4.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

16.1.4.3.4 Functional Requirements

1. Due to the nature of the switching function configuration, the switching function may buffer the Snapshot DeviceData messages to the computing function. The time between these messages may vary significantly between various implementations.
2. The switching function may include information for one or more connections in each segment.
3. The information reported in the sequence of segments generated from the Snapshot Device service request represents the state of the device at the time the Snapshot Device service is positively acknowledged.

17 Call Control Services & Events

This clause describes the Call Control features of this Standard. It includes:

- Call Control services
- Call Control events

For a description of fundamental concepts, such as connection states, please refer to 6.1.5 beginning on page 24.

17.1 Services

Table 17-1 Call Control Services Summary

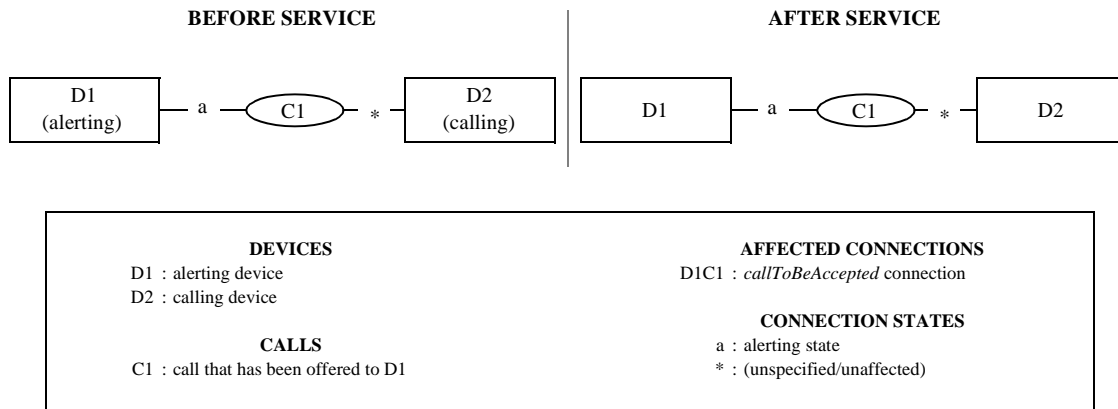
Call Control Service	Description	Pg.
17.1.1 Accept Call	Causes an offered call to transition to the Ringing or Entering Distribution mode of the Alerting state.	116
17.1.2 Alternate Call	Places an existing call on hold and then retrieves a previously held or alerting call at the same device.	118
17.1.3 Answer Call	Answers a call that is ringing, queued, or being offered to a device.	121
17.1.4 Call Back Call-Related	Allows a computing function to request that an originally called device return a call to the original calling device.	123
17.1.5 Call Back Message Call-Related	Allows a computing function to instruct the switching function to leave a pre-defined message requesting that the called device call the calling device.	126
17.1.6 Camp On Call	Queues a call at a busy device until the device becomes available.	129
17.1.7 Clear Call	Releases all of the devices associated with the specified call.	131
17.1.8 Clear Connection	Releases a specific device from a call.	134
17.1.9 Conference Call	Provides a conference of an existing held call and another active call at a conferencing device. The two calls are merged into a single call at the conferencing device.	138
17.1.10 Consultation Call	Places an existing active call at a device on hold and initiates a new call from the same device.	141
17.1.11 Deflect Call	Deflects a call to another device.	147
17.1.12 Dial Digits	Dials a digit sequence for a call that has already been initiated.	150
17.1.13 Directed Pickup Call	Picks a specified call. (Moves and connects a specified alerting or queued call.)	153
17.1.14 Group Pickup Call	Picks a call from a specified pick group. (Moves and connects any alerting call in a pick group to another device.)	156
17.1.15 Hold Call	Places a specific connection on hold.	159
17.1.16 Intrude Call	Allows a computing function to add the calling device to a call at a busy called device.	161
17.1.17 Join Call	Allows a computing function to request, on behalf of a device, that the device be joined into an existing call.	165
17.1.18 Make Call	Establishes a call between two devices.	169
17.1.19 Make Predictive Call (to be defined)	Establishes a call between two devices. The calling device is presented with the call only after the called device is alerted or has answered the call.	174
17.1.20 Park Call	Parks a call at a specified device. (Moves and queues a connected call to another device).	175
17.1.21 Reconnect Call	Clears an existing connection and then connects a previously held connection at the same device.	178
17.1.22 Retrieve Call	Connects to a call that had previously been placed on hold.	180
17.1.23 Single Step Conference Call	Adds a device to an existing call.	182
17.1.24 Single Step Transfer Call	Replaces a device in an existing call with another device.	186
17.1.25 Transfer Call	Transfers a held call to the consulted party.	189

17.1.1 **Accept Call**

C → S

The Accept Call service causes an offered call to transit from the offered mode to the Ringing or Entering Distribution mode of the alerting state.

Figure 17-1 Accept Call Service



17.1.1.1 **Service Request**

Table 17-2 Accept Call—Service Request

Parameter Name	Type	M/O/C	Description
callToBeAccepted	ConnectionID	M	Specifies the connection to be accepted.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.1.2 **Service Response**

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.1.2.1 **Positive Acknowledgement**

Table 17-3 Accept Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.1.2.2 **Negative Acknowledgement**

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.1.3 **Operational Model**

17.1.1.3.1 Connection State Transitions

Table 17-4 Accept Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callToBeAccepted)	Alerting (Offered mode only)	Alerting (Ringing or Entering Distribution mode)
other connections (i.e., D2C1)	(Unspecified)	(Unaffected; no transition due to this service).

17.1.1.3.2 Device-Type Monitoring Event Sequences

Table 17-5 Accept Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (Alerting device)	D1C1 (callToBeAccepted)	Delivered	Normal, Entering Distribution
D2 (Calling device or any other devices in call C1)	D1C1 (callToBeAccepted)	Delivered	Normal, Entering Distribution

17.1.1.3.3 Call-Type Monitoring Event Sequences

Table 17-6 Accept Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1 (callToBeAccepted)	Delivered	Normal, Entering Distribution

17.1.1.3.4 Functional Requirements

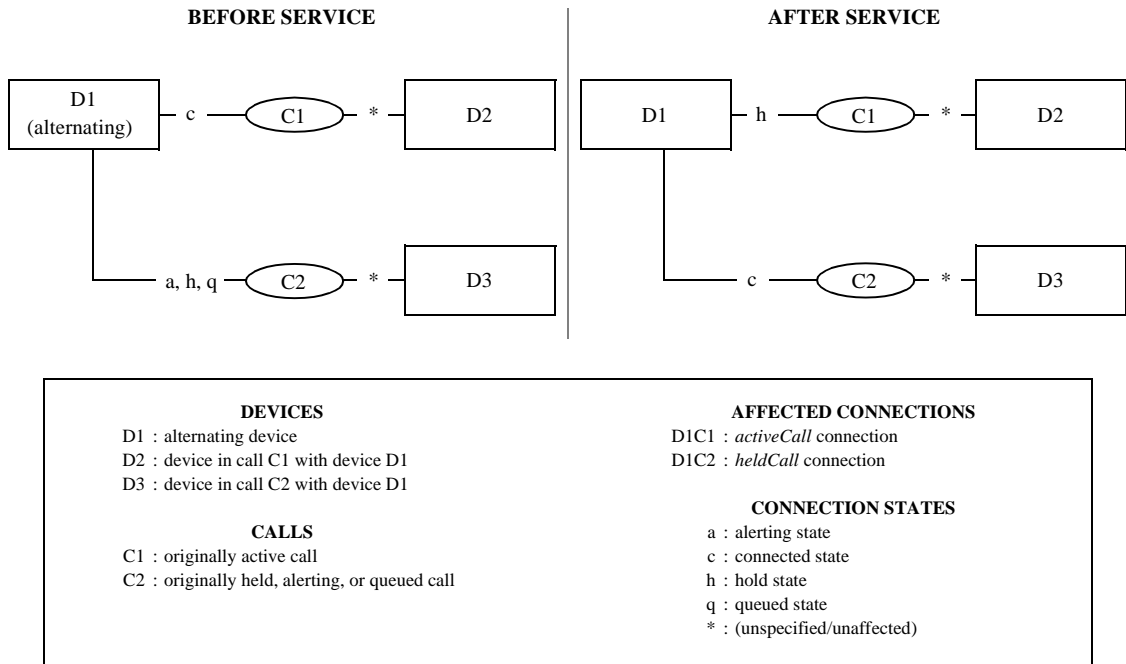
1. If the computing function wants to clear an active call prior to accepting an alerting call for a device, it shall first issue a Clear Connection service for the active call and then issue the Accept Call service for the alerting call.

17.1.2 Alternate Call

C → S

The Alternate Call service places an existing active call on hold and then retrieves a previously held call. This service is also used to place an active call on hold and then connect to an alerting or queued call at the same device (i.e., to answer a call-waiting call).

Figure 17-2 Alternate Call Service



17.1.2.1 Service Request

Table 17-7 Alternate Call—Service Request

Parameter Name	Type	M/O/C	Description
heldCall	ConnectionID	M	Specifies the held connection for the alternating device.
activeCall	ConnectionID	M	Specifies the active connection for the alternating device.
connectionReservation	Boolean	O	Specifies that the media stream channel(s) associated with the call being placed on hold be reserved for reuse at a later time. The complete set of possible values is: <ul style="list-style-type: none"> • True - channel(s) is to be reserved. • False - channel(s) is not to be reserved (default).

Table 17-7 Alternate Call—Service Request (continued)

Parameter Name	Type	M/O/C	Description
consultOptions	Enumerated	C	<p>This parameter indicates the potential actions following the Alternate Call so that certain facilities can be allocated prior to a Transfer or Conference. The complete set of possible values is:</p> <ul style="list-style-type: none"> • Consult Only • Transfer Only • Conference Only • Unrestricted (default) <p>If the switching function supports this parameter, the computing function shall supply one of the values supported. If the switching function does not support this parameter, the implicit value is Unrestricted. This parameter does not indicate a restriction or otherwise affect the switching function's capabilities in any way with respect to services other than Transfer or Conference.</p>
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information.

17.1.2.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.2.2.1 Positive Acknowledgement

Table 17-8 Alternate Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.2.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, "ErrorValue", on page 65.

17.1.2.3 Operational Model

17.1.2.3.1 Connection State Transitions

Table 17-9 Alternate Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (activeCall)	Connected	Hold
D1C2 (heldCall)	Hold, Alerting, or Queued	Connected
other connections (i.e., D2C1, D3C2)	(Unspecified)	(Unaffected; no transition due to this service).

17.1.2.3.2 Device-Type Monitoring Event Sequences

Sequence of events using call-type monitoring is defined in Table 17-10 for the following cases:

- Case A: Call C2 is on Hold at device D1 (Alternating Device)

- Case B: Call C2 is Alerting or Queued at device D1 (Alternating Device)

Table 17-10 Alternate Call—Device-Type Monitoring Event Sequences (Case A and B)

Monitored Device	Connection	Event	Event Cause
D1 (Alternating device)	D1C1 (activeCall)	Held	Transfer, Conference, Alternate, or Normal
	D1C2 (heldCall)	Retrieved (Case A)	Transfer, Conference, Alternate, or Normal
Established (Case B)			
D2 (or any other devices in conference with D2)	D1C1 (activeCall)	Held	Transfer, Conference, Alternate, or Normal
D3 (or any other devices in conference with D3)	D1C2 (heldCall)	Retrieved (Case A)	Transfer, Conference, Alternate, or Normal
		Established (Case B)	

17.1.2.3.3 Call-Type Monitoring Event Sequences

Sequence of events using call-type monitoring is defined in Table 17-11 for the following cases:

- Case A: Call C2 is on Hold at device D1 (Alternating Device)
- Case B: Call C2 is Alerting or Queued at device D1 (Alternating Device)

Table 17-11 Alternate Call—Call-Type Monitoring Event Sequences (Case A and B)

Monitored Call	Connection	Event	Event Cause
C1 (originally active call)	D1C1 (activeCall)	Held	Transfer, Conference, Alternate, or Normal
C2 (originally held, alerting, or queued call)	D1C2 (heldCall)	Retrieved (Case A)	Transfer, Conference, Alternate, or Normal
		Established (Case B)	

17.1.2.3.4 Functional Requirements

1. The Alternate Call service shall not be used to put an alerting call on hold and re-connect to another call that is on hold.
2. This service is a multiple step service that is equivalent to the computing function issuing the Hold Call service for the activeCall ConnectionID and then issuing one of the following services:
 - a. a Retrieve Call service for the heldCall ConnectionID.
 - b. an Answer Call service for the heldCall ConnectionID.
 - c. an Accept Call service for the heldCall ConnectionID.
3. The consultOptions parameter indicates the potential action following the Alternate Call service so that certain facilities can be allocated if a transfer or conference is desired. If the switching function supports the consultOptions parameter, the computing function shall provide one of the supported values of this parameter obtained through the capability exchange services. If the switching function requires preallocation of resources to support transfer or conference but is unable to allocate the resources required to support the indicated usage at this time, it shall reject the Alternate Call service request. This parameter does not indicate a restriction or otherwise affect the switching function’s capabilities in any way with respect to services other than Transfer or Conference.
4. If all appearances of a shared bridged device configuration are in the hold state and the heldCall parameter contains an appearance’s connection ID in the call, then the other appearances in the device configuration will return to the inactive mode (Queued state, Bridged Events).

17.1.3 Answer Call

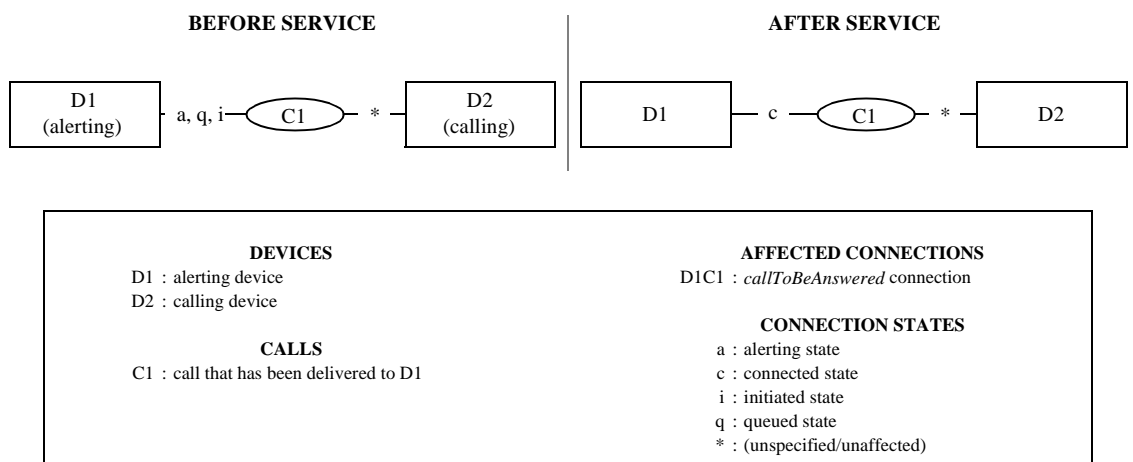
C → S

The Answer Call service connects an alerting or queued call.

This service is typically associated with devices that have attached speakerphone units and headset telephones to connect to a call via hands-free operation. For example, when the call is answered, one of the following actions may occur:

- If the specified device has a speaker and a microphone, the speaker and microphone are turned on.
- If the specified device only has a speaker, the speaker is turned on. The handset shall be picked up in order to have a two way conversation.
- If there is no speaker, then the handset shall be picked up in order to have a two-way conversation.
- If the specified device has a headset, the headset is turned on.

Figure 17-3 Answer Call Service



Note that D1 may also be the calling device (e.g. Make Predictive Call).

17.1.3.1 Service Request

Table 17-12 Answer Call—Service Request

Parameter Name	Type	M/O/C	Description
callToBeAnswered	ConnectionID	M	Specifies the connection to be answered.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.3.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.3.2.1 Positive Acknowledgement

Table 17-13 Answer Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.3.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.3.3 Operational Model

17.1.3.3.1 Connection State Transitions

Table 17-14 Answer Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callToBeAnswered)	Alerting, Initiated, or Queued	Connected
other connections (i.e., D2C1)	(Unspecified)	(Unaffected; no transition due to this service).

17.1.3.3.2 Device-Type Monitoring Event Sequences

Table 17-15 Answer Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (Alerting device)	D1C1 (callToBeAnswered)	Established	Normal
D2 (Calling device or any other devices in conference with D2)	D1C1 (callToBeAnswered)	Established	Normal

17.1.3.3.3 Call-Type Monitoring Event Sequences

Table 17-16 Answer Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1 (callToBeAnswered)	Established	Normal

17.1.3.3.4 Functional Requirements

1. If the computing function wants to clear an active call prior to answering an alerting or queued call for a device, it shall first issue a Clear Connection service for the active call and then issue the Answer Call service for the alerting or queued call.

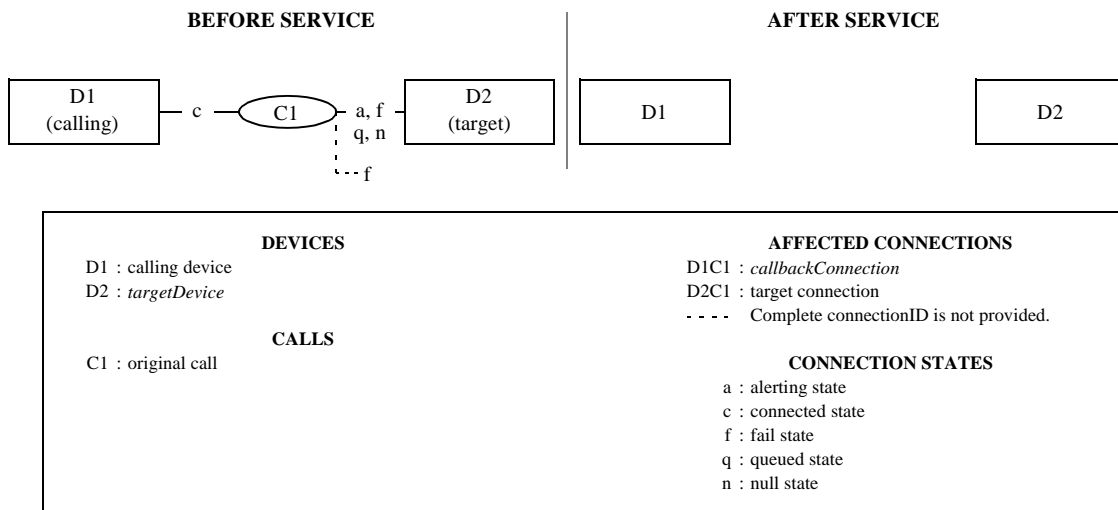
17.1.4 Call Back Call-Related

C → S

The Call Back Call-Related service allows a computing function to request that the calling device retry the call to the called device when the called device is in an appropriate state to accept the call.

As an example, the service might be used when a called device was busy so that the call is reattempted when the device becomes free.

Figure 17-4 Call Back Call-Related Service



Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

17.1.4.1 Service Request

Table 17-17 Call Back Call-Related —Service Request

Parameter Name	Type	M/O/C	Description
callbackConnection	ConnectionID	M	Specifies the call back connection at the calling device.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (Priority call, for example) to be associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values. If the supported characteristics cannot be honoured, the switching function shall reject the service request.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.4.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.4.2.1 Positive Acknowledgement

Table 17-18 Call Back Call-Related—Positive Acknowledgement

Parameter Name	Type	M/O/ C	Description
targetDevice	DeviceID	C	Specifies the deviceID of the device that the call back was initiated for. This parameter is mandatory if the switching function supports the Cancel Call Back service, otherwise it is optional.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.4.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.4.3 Operational Model

17.1.4.3.1 Connection State Transitions

Table 17-19 Call Back Call-Related —Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callback)	Connected	Null, Fail (see item #1)
D2C1 (called)	Fail, Alerting, Queued, or Null (e.g., null if call was forwarded or deflected from D2)	Null

1. A device may transition to the Fail state prior to Null if the device stays off-hook and receives busy or blocked tone.
2. In the case where a call is forwarded from a called device (busy forwarding, for example), it is switching function dependent if the call back request is placed on the called device or the device that the call was forwarded to.

17.1.4.3.2 Device-Type Monitoring Event Sequences

Table 17-20 Call Back Call-Related—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (calling device)	D2C1 (called)	Connection Cleared (See items #2 and #3)	Call Back or Normal Clearing
	D1C1 (callback)	Connection Cleared (see items #1 and #2)	Call Back or Normal Clearing
		Call Back	
D2 (called device)	D2C1 (called)	Connection Cleared (See items #2 and #3)	Call Back or Normal Clearing

17.1.4.3.3 Call-Type Monitoring Event Sequences

Table 17-21 Call Back Call-Related—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D2C1 (called)	Connection Cleared (see items #2 and #3)	Call Back or Normal Clearing
	D1C1 (callback)	Connection Cleared (see items #1 and #2)	Call Back or Normal Clearing
	(D1C1)	Call Cleared	Call Back or Normal Clearing

1. If a device stays off-hook and receives busy or blocked tone, the switching function sends the Failed event (event cause of Blocked or Busy) to indicate the status of the device, followed by a Connection Cleared event.
2. The exact sequence of the Connection Cleared events for this service is not specified. This is due to the fact that when multiple devices are removed from a call as a result of the service, each switching function may process the clearing of the devices in a different order.
3. This event is sent only if the connection associated with the called device was created.
4. The event sequence if the call is forwarded or deflected from D2 is not shown.

17.1.4.3.4 Functional Requirements

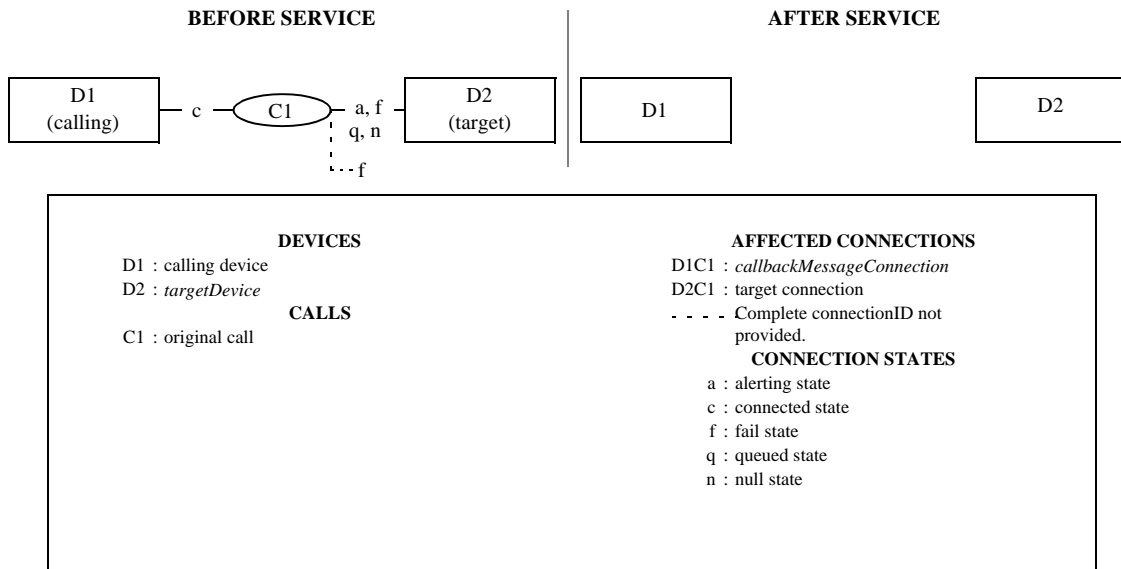
1. Call Back Call-Related is similar to the Camp On Call service. However for the Call Back Call-Related service the original call is cleared, but for the Camp On Call service, the call is queued.
2. Only one Call Back service request (Call-Related or Non-Call-Related) can be outstanding for any calling and called device pair. It is a switching function option (as indicated by the capability exchange services) if additional Call Back service requests (Call-Related or Non-Call-Related) for that pair result in a positive or negative acknowledgement from the switching function.
3. To cancel a Call Back (Call-Related or Non-Call-Related), the computing function shall issue the Cancel Call Back service, alternatively the Call Back should be manually canceled.

17.1.5 Call Back Message Call-Related

C → S

The Call Back Message Call-Related service allows a computing function to request that the switching function leave a pre-defined message requesting that the called device call the calling device. For example, the called device may have been busy when called.

Figure 17-5 Call Back Message Call-Related Service



Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

17.1.5.1 Service Request

Table 17-22 Call Back Message Call-Related—Service Request

Parameter Name	Type	M/O/C	Description
callbackMessageConnection	ConnectionID	M	Specifies the call back message connection at the calling device.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.5.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.5.2.1 Positive Acknowledgement

Table 17-23 Call Back Message Call-Related—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
targetDevice	deviceID	C	Specifies the deviceID of the device that the call back message was left for. This parameter is mandatory if the switching function supports the Cancel Call Back Message service, otherwise it is optional.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.5.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.5.3 Operational Model

17.1.5.3.1 Connection State Transitions

Table 17-24 Call Back Message Call-Related—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callbackMessage)	Connected	Null, Fail (see item #1)
D2C1 (called)	Fail, Alerting, Queued, or Null (e.g., null if call was forwarded, deflected from D2)	Null

1. A device may transition to the Fail state prior to Null if the device stays offhook and receives busy or blocked tone.
2. In the case where a call is forwarded from a called device (busy forwarding, for example), it is switching function dependent if the call back request is placed on the called device or the device that the call was forwarded to.

17.1.5.3.2 Device-Type Monitoring Event Sequences

Table 17-25 Call Back Message Call-Related—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (calling device)	D2C1 (called)	Connection Cleared (see items #3 and #4)	Call Back or Normal Clearing
	D1C1 (calling)	Connection Cleared (see items #2 and #4)	Call Back or Normal Clearing
		Call Back Message	
D2 (called device)	D2C1 (called)	Connection Cleared (see items #3 and #4)	Call Back or Normal Clearing

17.1.5.3.3 Call-Type Monitoring Event Sequences

Table 17-26 Call Back Message Call-Related—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D2C1 (called)	Connection Cleared (see items #3 and #4)	Call Back or Normal Clearing
	D1C1 (calling)	Connection Cleared (see items #2 and #4)	Call Back or Normal Clearing
	(D1C1)	Call Cleared	Call Back or Normal Clearing

1. The event sequence if the call is forwarded or deflected from D2 is not shown.
2. If a device stays off-hook and receives busy or blocked tone, the switching function sends the Failed event (event cause of Blocked or Busy) to indicate the status of the device, followed by a Connection Cleared event.
3. This event is sent only if the connection associated with the called device was created.
4. The exact sequence of the Connection Cleared events for this service is not specified. This is due to the fact that when multiple devices are removed from a call as a result of the service, each switching function may process the clearing of the devices in a different order.

17.1.5.3.4 Functional Requirements

1. Only one Call Back Message service request (Call-Related or Non-Call-Related) can be outstanding for any calling and called device pair. It is a switching function option (as indicated by the capability exchange

services) if additional Call Back Message service requests (Call-Related or Non-Call-Related) for that pair result in a positive or negative acknowledgement from the switching function.

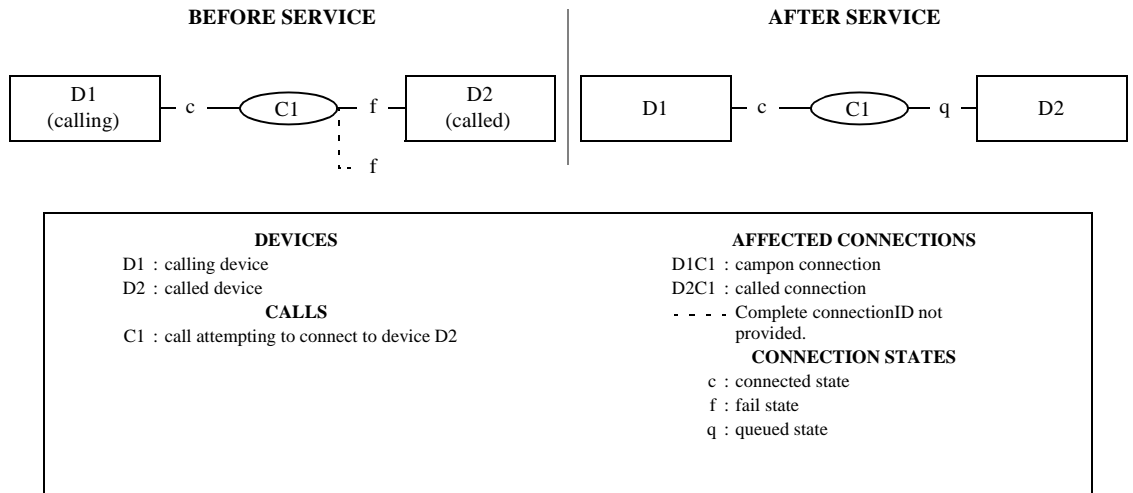
2. To cancel a Call Back Message (Call-Related or Non-Call-Related), the computing function shall issue the Cancel Call Back Message service, alternatively the Call Back Message should be manually canceled.
3. The Call Back Message service (Call-Related or Non-Call-Related) differs from the Call Back service in that with the Call Back Message service, the calling device will not call back the called device. Instead, this service leaves a message at the called device.
4. The switching function defines the message left at the called device. A computing function cannot use the service to specify the message content or how the switching function will notify the user (e.g., text message, voice message, indicator only).

17.1.6 Camp On Call

C → S

The Camp On Call service allows the computing function to queue a call for a device (that typically is busy) until that device becomes available (after finishing a current call or any previously queued calls, for example).

Figure 17-6 Camp On Call Service



Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

17.1.6.1 Service Request

Table 17-27 Camp On Call—Service Request

Parameter Name	Type	M/O/C	Description
camponConnection	ConnectionID	M	Specifies the connection of the calling device.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.6.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.6.2.1 Positive Acknowledgement

Table 17-28 Camp On Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.6.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.6.3 Operational Model

17.1.6.3.1 Connection State Transitions

Table 17-29 Camp On Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (campon)	Connected	Connected
D2C1 (called)	Fail	Queued

17.1.6.3.2 Device-Type Monitoring Event Sequences

Table 17-30 Camp On Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (Calling device)	D2C1 (campon)	Queued	Camp On or Camp On Trunks
D2 (Called device)	D2C1 (called)	Queued	Camp On or Camp On Trunks

17.1.6.3.3 Call-Type Monitoring Event Sequences

Table 17-31 Camp On Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D2C1 (called)	Queued	Camp On or Camp On Trunks

17.1.6.3.4 Functional Requirements

1. Only one Camp On Call feature can be active for any calling and called device pair. It is a switching function option (as indicated by the capability exchange services) if additional Camp On Call service requests for that pair result in a positive or negative acknowledgement from the switching function.
2. To cancel a Camp On Call service, the computing function can either:
 - Issue the Clear Connection service or Clear Call service with the campon connection.
 - Have the calling device go on-hook.

Note that if the Camp On Call service is successfully cancelled, normal call progress messages of Connection Cleared (with an event cause of Normal Clearing) will be generated.

17.1.7.2.1 Positive Acknowledgement

Table 17-33 Clear Call— Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.7.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.7.3 Operational Model

17.1.7.3.1 Connection State Transitions

Table 17-34 Clear Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1	(Any non-Null state)	Null, Failed (see item # 1)
D2C1	(Any non-Null state)	Null, Failed (see item # 1)
D3C1	(Any non-Null state)	Null, Failed (see item # 1)

1. A connection may transition to the Failed state prior to Null if the device stays off-hook and receives busy or blocked tone. The time between the state transition of Failed --> Null is device and switching function specific and the time may be substantial. Note that the service is completed when all connections transit to either the Null or the Failed state.

17.1.7.3.2 Device-Type Monitoring Event Sequences

Table 17-35 Clear Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1	D1C1	Failed (Optional) (see item #3)	Blocked, or Normal or Busy
		Connection Cleared (see items #1 and #2)	Normal Clearing
D2	D2C1	Failed (Optional) (see item #3)	Blocked, or Normal or Busy
		Connection Cleared (see items #1 and #2)	Normal Clearing
D3 (and any other devices in call C1)	D3C1	Failed (Optional) (see item #3)	Blocked, or Normal or Busy
		Connection Cleared (see items #1 and #2)	Normal Clearing

17.1.7.3.3 Call-Type Monitoring Event Sequences

Table 17-36 Clear Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Failed (Optional) (see item #3)	Blocked, or Normal, or Busy
		Connection Cleared (see item #2)	Normal Clearing
	D2C1	Failed (Optional) (see item #3)	Blocked, or Normal, or Busy
		Connection Cleared (see item #2)	Normal Clearing
	D3C1	Failed (Optional) (see item #3)	Blocked, or Normal, or Busy
		Connection Cleared (see item #2)	Normal Clearing
	(D3C1)	Call Cleared, once the last device has left the call. (Note that this event is only sent for call-type monitors.)	Normal Clearing

1. For device-type monitors, it is always guaranteed that the Connection Cleared event associated with the monitored device is provided, and that this is the last Connection Cleared event for the cleared call over the monitor. Depending on how the switching function processes the service request, Connection Cleared events for the other devices in the call may precede this last Connection Cleared event.
2. The exact sequence of the Connection Cleared events for this service is not specified. This is due to the fact that when multiple devices are removed from a call as a result of the service, each switching function may process the clearing of those devices in a different order.
3. If a device stays off-hook and receives busy or blocked tone, the switching function sends the Failed event (event cause of Blocked or Busy) to indicate the status of the device, followed by a Connection Cleared event. The time between the generation of the Failed event and the Connection Cleared event is device and switching function specific and the time may be substantial. As for generation of the Failed event for a given monitor and device in the call, it follows the same conditions as the Connection Cleared events (if the event is supported by the switching function): No events reporting connection transitions for other devices in the call will be sent. However, the Connection Cleared event for the given monitor will be sent (and will be the last event associated with the cleared call sent over that monitor) when the connection transits to Null.

17.1.7.3.4 Functional Requirements

1. The Clear Call service shall only affect the callToBeCleared ConnectionID's call. Other calls that exist at the callToBeCleared ConnectionID's device remain unaffected.
2. The connection ID provided in the request may consist of only a callID portion. This is an exception to the rule of not providing deviceIDs in connectionIDs of service requests (see 6.1.5, "Connection", on page 24).

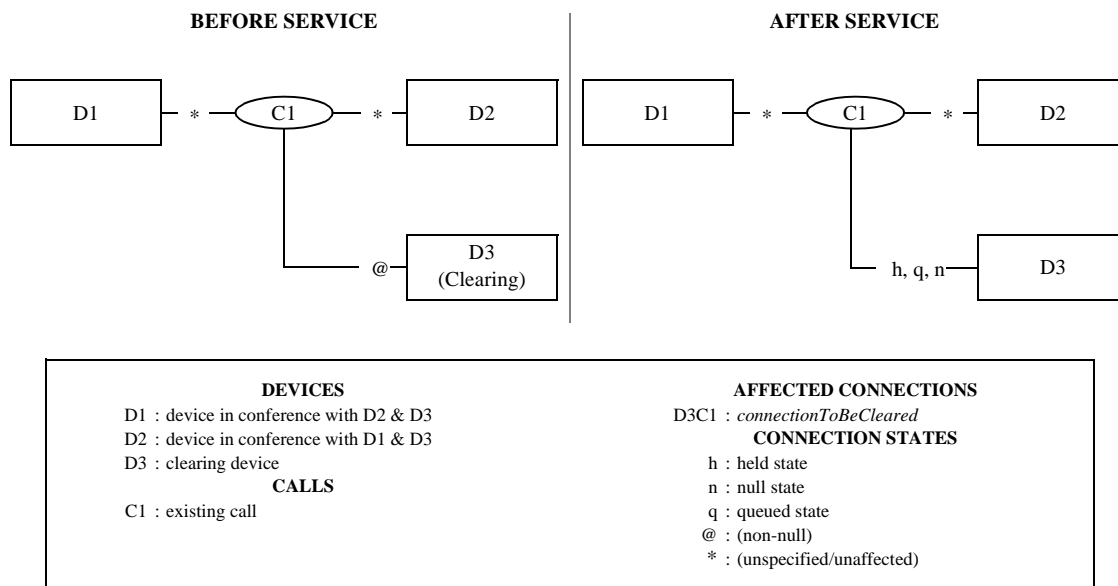
17.1.8 Clear Connection

C → S

The Clear Connection service releases a specific device from a call. In the case of a two-party call, this may result in the call being torn down. In the case of a conference call, this results in the specific party being removed from the conference. This service can also be used to inactivate a bridged appearance.

The Connection ID provided in the request is released.

Figure 17-8 Clear Connection Service



17.1.8.1 Service Request

Table 17-37 Clear Connection—Service Request

Parameter Name	Type	M/O/C	Description
connectionToBeCleared	ConnectionID	M	Specifies the connection to be cleared.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.8.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.8.2.1 Positive Acknowledgement

Table 17-38 Clear Connection—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.8.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, "ErrorValue", on page 65.

17.1.8.3 Operational Model

17.1.8.3.1 Connection State Transitions

Table 17-39 Clear Connection—Connection State Transitions

Connection	Initial State (Required)	Final State
D3C1 (connectionToBe-Cleared)	(Any non-Null state)	Failed (See item #1), Held (See item #4), Null, Queued (See item #5)
other Connections in call C1 (i.e., D1C1, D2C1)	(Unspecified)	Null, Failed (See item #3).

1. A connection may transition to the Failed state prior to Null if the device stays off-hook and receives busy or blocked tone. The time between the state transition of Failed -> Null is device and switching function specific and the time may be substantial.
2. The Clear Connection service is completed when the connectionToBeCleared connection transits to either the Null, Failed, Held, or Queued state.
3. If the call involves only two connections or the call has parties outside the switching sub-domain, the other connection(s) may transition to Null or Failed. If the call involves more than two connections, the other connections remain unaffected.
4. The Held state is valid when a call is “suspended” for a time period prior to going to Null.
5. The Queued state is only valid for a shared bridged device configuration when there are other appearances connected into the call.

17.1.8.3.2 Device-Type Monitoring Event Sequences

Table 17-40 Clear Connection—Device-Type Monitoring Event Sequences (Generic Case)

Monitored Device	Connection	Event	Event Cause
D3 (clearingdevice)	D3C1 (connectionToBeCleared)	Failed (see item #2)	Blocked, or Normal, or Busy
		Connection Cleared	Normal Clearing
D1	D3C1 (connectionToBe-Cleared)	Failed (see item #2)	Blocked, or Normal, or Busy
		Connection Cleared	Normal Clearing
D2 and any other devices in call C1	D3C1 (connectionToBe-Cleared)	Failed (see item #2)	Blocked, or Normal, or Busy
		Connection Cleared	Normal Clearing

Table 17-41 Clear Connection—Device-Type Monitoring Event Sequences Associated with Shared Bridged Configurations ¹

Monitored Device	Connection	Event	Event Cause
D3 (clearingdevice)	D3C1 (connectionToBeCleared)	Bridged	Normal
D1	D3C1 (connectionToBe-Cleared)	Bridged	Normal
D2 and any other devices in call C1	D3C1 (connectionToBe-Cleared)	Bridged	Normal

1. This event sequence only occurs when an appearance in the device configuration remains connected in the call after the service request.

Table 17-42 Clear Connection—Device-Type Monitoring Event Sequences Associated with Suspend

Monitored Device	Connection	Event	Event Cause
D3 (clearingdevice)	D3C1 (connectionToBeCleared)	Held	Suspend
		Connection Cleared	Normal
D1	D3C1 (connectionToBeCleared)	Held	Suspend
		Connection Cleared	Normal
D2 and any other devices in call C1	D3C1 (connectionToBeCleared)	Held	Suspend
		Connection Cleared	Normal

17.1.8.3.3 Call-Type Monitoring Event Sequences

Table 17-43 Clear Connection—Call-Type Monitoring Event Sequences (Generic Case)

Monitored Call	Connection	Event	Event Cause
C1	D3C1 (callToBeCleared)	Failed (see item #2)	Blocked, or Normal, or Busy
		Connection Cleared	Normal Clearing
		Call Cleared (see item #1)	

Table 17-44 Clear Connection—Call-Type Monitoring Event Sequences Associated with Shared Bridged Configurations

Monitored Call	Connection	Event	Event Cause
C1	D3C1 (callToBeCleared)	Bridged	Normal

Note that the above sequence only occurs when an appearance in the device configuration remains connected in the call after the service request.

Table 17-45 Clear Connection—Call-Type Monitoring Event Sequences with Suspend

Monitored Call	Connection	Event	Event Cause
C1	D3C1(callToBeCleared)	Held (see item #3)	Suspend
		Connection Cleared	Normal Clearing
		Call Cleared (see item #1)	Normal

1. For call-type monitors, if the connection being cleared is the last connection in the call then the Call Cleared event will be reported.
2. If a device stays off-hook and receives busy or blocked tone, the switching function sends the Failed event (event cause of Blocked or Busy) to indicate the status of the device, followed by a Connection Cleared event. The time between the generation of the Failed event and the Connection Cleared event is device and switching function specific and the time may be substantial.
3. Prior to going to Null, a switching function may “suspend” a cleared connection (and transition the connection to the Held state) for a time period. The time between the generation of the Held event and the Connection Cleared event is device and switching function specific and the time may be substantial.

17.1.8.3.4 Functional Requirements

1. If the call associated with Clear Connection service has multiple devices that are outside the switching sub-domain, they may remain connected after this service is complete.

2. When the last connected appearance in a shared bridged device configurations is cleared using the Clear Connection service, all other device configurations appearances are also cleared from the call (i.e., Connection Cleared events).
3. The switching function should only accept the Clear Call service request and not the Clear Connection service request if clearing a connection that is involved in a conference call causes the entire conference call to be terminated. However, some switching functions accept a Clear Connection service when it results in an entire conference call to be terminated. The capability exchange services indicate if the switching functions protects or allows a conference call from being torn down via the Clear Connection service.

17.1.9 Conference Call

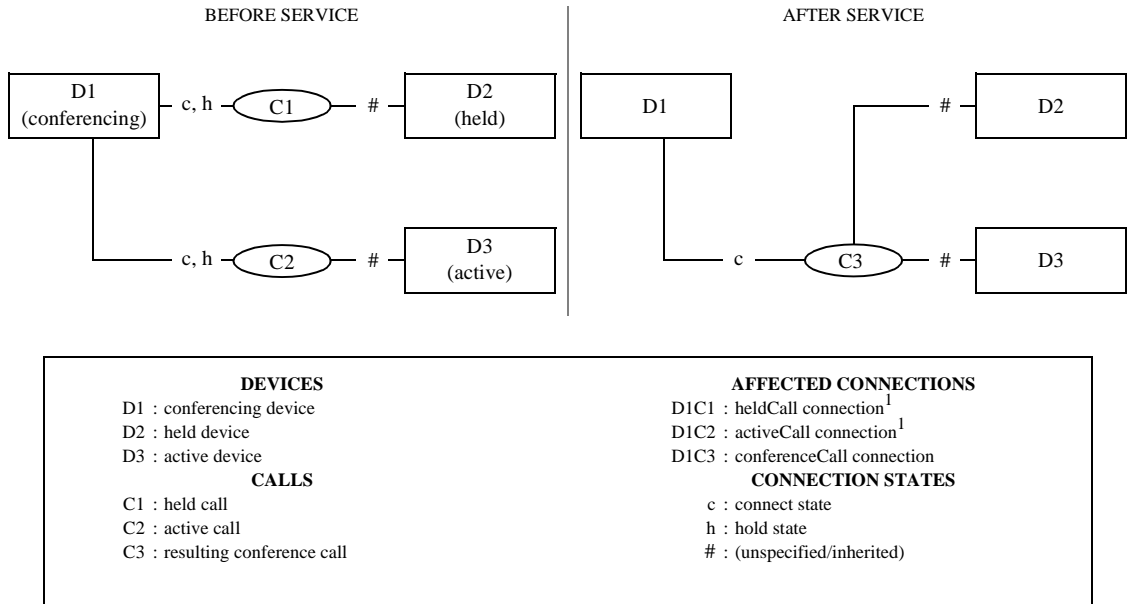
C → S

The Conference Call service provides a conference of an existing held call and another active call at a conferencing device.

The two calls are merged into a single call and the two connections at the conferencing device are resolved into a single connection. The Connection IDs formerly associated with the conferenced connections are released and a new Connection ID for the resulting connection is created.

The existing held call may consist of two or more devices.

Figure 17-9 Conference Call Service



1. See the Connection State Transitions for the appropriate initial states for these connections.

17.1.9.1 Service Request

Table 17-46 Conference Call—Service Request

Parameter Name	Type	M/O/C	Description
heldCall	ConnectionID	M	Specifies the held connection.
activeCall	ConnectionID	M	Specifies the active connection.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

In the case where both connections are in the Held state, one connection shall be the heldCall connection and the other connection shall be the activeCall connection.

17.1.9.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.9.2.1 Positive Acknowledgement

Table 17-47 Conference Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
conferenceCall	ConnectionID	M	Specifies the resulting connection to the new call. The ConnectionID shall have the CallID of the resulting conference call and the DeviceID of the conferencing device.
connections	ConnectionList	O	Specifies information on each endpoint/ConnectionID that has changed as a result of the service.
conferenceCallInfo	ConnectionInformation	O	Specifies the connection information associated with the conferenceCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.9.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.9.3 Operational Model

17.1.9.3.1 Connection State Transitions

Table 17-48 Conference Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (heldCall)	Hold	Null
	Connected, only if activeCall connectionID is in Connected state.	
D1C2 (activeCall)	Connected	Null
	Hold, only if heldCall connectionID is in the Hold state. See item #2.	
D2C1 and other connections in call C1	(unspecified)	Null
D1C3	Null	Connected
D3C2	(unspecified)	Null
other connections in call C3	Null	(Inherited from the corresponding connections in C1 and C2)

1. New ConnectionIDs will be assigned to all devices that remain in the call due to the Conference. The final connection states of the new ConnectionIDs will be inherited from the states of the corresponding original calls, except for the conferencing device.
2. D1C2 can be in the Hold state, if D1C1 is also in the Hold state. This allows the conferencing of two held calls.

17.1.9.3.2 Device-Type Monitoring Event Sequences

Table 17-49 Conference Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (Conferencing device)	(see item #2)	Conferenced	Normal
D2 and any other devices in call C1 (held call)	(see item #2)	Conferenced	Normal
D3 and any other devices in call C2 (active call)	(see item #2)	Conferenced	Normal

17.1.9.3.3 Call-Type Monitoring Event Sequences

Table 17-50 Conference Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	(see item #2)	Conferenced	Normal
C2	(see item #2)	Conferenced	Normal

1. For Device-Type Monitoring, a Connection Cleared event will not be seen for the activeCall or heldCall ConnectionIDs. The Conferenced event implies that the connections for these calls at this device are cleared.
2. There are multiple connections affected by this service.

17.1.9.3.4 Functional Requirements

1. The Conference service only affects the two calls specified on the service. If other calls exist at the conferencing device, they remain unaffected.
2. To get the connections for each of the devices to their initial states, the computing function can:
 - Use the Consultation Call service to place a call on hold and place a new call.
 - Use the Alternate Call service to place a call on hold and answer an alerting call.

In either of these cases, if the switching function supports the consultOptions parameter in these services, the computing function shall provide this parameter with a value of either “Conference Only” or “Unrestricted”.

Some switching function support a third approach for preparing for the Conference Call service involving the use of a hold service followed by a Make Call.

The fourth approach supported by some switching functions involves two held calls at the same device.

Certain switching functions also support a fifth approach involving two active calls at the same device.

The computing function should use the capabilities exchange services to determine which of these approaches is supported by the switching function.

3. If the computing function uses the Consultation Call service to specify the consultOptions parameter with a value of Transfer, and then attempts to complete the consultation call with a Conference Call service, it will be rejected with a negative acknowledgement.
4. The appearances in a shared bridged device configuration are unaffected by this service.
5. The maximum number of devices in a conference is subject to switching function limits.
6. The computing function should never assume the reuse of callIDs, although some switching functions may reuse one or the other.

17.1.10 Consultation Call

C → S

The Consultation Call service places an existing active call at a device on hold and initiates a new call from the same device. The existing active call may include two or more devices.

Figure 17-10 Consultation Call Service—Case A: Complete Dialling Sequence Provided

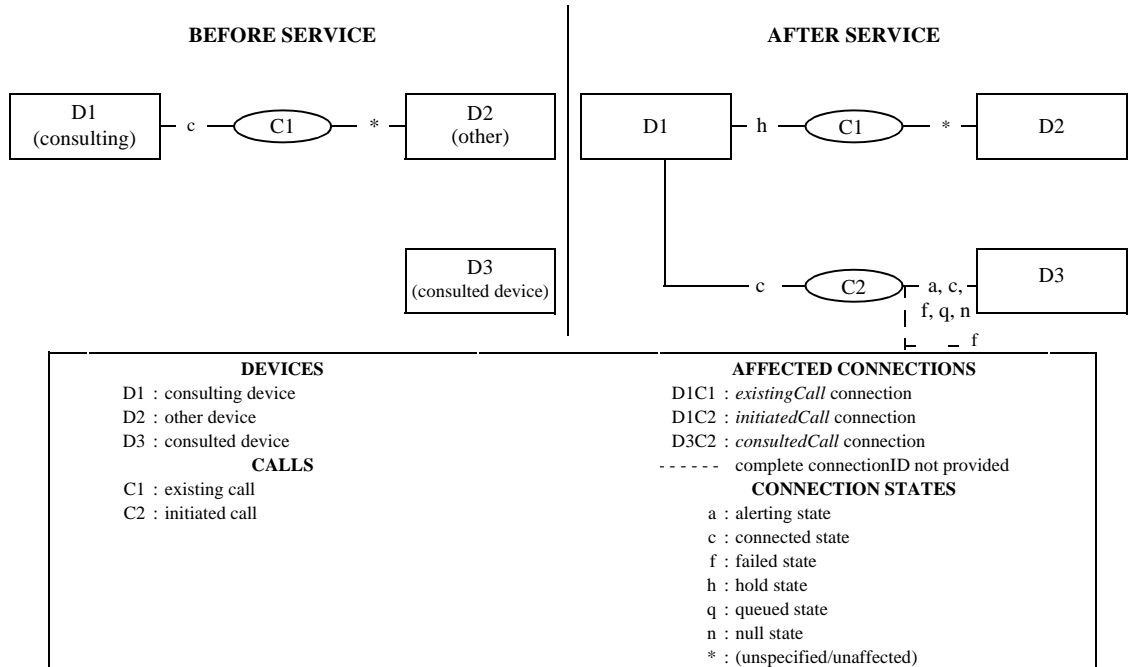
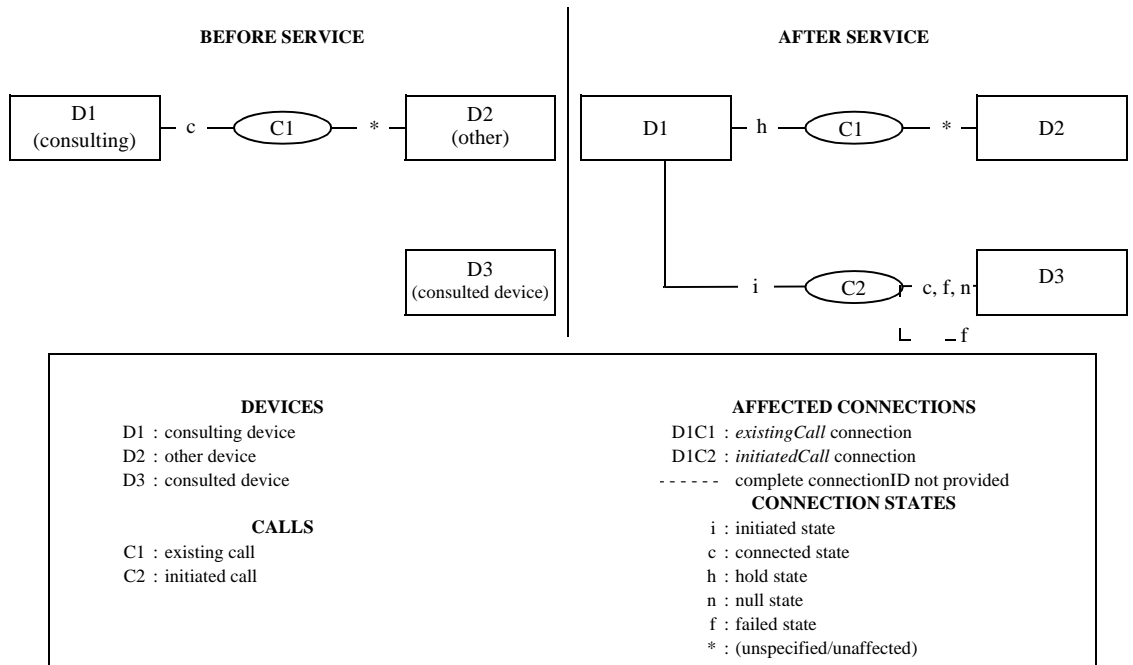


Figure 17-11 Consultation Call Service—Case B: Partial Dialling Sequence Provided



Note that connected state (D3C2) exists when D3 is a NID.

Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

17.1.10.1 Service Request

Table 17-51 Consultation Call—Service Request

Parameter Name	Type	M/O/C	Description
existingCall	ConnectionID	M	Specifies the active connection.
consultedDevice	DeviceID	M	Specifies the device to be consulted.
connectionReservation	Boolean	O	Specifies that the media stream channel(s) associated with the call being placed on hold be reserved for reuse at a later time. The complete set of possible values is: <ul style="list-style-type: none"> • True - channel(s) is to be reserved. • False - channel(s) is not to be reserved (default).
accountCode	AccountInfo	O	Specifies the account code to associate with the consulted call.
authCode	AuthCode	O	Specifies the authorization code to allow the call.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (Priority call, for example) to be associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values. If the supported characteristics cannot be honoured, the switching function shall reject the service request.
mediaCallCharacteristics	MediaCallCharacteristics	O	This specifies the media characteristics to be associated with the call being made for the consultation. If this parameter is not present then the media class is Voice.
callingConnectionInfo	ConnectionInformation	O	This specifies the connection information needed for the creation of the new connection at the consulting device. If this parameter is not present then the connection information is switching function specific.
consultOptions	Enumerated	C	This parameter indicates the potential actions following the Consultation Call so that certain facilities can be allocated prior to a Transfer or Conference. The complete set of possible values is: <ul style="list-style-type: none"> • Consult Only • Transfer Only • Conference Only • Unrestricted (default) If the switching function supports this parameter, the computing function shall supply one of the values supported. If the switching function does not support this parameter, the implicit value is “Unrestricted”. This parameter does not indicate a restriction or otherwise affect the switching function’s capabilities in any way with respect to services other than Transfer or Conference.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.10.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.10.2.1 Positive Acknowledgement

Table 17-52 Consultation Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
initiatedCall	ConnectionID	M	Specifies the initial connection to the new call. The ConnectionID shall have the CallID of the resulting new call and the DeviceID of the consulting device.
mediaCallCharacteristics	MediaCallCharacteristics	C	This specifies the adjusted media characteristics for the call being made for the consultation. This parameter shall be provided if the media characteristics have been adjusted, otherwise the parameter is optional.
InitiatedCallInfo	ConnectionInformation	O	This specifies the adjusted connection information used during the creation of the initiatedCall for the consulting device. If this parameter is not present then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.10.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.10.3 Operational Model

17.1.10.3.1 Connection State Transitions

Table 17-53 Consultation Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (existingCall)	Connected	Hold
D1C2 (initiatedCall)	Null	Initiated, Connected
D3C2 (consulted device)	Null	Alerting, Connected, Queued, Fail, or Null (Null if call moves away from the D3 (i.e., forwarded)).
Other connections in call C1 (existingCall) (i.e., D2C1)	(Unspecified)	(Unaffected; no transition due to this service).

1. When providing a partial dialling sequence (Case B), the connected state, for Device D3, only applies to external outbound calls. It indicates that enough of the dial string has been communicated for the switching function to connect the call to the network interface device that will be associated with the called device in the external network.

17.1.10.3.2 Device-Type Monitoring Event Sequences

There are two types of Device-Type Monitoring Event sequences depending on the dialling sequence used for the called device.

- Case A: The dialling sequence for the called device is the complete sequence for the device.

Table 17-54 Consultation Call—Device-Type Monitoring Event Sequences (Case A)

Monitored Device	Connection	Event	Event Cause
D1 (consulting device)	D1C1	Held	Normal or Consultation or Conference or Transfer ¹
	D1C2	Service Initiated (optional)	Consultation or Conference or Transfer ¹
	D1C2	Originated	Consultation or Conference or Transfer ¹
	D3C2	Events depend on the type of consulted device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). ²	
D2 and any other devices in C1 (original call)	D1C1	Held	Normal or Consultation or Conference or Transfer ¹
D3 (consulted device)	D3C2	Events depend on the type of consulted device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). ²	

1. The Conference event cause will only be present when the consultOptions parameter is set to “Conference Only.” The Transfer event cause will only be present when the consultOptions parameter is set to “Transfer Only.”
2. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

- Case B: The dialling sequence for the called device is a partial sequence for the device.

Table 17-55 Consultation Call—Device-Type Monitoring Event Sequences (Case B)

Monitored Device	Connection	Event	Event Cause
D1 (consulting device)	D1C1	Held	Normal or Consultation or Conference or Transfer ¹
	D1C2	Service Initiated (optional)	Consultation or Conference or Transfer ¹
	D1C2	Digits Dialed	Consultation or Conference or Transfer ¹
	D3C2	Events depend on the type of consulted device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). ²	
D2 and any other devices in C1 (original call)	D1C1	Held	Normal or Consultation or Conference or Transfer ¹
D3 (consulted device)	D3C2	Events depend on the type of consulted device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). ²	

1. The Conference event cause will only be present when the consultOptions parameter is set to “Conference Only.” The Transfer event cause will only be present when the consultOptions parameter is set to “Transfer Only.”
2. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

17.1.10.3.3 Call-Type Monitoring Event Sequences

There are two types of Device-Type Monitoring Event sequences depending on the dialling sequence used for the called device.

- Case A: The dialling sequence for the called device is the complete sequence for the device.

Table 17-56 Consultation Call—Call-Type Monitoring Event Sequences (Case A)

Monitored Call	Connection	Event	Event Cause
C1 (original call)	D1C1	Held	Normal or Consultation or Conference or Transfer ¹
C2 (initiated call)	D1C2	Service Initiated (optional)	Consultation or Conference or Transfer ¹
	D1C2	Originated	Consultation or Conference or Transfer ¹
	D3C2	Events depend on the type of consulted device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). ²	

1. The Conference event cause will only be present when the consultOptions parameter is set to “Conference Only.” The Transfer event cause will only be present when the consultOptions parameter is set to “Transfer Only.”
2. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

- Case B: The dialling sequence for the called device is a partial sequence for the device.

Table 17-57 Consultation Call—Call-Type Monitoring Event Sequences (Case B)

Monitored Call	Connection	Event	Event Cause
C1 (original call)	D1C1	Held	Normal or Consultation or Conference or Transfer ¹
C2 (initiated call)	D1C2	Service Initiated (optional)	Consultation or Conference or Transfer ¹
	D1C2	Digits Dialed	Consultation or Conference or Transfer ¹
	D3C2	Events depend on the type of consulted device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). ²	

1. The Conference event cause will only be present when the consultOptions parameter is set to “Conference Only.” The Transfer event cause will only be present when the consultOptions parameter is set to “Transfer Only.”
2. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

17.1.10.3.4 Functional Requirements

1. For the consultedDevice, all active features for this device will be honoured while the call is being made to it.
2. The consultOptions parameter indicates the potential action following the Consultation Call service so that certain facilities can be allocated if a transfer or conference is desired. If the switching function supports the consultOptions parameter, the computing function shall provide one of the supported values of this parameter obtained through the capability exchange services. If the switching function is unable to allocate the facilities for the requested Transfer or Conference, it will reject the Consultation Call service request. This parameter does not indicate a restriction or otherwise affect the switching function’s capabilities in any way with respect to services other than Transfer or Conference.
3. It is switching function specific whether a switching function may still accept a request for a Conference or Transfer if the consultOptions of Conference or Transfer was not requested as part of the Consultation Call service.
4. If the computing function specifies the consultOptions parameter with a value of Conference, then the computing function can not complete the consultation with a Transfer Call service or transfer feature.

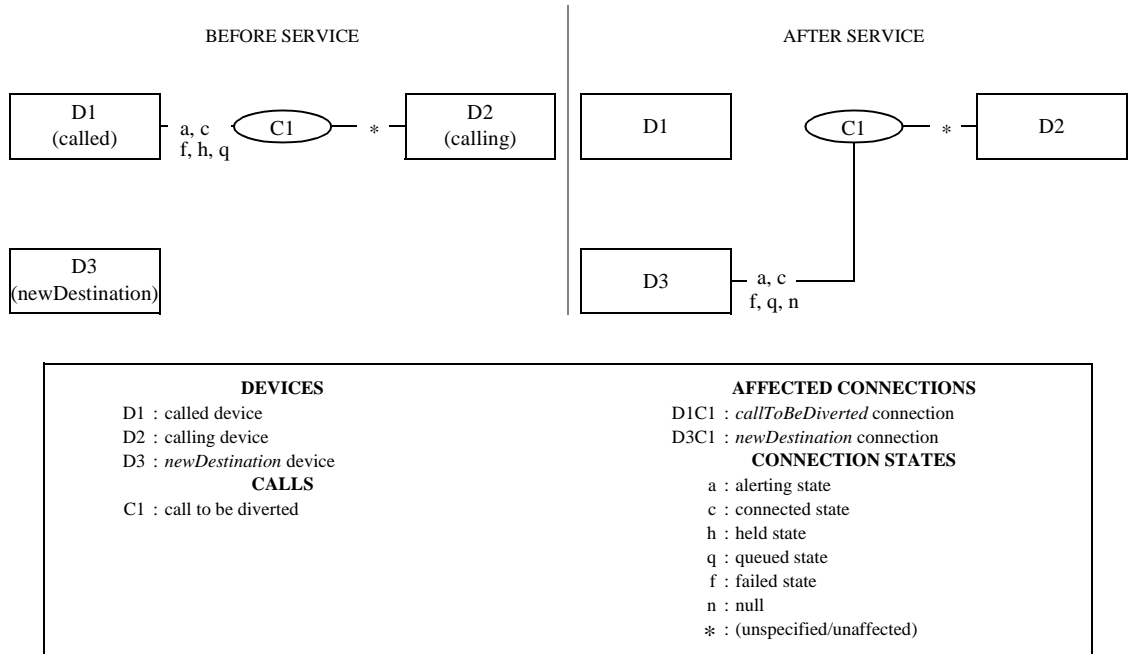
5. If the computing function specifies the consultOptions parameter with a value of Transfer, then the computing function can not complete the consultation with a Conference Call service or conference feature.
6. The consultedDevice parameter may contain a device identifier of null or contain a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the “;” character) at the end of it. When this parameter is used in this manner, the computing function is indicating that it wishes to stage the dialling sequence. The completion of the dialling sequence can be accomplished either by entering the rest of the sequence manually at the actual device or the computing function can use the Dial Digits service to complete the sequence. As for the other types of consultedDevice parameter, they shall contain a complete dialling sequence. The switching function may have a time-out period for multi-stage dialling. If the dialling sequence does not complete prior to this timeout, it may either abort the call or attempt to use the digits already dialled and signal that dialling is complete with an originated event.
7. If the Consultation Call service is used to initiate a multistage dialling sequence, the computing function is signalled to continue the dialling sequence via either the Service Initiated event (i.e., the consultedDevice is Null) or the Digits Dialled event (i.e., the consultedDevice has a partial dialling sequence character).
8. A feature of initiating a digital data call as a result of this service (i.e., initiatedCall) is that the switching function may or may not adjust the digital data characteristics (e.g., connection rate) and connection information (e.g., number of channels) that were supplied on the Consultation Call service request (Use the capabilities exchange services to determine which feature the switching function supports). If the switching function does not support the adjusting of the characteristics/connection information, the service request will be rejected with an appropriate error code in the negative acknowledgement. If the switching function does support the adjusting of the characteristics/connection information, the positive acknowledgement will contain the adjusted value or values. If the computing function determines that the adjusted values are not adequate, it can terminate the digital data call (e.g., Clear Call).
9. If the computing function makes a digital data call using this service and wants to also bind a particular Media Service to the call, then the computing function shall use the Media Attach service.

17.1.11 Deflect Call

C → S

The Deflect Call service allows the computing function to divert a call to another destination that may be inside or outside the switching sub-domain.

Figure 17-12 Deflect Call Service



Note that D1 may also be the calling device.

17.1.11.1 Service Request

Table 17-58 Deflect Call—Service Request

Parameter Name	Type	M/O/C	Description
callToBeDiverted	ConnectionID	M	Specifies the connection to be diverted.
newDestination	DeviceID	M	Specifies the device to which the call is to be diverted (newDestination device).
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.11.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.11.2.1 Positive Acknowledgement

Table 17-59 Deflect Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.11.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.11.3 Operational Model

17.1.11.3.1 Connection State Transitions

Table 17-60 Deflect Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callToBeDiverted)	Alerting, Connected, Failed, Held, Queued	Null
D2C1 (calling device)	(Unspecified)	Unaffected; no transition due to this service.
D3C1 (newDestination)	Null	Alerting, Connected, Queued, Fail, or Null (Null if call moves away from D3 (i.e., forwarded)).

17.1.11.3.2 Device-Type Monitoring Event Sequences

Table 17-61 Deflect Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (called device)	D1C1 (callToBeDiverted)	Diverted (see item #3)	Redirected or Normal
D2 (calling device)	D1C1	Diverted (optional) (see items #2 and #3)	Redirected or Normal
	D3C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #4)	
D3 (newDestination device)	D3C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #4)	

17.1.11.3.3 Call-Type Monitoring Event Sequences

Table 17-62 Deflect Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Diverted (optional) (see items #1 and #3)	Redirected or Normal
	D3C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #4)	

1. The switching function provides the Diverted event for C1 only if it is providing Diverted events for call-type monitors. This is indicated through the capabilities exchange services.

2. The switching function provides the Diverted event for D2 only if it is providing Diverted events for all devices in the call. This is indicated through the capabilities exchange services.
3. A Connection Cleared Event will not be provided for D1 (called device). The Diverted event implies the ConnectionID has gone to Null.
4. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

17.1.11.3.4 Functional Requirements

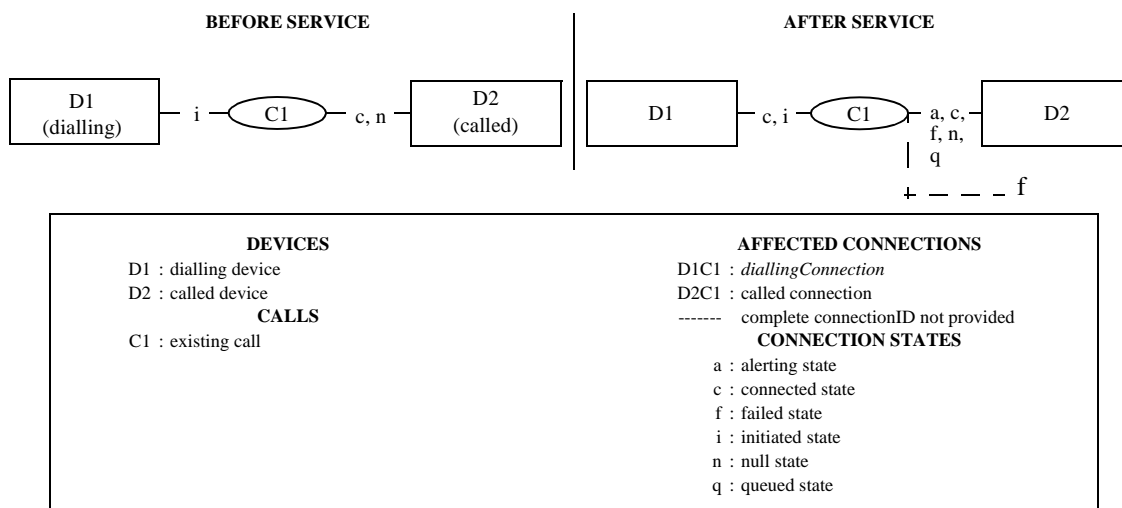
1. The Deflect Call service differs from the (Directed and Group) Pickup services in that the Pickup services redirects a call from a device and connects it to a specified device.
2. The Deflect Call service only affects the callToBeDiverted ConnectionID's call. If other calls exist at the callToBeDiverted ConnectionID's device, they remain unaffected.
3. For the newDestination, all active features for this device will be honoured while the call is being deflected to it. For example, due to active features at the called device, the call may or may not be deflected to the new destination device.
4. As a result of the Deflect Call service, the call ID associated with this call remains unchanged.
5. If the callToBeDiverted connection identifier is associated with a shared bridged device configuration appearance in the queued state (i.e., inactive mode), then the service request will be rejected with a negative acknowledgement.
6. This service request does not support the use of a null device identifier or a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the ";" character) in it for the newDestination parameter. A complete and valid device identifier shall be provided, otherwise the switching function will reject the service request with a negative acknowledgement.
7. If this service is used to deflect a digital data call, the connection for the newDestination will inherit the characteristics of the call.

17.1.12 Dial Digits

C → S

The Dial Digits service allows the computing function to perform a dialling sequence that is associated with a call that has already been initiated (i.e., has manually gone off-hook or has been initiated via a Make Call or Consultation Call service). This service is also used to perform the dialling sequences associated with completing a multi-stage dialled call.

Figure 17-13 Dial Digits Service



Note that the initial state of connected for connection D2C1 exists when D2 is a NID.

Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

17.1.12.1 Service Request

Table 17-63 Dial Digits—Service Request

Parameter Name	Type	M/O/C	Description
diallingConnection	ConnectionID	M	Specifies the connection which is dialling the digits.
diallingSequence	DeviceID	M	Specifies the actual string of digits to be dialled. To specify a partial dialling sequence, the Diallable Digits format (DD) of the DeviceID with the “;” character as the last digit in the string shall be used.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.12.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.12.2.1 Positive Acknowledgement

Table 17-64 Dial Digits—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.

Table 17-64 Dial Digits—Positive Acknowledgement (continued)

Parameter Name	Type	M/O/C	Description
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.12.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.12.3 Operational Model

17.1.12.3.1 Connection State Transitions

Table 17-65 Dial Digits—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (dialling)	Initiated	Initiated, Connected
D2C1 (called)	Null, Connected (The connected state only applies to external outbound calls. It indicates that enough of the dial string has been communicated for the switching function to connect the call to the network interface device that will be associated with the called device in the external network.)	Alerting, Connected, Failed, Queued, or Null (Null if call moves away from D3 (i.e., forwarded).

17.1.12.3.2 Device-Type Monitoring Event Sequences

Table 17-66 Dial Digits—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Cause Code
D1 (dialling device)	D1C1	Digits Dialed (see item #1)	Normal or Network Dialling
	D1C1	Originated (when the dialling sequence is complete and the D1C1 connection is connected into the call (i.e., D1C1 final state = connected)).	Normal or Network Dialling
	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #2)	
D2 (called device)	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #2)	

17.1.12.3.3 Call-Type Monitoring Event Sequences

Table 17-67 Dial Digits—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Cause Code
C1	D1C1	Digits Dialed (see item #1)	Normal or Network Dialling
	D1C1	Originated (when the dialling sequence is complete and the D1C1 connection is connected into the call (i.e., D1C1 final state = connected))	Normal or Network Dialling
	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #2)	

1. This event will be provided for each diallable digits segment received by the switching function.
2. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

17.1.12.3.4 Functional Requirements

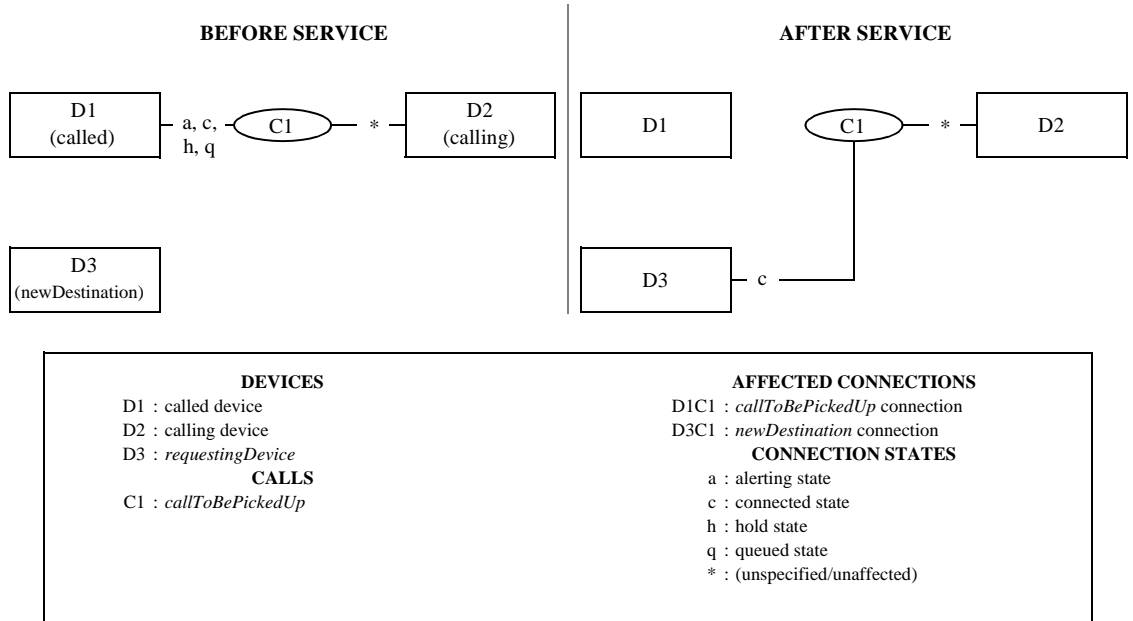
1. The Dial Digits service only effects the diallingConnection call. If other calls exist at the diallingConnection Connection ID's device, they remain unaffected.
2. This service can not be used to generate DTMF tones on an existing connected call. The Generate Digits service is used for this. This service is only used to perform the dialling sequences for placing a call (i.e., outbound dialling).
3. If no digits have been dialled for the diallingConnection through the original Make Call service or a previous Dial Digits service, the diallingSequence parameter can be any format of device identifier. Otherwise the diallingSequence parameter shall be in Diallable Digits format.
4. If the diallingSequence parameter is intended to provide just a portion of a longer dialling sequence to follow, then it shall be in Diallable Digits format with the last character of the sequence being “;”. Otherwise the diallingSequence parameter will be interpreted as the last in the dialling sequence for the diallingConnection.
5. The premature termination of a dialling sequence can be done by either using the Clear Connection service, Clear Call service, or having the dialling connection manually cleared.
6. The switching function may have a time-out period for multi-stage dialling. If the dialling sequence does not complete prior to this timeout, it may either abort the call or attempt to use the digits already dialled and signal that dialling is complete with an originated event.
7. If the switching function determines that dialling is complete even if a “;” was supplied, it will originate the call and ignore any subsequent digits.

17.1.13 Directed Pickup Call

C → S

The Directed Pickup Call service moves a specified call and connects it at a new specified destination. This results in the connection being diverted to a new destination inside the switching sub-domain.

Figure 17-14 Directed Pickup Call Service



Note that D1 could also be the calling device.

17.1.13.1 Service Request

Table 17-68 Directed Pickup Call—Service Request

Parameter Name	Type	M/O/C	Description
callToBePickedUp	ConnectionID	M	Specifies the connection to be picked up.
requestingDevice	DeviceID	M	Specifies the device which is picking up the call.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.13.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.13.2.1 Positive Acknowledgement

Table 17-69 Directed Pickup Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
pickedCall	ConnectionID	O	Specifies the connectionID for the device which is picking up the call.

Table 17-69 Directed Pickup Call—Positive Acknowledgement (continued)

Parameter Name	Type	M/O/C	Description
pickedCallInfo	ConnectionInformation	O	Specifies the connection information associated with the pickedCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.13.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.13.3 Operational Model

17.1.13.3.1 Connection State Transitions

Table 17-70 Directed Pickup Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callToBePickedUp)	Alerting, Connect, Hold, or Queued	Null
D2C1 (calling device)	(Unspecified)	(Unaffected; no transition due to this service.)
D3C1 (requestingDevice)	Null	Connected

17.1.13.3.2 Device-Type Monitoring Event Sequences

Table 17-71 Directed Pickup Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (called device)	D1C1 (callToBePickedUp)	Diverted (see item #2)	Call Pickup
D2 (calling device)	D1C1	Diverted (optional) (see items #1 and #2)	Call Pickup
	D3C1 (see item #5)	Service Initiated (optional) (see item #5)	Call Pickup (prompting)
	D3C1	Established	Call Pickup
D3 (requestingDevice)	D3C1	Service Initiated (optional) (see item #5)	Call Pickup (prompting)
	D3C1	Established	Call Pickup

17.1.13.3.3 Call-Type Monitoring Event Sequences

Table 17-72 Directed Pickup Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Diverted (optional) (see Items #2 and #4)	Call Pickup
	D3C1 (item #5)	Service Initiated (optional) (item #5)	Call Pickup (prompting)
	D3C1	Established	Call Pickup

1. The switching function provides the Diverted event for D2 only if it is providing Diverted events for all devices in the call. This is indicated through the capabilities exchange services.
2. A Connection Cleared event will not be provided. The Diverted event implies the ConnectionID (for D1C1) has gone to Null.

3. If there is already a connection in the initiated state at the requestingDevice device, the computing function may receive a Connection Cleared event for that connection and an Established event for the connection involving the picked call.
4. The switching function provides the Diverted event for C1 only if it is providing Diverted events for call-type monitors. This is indicated through the capabilities exchange services.
5. The Service Initiated event is dependent upon the prompting mode (as described in 6.7.10, “Prompting”, on page 46).
 - For the “prompting is a pre-condition of the service” mode, the Service Initiated is generated before any service specific events and is not part of the service completion criteria. The connectionID associated with the Service Initiated event is not associated with the Directed Pickup service.
 - For the “prompting is part of the service” mode”, the Service Initiated event is part of the service completion criteria, and is generated after the Diverted event and contains the same connectionID as the Diverted and Established events.
6. If a Bridged event is generated for an Independent Shared Bridged device configuration (see Functional Requirement #4), it is not part of the service completion criteria.

17.1.13.3.4 Functional Requirements

1. This service differs from the Deflect Call service in that the Deflect Call service redirects a call to another destination (in which the resulting state of the new destination depends on the destination’s type and active features). The Directed Pickup Call service redirects a call to another destination which is immediately connected to the call.
2. For the newDestination, all features such as Forwarding and Do Not Disturb for this device will be ignored while the call is being redirected to it.
3. As a result of the Directed Pickup service, the call ID associated with this call remains unchanged.
4. For Shared Bridged device configurations, when a call is picked from an appearance (callToBePickedUp) all appearances will be cleared from the call (i.e., Connection Cleared events with a cause of Normal), except when the call that is being picked is part of an Independent Shared Bridged device configuration and the appearance from which the call is being picked is not the last appearance connected into the call. In this case the appearance from which the call is being picked will return to the inactive mode (i.e., Bridged event with a cause of Normal) and the other appearances are unaffected. For more information, refer to Annex B.2.3, “Shared-Bridged”.
5. This service request does *not* support the use of a null device identifier or a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the “;” character) in it for the newDestination parameter. A complete and valid device identifier shall be provided, otherwise the switching function will reject the service request with a negative acknowledgement.
6. This service is used when the device associated with the callToBePickedUp connection ID is different from the newDestination device. If the devices are the same, then the service is rejected. The Answer Call and the Retrieve Call services should be used instead.

17.1.14 Group Pickup Call

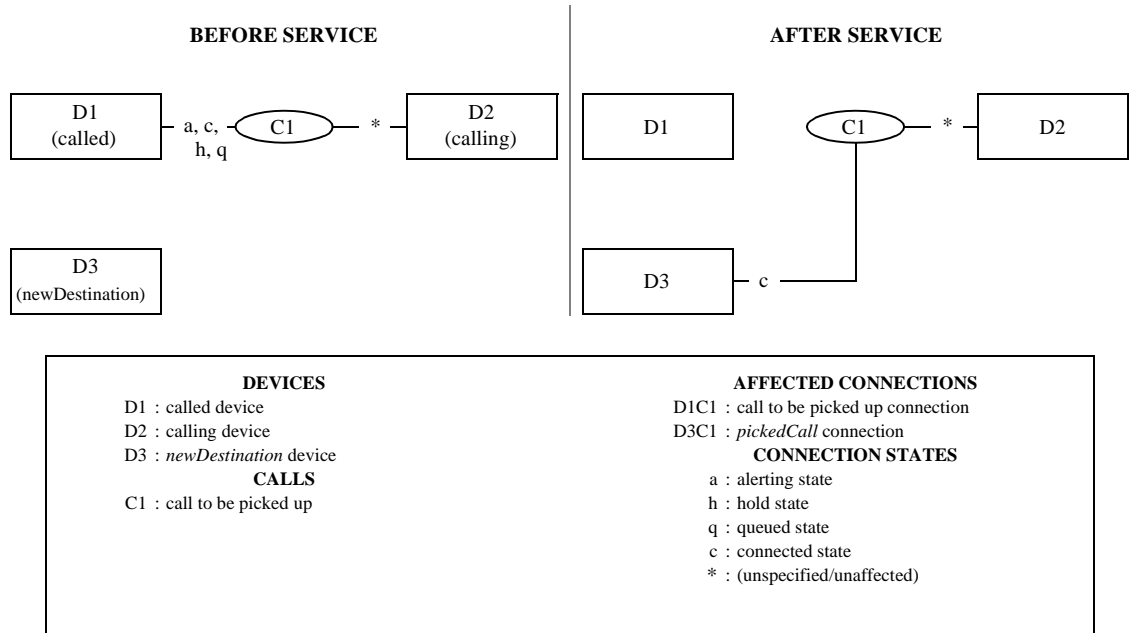
C → S

The Group Pickup Call service moves a call that is a member of a specified or default pickup group to a new specified destination.

This results in a connection in a pickup group to be connected to a new specified destination inside the switching sub-domain.

Note that the difference between this service and the Directed Pickup Call service is that Directed Pickup Call service specifies the actual connection to be picked up whereby the Group Pickup Call service does not.

Figure 17-15 Group Pickup Call Service



Note that D1 may also be the calling device.

17.1.14.1 Service Request

Table 17-73 Group Pickup Call—Service Request

Parameter Name	Type	M/O/C	Description
newDestination	DeviceID	M	Specifies the device which is picking up the call.
pickGroup	DeviceID	O	Specifies the pick group. If this parameter is not provided, the switching function may use a pick group associated with the newDestination device.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.14.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.14.2.1 Positive Acknowledgement

Table 17-74 Group Pickup Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
pickedCall	ConnectionID	O	Specifies the connection ID of call C1 at device D3.
pickedCallInfo	ConnectionInformation	O	Specifies the connection information associated with the pickedCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.14.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.14.3 Operational Model

17.1.14.3.1 Connection State Transitions

Table 17-75 Group Pickup Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (call to be picked up)	Alerting, Connected, Hold, or Queued	Null
D2C1 (calling device)	(Unspecified)	(Unaffected; no transition due to this service.)
D3C1 (pickedCall)	Null	Connected

17.1.14.3.2 Device-Type Monitoring Event Sequences

Table 17-76 Group Pickup Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (called device)	D1C1 (callToBepickedUp)	Diverted (see item #1)	Call Pickup
D2 (calling device)	D1C1	Diverted (optional) (see item #1 and #2)	Call Pickup
	D3C1 (see item 2)	Service Initiated (optional) (see item #5)	Call Pickup (prompting)
	D3C1	Established	Call Pickup
D3 (newDestination device)	D3C1	Service Initiated (optional)	Call Pickup (prompting)
	D3C1	Established	Call Pickup

17.1.14.3.3 Call-Type Monitoring Event Sequences

Table 17-77 Group Pickup Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Diverted (optional) (see item #1 and #3)	Call Pickup
	D3C1 (see item 2)	Service Initiated (optional) (see item 5)	Call Pickup (prompting)
	D3C1	Established	Call Pickup

1. A Connection Cleared event will not be provided. The Diverted event implies the ConnectionID (for D1C1) has gone to Null.

2. The switching function provides the Diverted event for D2 only if it is providing Diverted events for all devices in the call. This is indicated through the capabilities exchange services.
3. The switching function provides the Diverted event for C1 only if it is providing Diverted events for call-type monitors. This is indicated through the capabilities exchange services.
4. If there is already a connection in the initiated state at the newDestination device, the computing function may receive a Connection Cleared event for that connection and an Established event for the connection involving the picked call.
5. The Service Initiated event is dependent upon the prompting mode (as described in 6.7.10, “Prompting”, on page 46).
 - For the “prompting is a pre-condition of the service” mode, the Service Initiated is generated before any service specific events and is not part of the service completion criteria. The connectionID associated with the Service Initiated event is not associated with the Group Pickup service.
 - For the “prompting is part of the service” mode”, the Service Initiated event is part of the service completion criteria, and is generated after the Diverted event and contains the same connectionID as the Diverted and Established events.
6. If a Bridged event is generated for an Independent Shared Bridged device configuration (see Functional Requirement #6), it is not part of the service completion criteria.

17.1.14.3.4 Functional Requirements

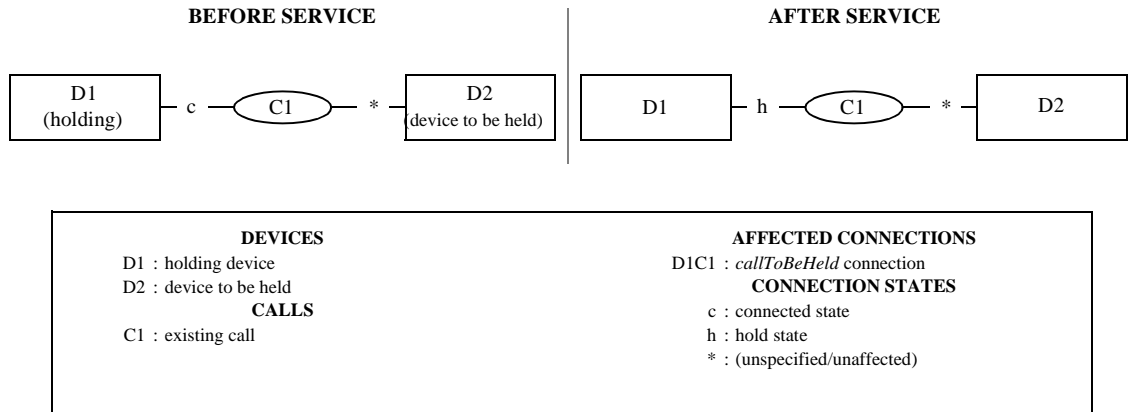
1. This service differs from the Deflect Call service in that the Deflect Call service redirects a call to another destination (in which the resulting state of the new destination depends on the destination’s type and active features). The Group Pickup Call service redirects a call to another destination which is immediately connected to the call.
2. The Group Pickup service is administered by the switching function. The switching function determines which call the newDestination connects to by first determining which group the newDestination belongs to (either as specified by the pickGroup parameter or by a switching function administered group associated with the newDestination device) and then connecting it to the appropriate call.
3. For the newDestination, all features such as Forwarding and Do Not Disturb for this device will be ignored while the call is being redirected to it.
4. As a result of the Group Pickup service, the call ID associated with this call remains unchanged.
5. This service request does *not* support the use of a null device identifier or a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the “;” character) in it for the newDestination or pickGroup parameters. A complete and valid device identifier shall be provided, otherwise the switching function will reject the service request with a negative acknowledgement.
6. For Shared Bridged device configurations, when a call is picked from an appearance all appearances will be cleared from the call (i.e., Connection Cleared events with a cause of Normal), except when the call that is being picked is part of an Independent Shared Bridged device configuration and the appearance from which the call is being picked is not the last appearance connected into the call. In this case the appearance from which the call is being picked will return to the inactive mode (i.e., Bridged event with a cause of Normal) and the other appearances are unaffected. For more information, refer to Annex B.2.3, “Shared-Bridged”.

17.1.15 Hold Call

C → S

The Hold Call service places a connected connection on hold at the same device.
This service interrupts communication for an existing call at a device.

Figure 17-16 Hold Call Service



17.1.15.1 Service Request

Table 17-78 Hold Call—Service Request

Parameter Name	Type	M/ O/C	Description
callToBeHeld	ConnectionID	M	Specifies the active connection to be held.
connectionReservation	Boolean	O	Specifies that the media stream channel(s) associated with the call being placed on hold be reserved for reuse at a later time. The complete set of possible values is: <ul style="list-style-type: none"> • True - channel(s) is to be reserved. • False - channel(s) is not to be reserved (default).
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information.

17.1.15.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.15.2.1 Positive Acknowledgement

Table 17-79 Hold Call—Positive Acknowledgement

Parameter Name	Type	M/ O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information.

17.1.15.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.15.3 Operational Model

17.1.15.3.1 Connection State Transitions

Table 17-80 Hold Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callToBeHeld)	Connected	Hold
other connections in the call (i.e., D2C1)	(Unspecified)	(Unaffected; no transition due to this service).

17.1.15.3.2 Device-Type Monitoring Event Sequences

Table 17-81 Hold Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (Holding device)	D1C1 (callToBeHeld)	Held	Normal
D2 (device to be held and any other connections in call C1)	D1C1 (callToBeHeld)	Held	Normal

17.1.15.3.3 Call-Type Monitoring Event Sequences

Table 17-82 Hold Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1(callToBe-Held)	Held	Normal

1. If a Bridged event is generated for an Independent Shared Bridged device configuration (see Functional Requirement #2), it is not part of the service completion criteria.

17.1.15.3.4 Functional Requirements

1. The switching function may have a time-out period for the call once it is held. If the call is not retrieved before the time-out period ends, then the held call may ringback either the holding device or a device predefined by the switching function (such as an attendant console, for example).
2. For Shared Bridged device configurations, when a call is held by an appearance (callToBeHeld) all appearances will transition to the hold state (i.e., Held events with a cause of Normal), except when the call that is being held is part of an Independent Shared Bridged device configuration and the appearance that is holding the call is not the last appearance connected into the call. In this case only the appearance holding the call will transition to the hold state and the other appearances are unaffected. For more information, refer to Annex B.2.3, “Shared-Bridged”.
3. As a result of placing a connection associated with a digital data call (if supported by the switching function) on hold, the other calls at the device or device configuration are unrelated with this call.

17.1.16 Intrude Call

C → S

The Intrude Call service adds the calling device to a call at a busy called device. Depending upon the switching function, the result will be that the calling device is either actively or silently participating in the called device's existing call or consulting with the called device with a new call

There are two cases specified for the Intrude Call service:

- Case A: In this case, the calling device (D1) joins call C2 with devices D2 and D3.
- Case B: In this case, the called device (D2) places its existing active call on hold first and then connects to the new calling device (D1).

Figure 17-17 Intrude Call Service (Case A)

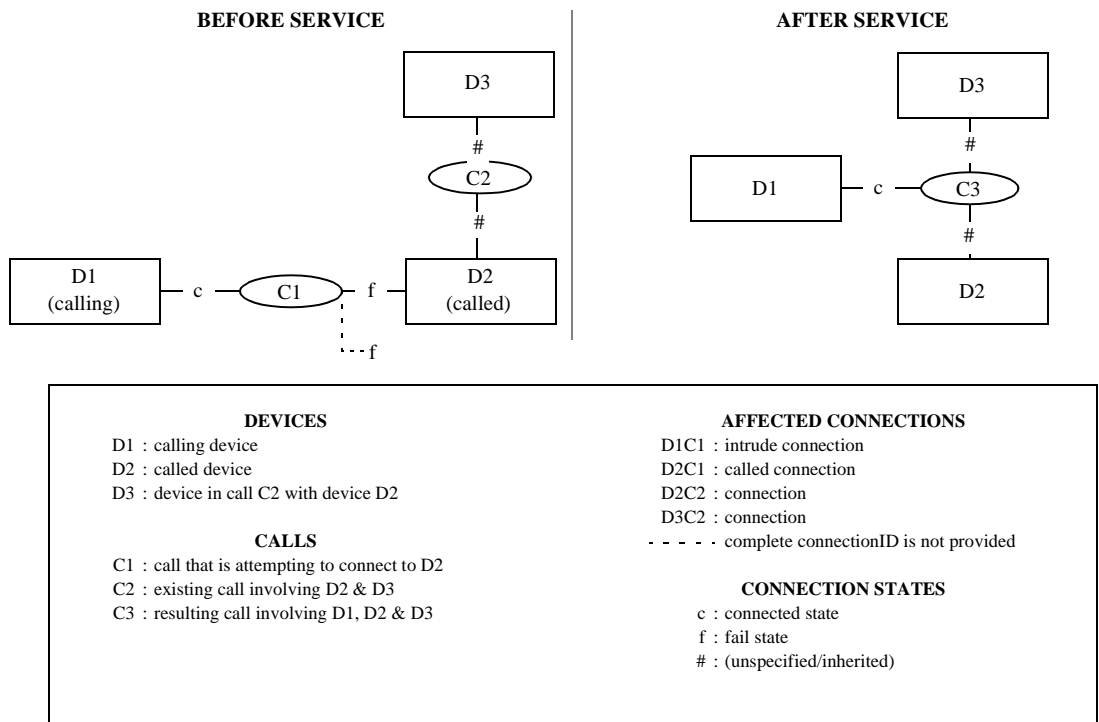
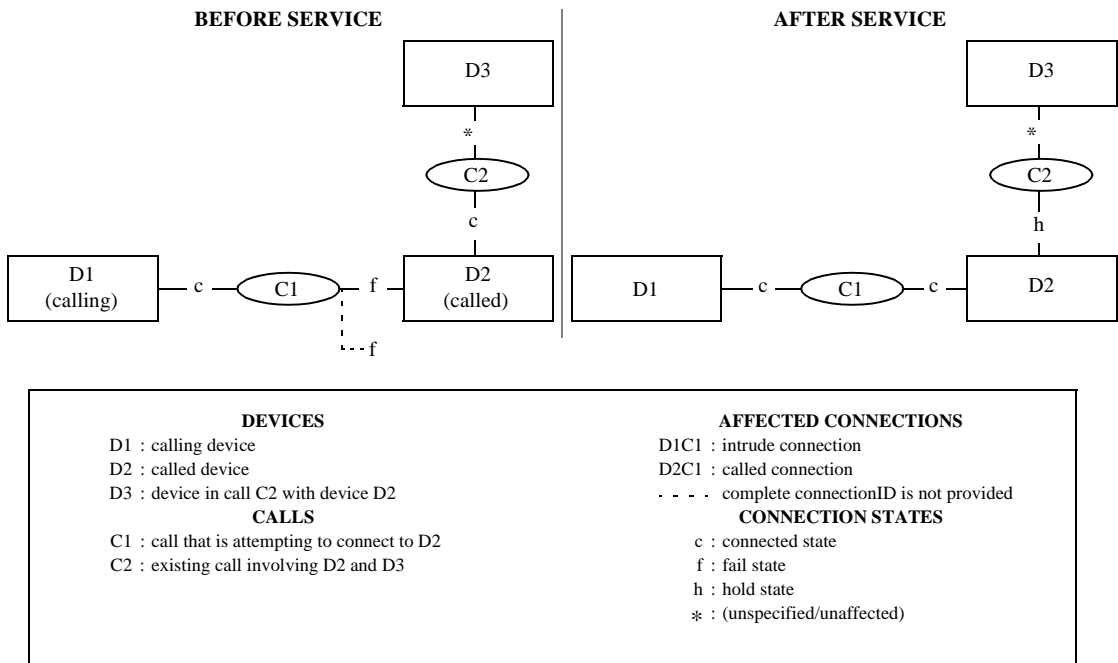


Figure 17-18 Intrude Call Service (Case B)



Refer to section 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

17.1.16.1 Service Request

Table 17-83 Intrude Call—Service Request

Parameter Name	Type	M/O/C	Description
intrude	ConnectionID	M	Specifies the connection of the calling device.
participationType	Enumerated	O	Specifies the type of participation the added device has in the resulting call. The complete set of possible values is: <ul style="list-style-type: none"> • active (default) - The added device actively participates in the resulting conference call. • silent - The added device can listen but cannot actively participate in the resulting conference call. If the switching function only supports “Case B” of the service and the participationType parameter is supplied, then the service will be rejected by the switching function.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.16.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.16.2.1 Positive Acknowledgement

Table 17-84 Intrude Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
conferencedCall	ConnectionID	C	ConnectionID includes the CallID of the resulting call and the DeviceID of the calling device. Mandatory, if the Intrude Call service creates a new ConnectionID for the calling device as in Case A; otherwise, the parameter is not provided.
conferencedCallInfo	ConnectionInformation	O	Specifies the connection information associated with the conferencedCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information.

17.1.16.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.16.3 Operational Model

17.1.16.3.1 Connection State Transitions

Table 17-85 Intrude Call—Connection State Transitions

Connection	Initial State (Required)	Final State	
		Case A	Case B
D2C1 (called device)	Fail	Null	Connected
D1C1 (intrude)	Connected	Null	Connected
D1C3	Null	Connected	N/A
D2C2	Case A: (Unspecified) Case B: Connected	Null	Hold
D3C2	(Unspecified)	Null	(Unaffected)
D2C3 and all connections in call C3 that had corresponding connections in call C2.	Null	(Inherited from corresponding connections in call C2)	N/A

17.1.16.3.2 Device-Type Monitoring Event Sequences

- Case A: D1, the calling device, joins call C3 with devices D2 and D3.

Table 17-86 Intrude Call—Device-Type Monitoring Event Sequences (Case A)

Monitored Device	Connection	Event	Event Cause
D1 (calling device)	D1C3	Conferenced	Override, Silent Participation, or Active Participation
D2 (called device)	D2C3	Conferenced	Override, Silent Participation, or Active Participation
other devices in original call C2 (i.e., D3)	D3C3	Conferenced	Override, Silent Participation, or Active Participation

- Case B: The called device places its existing active call on hold first and then connects to the new calling device (D1).

Table 17-87 Intrude Call—Device-Type Monitoring Event Sequences (Case B)

Monitored Device	Connection	Event	Event Cause
D1 (calling device)	D2C1 (called)	Established	Intrude
D2 (called device)	D2C2	Held	Normal or Intrude
	D2C1 (called)	Established	Intrude
Other devices in original call C2 (i.e., D3)	D2C2	Held	Normal or Intrude

17.1.16.3.3 Call-Type Monitoring Event Sequences

- Case A: D1, the calling device, joins call C3 with devices D2 and D3.

Table 17-88 Intrude Call—Call-Type Monitoring Event Sequences (Case A)

Monitored Call	Connection	Event	Event Cause
C1 (call attempting to connect to device D2)	D1C3	Conferenced	Override, Silent Participation, or Active Participation
C2 (resulting conference call)	D1C3	Conferenced	Override, Silent Participation, or Active Participation

- Case B: The called device places its existing active call on hold first and then connects to the new calling device (D1).

Table 17-89 Intrude Call—Call-Type Monitoring Event Sequences (Case B)

Monitored Call	Connection	Event	Event Cause
C2 (original call)	D2C2	Held	Intrude or Normal
C1 (call attempting to connect to device D2)	D2C1 (called)	Established	Intrude

1. The event cause is based on the value of the participationType parameter:
 - If the parameter was set to active, then the event cause will be Active Participation.
 - If the parameter was set to silent, then the event cause will be Silent Participation.
 - If the switching function cannot determine the type used, then the event cause will be Override.

17.1.16.3.4 Functional Requirements

1. To cancel an Intrude Call service, the computing function can either:
 - Issue the Clear Connection or Clear Call service with the ConnectionID of either the calling (cases A or B) or called (case B) device.
 - Have the calling device go on-hook.

If the above is successfully executed, normal call progress messages of Connection Cleared (with an event cause of Normal Clearing) will be generated.

2. The computing function should never assume the reuse of callIDs, although some switching functions may reuse one or the other.
3. If the called device has more than one call during the execution of this service, the switching function will choose which call to intrude upon.

17.1.17 Join Call

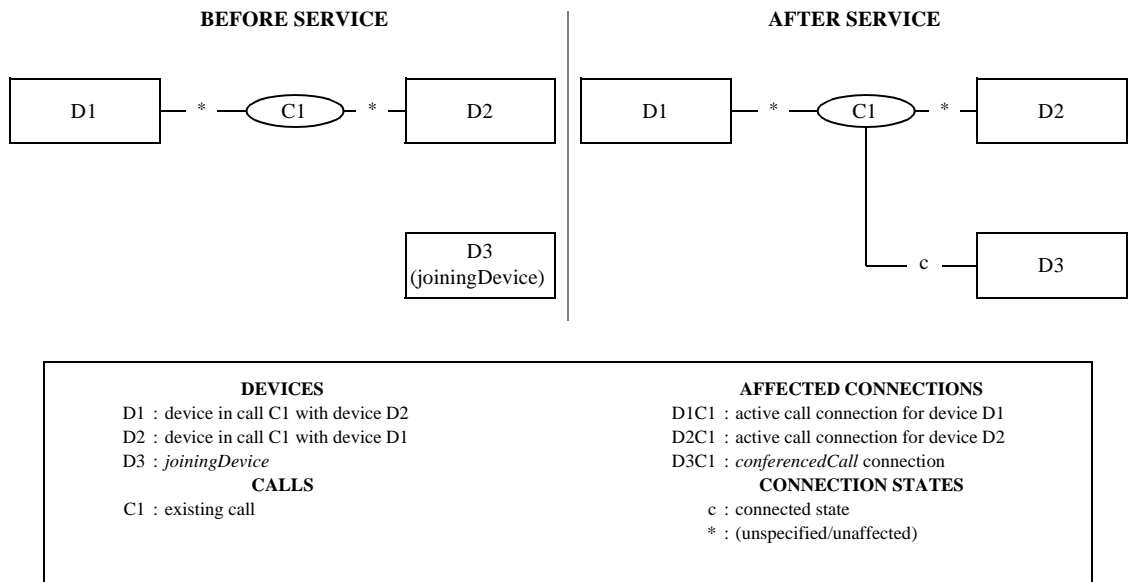
C → S

The Join Call service allows a computing function to request, on behalf of a device, that the device be joined into an existing call. In the process of establishing a connection with the joiningDevice, the joiningDevice may be prompted to go off-hook (if necessary) and when that device does so, it is added into the call.

This service is different from the Single Step Conference service in that the request is made on behalf of the joiningDevice (originating device).

This service is different from the Intrude Call service in that there is no prior failed call at the intruded-upon connection.

Figure 17-19 Join Call Service



17.1.17.1 Service Request

Table 17-90 Join Call—Service Request

Parameter Name	Type	M/O/C	Description
activeCall	connectionID	M	Specifies an existing connection in an active call to which the new device is to be added (or joined)
joiningDevice	DeviceID	M	Specifies the device that is to be added to (join) the existing call
autoOriginate	Enumerated	O	Specifies if the joining device is to be prompted or not (hands-free mode). The complete set of possible values is: <ul style="list-style-type: none"> Prompt (default) Do Not Prompt (auto originate)
participationType	Enumerated	O	Specifies the type of participation the joining device has in the resulting call. This is one of the following: <ul style="list-style-type: none"> active (default) - the joining device actively participates in the resulting conference call. As a result, the flow direction of the joiningDevice's connection (i.e., <i>conferencedCall</i>) will be Transmit and Receive. silent - the joining device can listen but cannot actively participate in the resulting conference call. As a result, the flow direction of the joiningDevice's connection (i.e., <i>conferencedCall</i>) will be Receive.

Table 17-90 Join Call—Service Request (continued)

Parameter Name	Type	M/O/C	Description
accountCode	AccountInfo	O	Specifies the account code to associate with the call.
authCode	AuthCode	O	Specifies the authorization code to allow the call.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.17.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.17.2.1 Positive Acknowledgement

Table 17-91 Join Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
conferencedCall	ConnectionID	M	Specifies the callID of the existing active call and the DeviceID of the joining device.
conferencedCallInfo	ConnectionInformation	O	Specifies the connection information associated with the conferencedCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.17.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.17.3 Operational Model

17.1.17.3.1 Connection State Transitions

Table 17-92 Join Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1	(Unspecified)	(Unaffected; no transition due to this service)
D2C1	(Unspecified)	(Unaffected; no transition due to this service)
D3C1 (conferencedCall)	Null	Connected

17.1.17.3.2 Device-Type Monitoring Event Sequences

Table 17-93 Join Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1	D3C1 (conferencedCall)	Conferenced	Join Call, ActiveParticipation, Silent Participation
	D3C1 (conferencedCall) (see item 2)	Service Initiated (optional) (see item 2)	Join Call (prompting)
	D3C1 (conferenced Call)	Established	Join Call, ActiveParticipation, Silent Participation
D2	D3C1 (conferencedCall)	Conferenced	Join Call, ActiveParticipation, Silent Participation
	D3C1 (conferencedCall) (see item 2)	Service Initiated (optional) (see item 2)	Join Call (prompting)
	D3C1 (conferencedCall)	Established	Join Call, ActiveParticipation, Silent Participation
D3 (joiningDevice)	D3C1 (conferencedCall)	Conferenced	Join Call, ActiveParticipation, Silent Participation
	D3C1 (conferencedCall) (see item 2)	Service Initiated (optional) (see item 2)	Join Call (prompting)
	D3C1 (conferencedCall)	Established	Join Call, ActiveParticipation, Silent Participation

17.1.17.3.3 Call-Type Monitoring Event Sequences

Table 17-94 Join Call—Call-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
C1	D3C1	Conferenced	Join Call, ActiveParticipation, Silent Participation
	D3C1 (see item 2)	Service Initiated (optional) (see item 2)	Join Call (prompting)
	D3C1	Established	Normal

1. For the Conferenced event, the event cause is based upon the value of the participationType parameter:
 - If it was set to active, then the event cause is Active Participation.
 - If it was set to silent, then the event cause is Silent Participation.
 - If the switching function cannot determine the type used, then the event cause is Join Call.
2. The Service Initiated event is dependent upon the prompting mode (as described in 6.7.10, “Prompting”, on page 46).
 - For the “prompting is a pre-condition of the service” mode, the Service Initiated is generated before any service specific events and is not part of the service completion criteria. The connectionID associated with the Service Initiated event is not associated with the Join Call service.
 - For the “prompting is part of the service” mode”, the Service Initiated event is part of the service completion criteria, and is generated after the Conferenced event and contains the same connectionID as the Conferenced and Established events.

17.1.17.3.4 Functional Requirements

1. The appearances in a shared bridged device configuration are unaffected by this service.
2. This service request does not support the use of a null device identifier or a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the “;” character) in it for the

joiningDevice parameter. A complete and valid device identifier shall be provided, otherwise the switching function will reject the service request with a negative acknowledgement.

3. Prompting of the joining device during the processing of the Join Call service is switching function specific (display flashing, ring pattern, lamp blinking, etc.).
4. If the autoOriginate parameter has a value of “Do Not Prompt”, then the device should be a speakerphone, hands-free telephone, or other device that can be automatically answered. If the device is not of this type, then the processing of this parameter is switching function specific. For example, the switching function may choose to accept this service and ignore this parameter, or it may reject the Join Call service with negative acknowledgement.
5. Call Forwarding and Do Not Disturb features for the joining device are not honoured. If a switching function supports prompting of the joining device and detects that a feature was activated while processing the Join Call service and this feature cannot be overridden, then the switching function will return a negative acknowledgement.
6. The call ID in the ConnectionIDs for the resulting conference call is inherited from the original active call.
7. When prompting a device which has a call appearance, bridged, or hybrid device configuration, only one of the appearances will be delivered the call.
8. If this service is used to join a device into a digital data call, the connection for the joiningDevice (i.e., conferencedCall) will inherit the characteristics of the call with the exception of connection flow direction and the number of channels used. In these cases, the connection flow direction is dependent on the value of participationType and the number of channels is based on what the switching function will use to allow the joiningDevice to effectively participate in the call.

17.1.18 Make Call

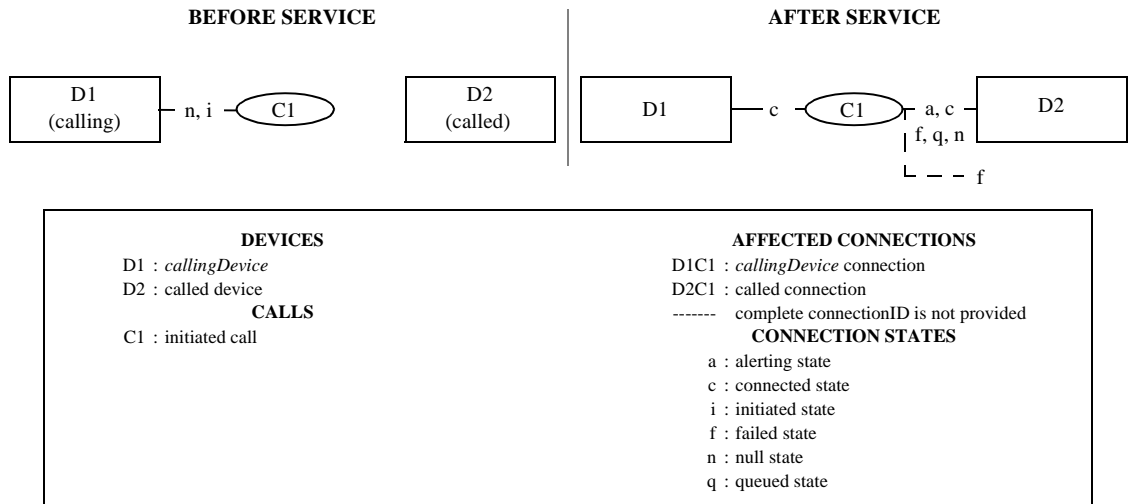
C → S

The Make Call service allows the computing function to set up a call between a calling device and a called device.

The service creates a new call and establishes an initiated or connected connection with the calling device. The Make Call service assigns a ConnectionID to the calling device and returns it in the positive acknowledgement.

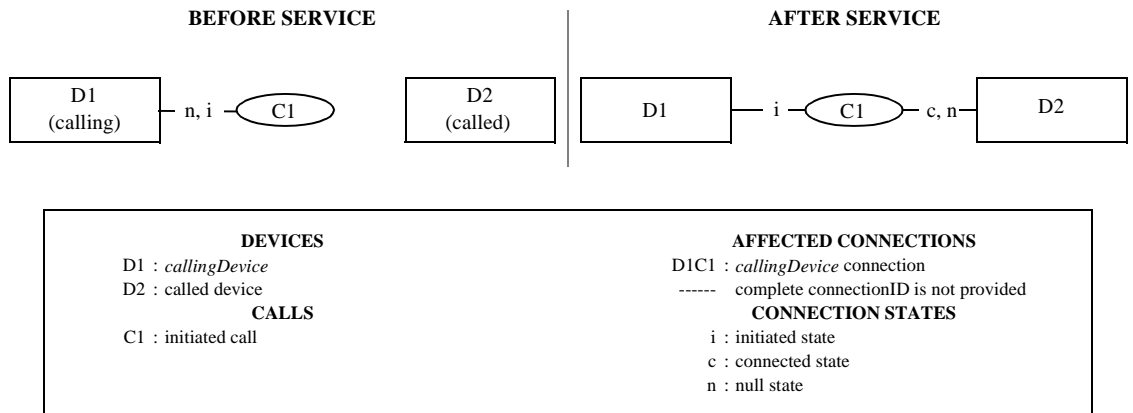
In the process of establishing the connection with the calling device, the calling device may be prompted to go off-hook (if necessary) and when that device does so, a call to the called device is originated or the calling device is still in the process of dialling the called device.

Figure 17-20 Make Call Service—Case A: Complete Dialling Sequence Provided



Note that (Case A) the Initiated state (D1C1) is used for Offhook Dialling (see Functional Requirement #6).

Figure 17-21 Make Call Service—Case B: Partial Dialling Sequence Provided



Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

Note that (Case B) the connected state (D2C1) exists when D2 is a NID.

17.1.18.1 Service Request

Table 17-95 Make Call—Service Request

Parameter Name	Type	M/O/C	Description
callingDevice	DeviceID	M	Specifies the calling/originating device. Note that this device may be a device that represents a group of devices. In this case the callingDevice in the service request is different from the actual calling device.
calledDirectoryNumber	DeviceID	M	Specifies the called device.
accountCode	AccountInfo	O	Specifies the account code to associate with the new call.
authCode	AuthCode	O	Specifies the authorization code to allow the call.
autoOriginate	Enumerated	O	Specifies if the calling device's connection is automatically answered (hands-free mode). The complete set of possible values is: <ul style="list-style-type: none"> • Prompt (default) • Do Not Prompt (auto originate)
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (Priority call, for example) to be associated with the call. See 12.2.5, "CallCharacteristics", on page 61 for the complete set of possible values. If the supported characteristics cannot be honoured, the switching function shall reject the service request.
mediaCallCharacteristics	MediaCallCharacteristics	O	This specifies the media characteristics to be associated with the call being made. If this parameter is not present then the media class is Voice.
callingConnectionInfo	ConnectionInformation	O	This specifies the connection information needed for the creation of a connection at the callingDevice. If this parameter is not present then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information.

17.1.18.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.18.2.1 Positive Acknowledgement

Table 17-96 Make Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
callingDevice	ConnectionID	M	Specifies the initial connection to the new call. The ConnectionID shall have the CallID of the resulting new call and the DeviceID of the calling device. Note that the calling device parameter in the service request may be different from the deviceID in the connectionID of callingDevice in the positive acknowledgement (when the callingDevice in the service request represents a group of stations, for example).
mediaCallCharacteristics	MediaCallCharacteristics	C	This specifies the adjusted media characteristics for the call being made. This parameter shall be provided if the call characteristics have been adjusted, otherwise it is optional.
initiatedCallInfo	ConnectionInformation	O	This specifies the adjusted connection information used during the creation of the initiatedCall for the callingDevice. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.18.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.18.3 Operational Model

17.1.18.3.1 Connection State Transitions

Table 17-97 Make Call—Connection State Transitions (Case A: Complete Dialling Sequence)

Connection	Initial State (Required)	Final State
D1C1 (callingDevice)	Null, Initiated (see item #17.1.18.3.1)	Initiated, Connected
D2C1 (called)	Null	Alerting, Connected, Queued, Fail, or Null (Null if call moves away from the originally called device [i.e., forwarded]).

Table 17-98 Make Call—Connection State Transitions (Case B: Partial Dialling Sequence)

Connection	Initial State (Required)	Final State
D1C1 (callingDevice)	Null, Initiated (see item #17.1.18.3.1)	Initiated
D2C1 (called)	Null	Connected (see item #1), Null

1. When providing a partial dialling sequence (Case B) the connected state for D2C1 only applies to external outbound calls. It indicates that enough of the dial string has been communicated for the switching function to connect the call to the Network Interface Device that will be associated with the called device in the external network.
2. The initial state of Initiated for D1C1 is used when D1 is “offhook” prior to the Make Call service request. The callID used for the Make Call shall be the same as the callID used for the initiated call.

17.1.18.3.2 Device-Type Monitoring Event Sequences

There are two types of Device-Type Monitoring Event sequences depending on the dialling sequence used for the called device.

- Case A: The dialling sequence for the called device is complete.

Table 17-99 Make Call—Device-Type Monitoring Event Sequences (Case A)

Monitored Device	Connection	Event	Event Cause
D1 (calling device)	D1C1	Service Initiated (optional)	Make Call, if calling device is prompted by the switching function.
			New Call, if the Make Call is done manually.
	D1C1	Originated	Normal
D2 (called device)	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (i.e., Forward, Do Not Disturb, etc.).(see item #1)	
	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (i.e., Forward, Do Not Disturb, etc.). (see item #1)	

- Case B: The dialling sequence for the called device is incomplete.

Table 17-100 Make Call—Device-Type Monitoring Event Sequences (Case B)

Monitored Device	Connection	Event	Event Cause
D1 (calling device)	D1C1	Service Initiated (optional)	Make Call, if calling device is prompted by the switching function.
			New Call, if the Make Call is done manually.
	D1C1	Digits Dialed	Normal
D2 (called device)	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (i.e., Forward, Do Not Disturb, etc.). (see item #1)	
	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (i.e., Forward, Do Not Disturb, etc.). (see item #1)	

17.1.18.3.3 Call-Type Monitoring Event Sequences

- Case A: The dialling sequence for the called device is complete.

Table 17-101 Make Call—Call-Type Monitoring Event Sequences (Case A)

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Service Initiated (optional)	Make Call, if calling device is prompted by the switching function.
			New Call, if the Make Call is done manually.
	D1C1	Originated	Normal
D2 (called device)	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (i.e., Forward, Do Not Disturb, etc.). (see item #1)	
	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (i.e., Forward, Do Not Disturb, etc.). (see item #1)	

- Case B: The dialling sequence for the called device is an incomplete sequence.

Table 17-102 Make Call—Call-Type Monitoring Event Sequences (Case B)

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Service Initiated (optional)	Make Call, if calling device is prompted by the switching function.
			New Call, if the Make Call is done manually.
	D1C1	Digits Dialed	Normal
	D2C1	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (i.e., Forward, Do Not Disturb, etc.)(see item #1)	

1. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

17.1.18.3.4 Functional Requirements

1. Prompting of the calling device during the processing of the Make Call service is switching function specific (display flashing, ring pattern, lamp blinking, etc.).
2. If the validation of the Make Call service fails for any reason, the computing function receives a negative acknowledgement, and no valid ConnectionIDs will have been created.
3. If the autoOriginate parameter is supported by the switching function and has a value of “Do Not Prompt”, and if the calling device is not capable of automatically answering the device, then the processing of this parameter is switching function dependent.
4. Call Forwarding and Do Not Disturb features for the calling device are not honoured for the call being made. If a switching function supports prompting of the calling device and detects that a feature was activated while processing the Make Call service and this feature cannot be overridden, then the switching function will return a negative acknowledgement.
5. All active features are honoured for the called device.
6. If a Make Call service is issued for a calling device that has a connection in the Initiated state (i.e., the computing function got a Service Initiated event for a phone manually going off-hook), and the switching function can support the issuing of the Make Call service under this condition, then the Connection ID in the positive acknowledgement of the Make Call service will contain the same Connection ID received on the Service Initiated event. If the switching function does not support the Make Call service under this condition, then the service will be rejected with a negative acknowledgement. (This note is an exception to requirement #2 in 9.5.1 on page 48.)
7. The calledDirectoryNumber parameter may contain a device identifier of null or contain a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the “;” character) at the end of it. When this parameter is used in this manner, the computing function is indicating that it wishes to stage the dialling sequence. The completion of the dialling sequence can be accomplished either by entering the rest of the sequence manually at the actual device or the computing function can use the Dial Digits service to complete the sequence. The other types of calledDirectoryNumber parameter shall contain a complete dialling sequence. The switching function may have a timeout period for multi-stage dialling. If the dialling sequence does not complete prior to this timeout, it may either abort the call or attempt to use the digits already dialed and signal that dialling is complete with an originated event.
8. If the Make Call service is used to initiate a multistage dialling sequence, the computing function is signalled to continue the dialling sequence via either the Service Initiated event (i.e., if the calledDirectoryNumber is Null) or the Digits Dialed event (i.e., if the calledDirectoryNumber has a partial dialling sequence character).

9. When prompting a device which has a call appearance, bridged, or hybrid device configuration, only one of the appearances will be delivered the call.
10. The switching function may or may not adjust the digital data characteristics (e.g., connection rate) and connection information (e.g., number of channels) that were supplied on the Make Call service request (Use the capabilities exchange services to determine which feature the switching function supports). If the switching function supports adjusting the characteristics/connection information but cannot adjust them in this case, the service request will be rejected with an appropriate error code, if the original characteristics/connection information cannot be provided. If the switching function can adjust of the characteristics/connection information, the positive acknowledgement will contain the adjusted value or values. If the computing function determines that the adjusted values are not adequate, it can terminate the digital data call (e.g., Clear Call).
11. If the computing function makes a digital data call and wants to also bind a particular Media Service to the call, then the computing function shall use the Media Attach Service service.
12. The callingDevice parameter in the Make Call service request may represent a group of devices from which the actual calling device is selected. In this case, the callingDevice parameter in subsequent events shall refer to the actual calling device. In addition, the Originated event contains a parameter (originatingDevice) that represents the group of devices (the callingDevice parameter passed in service request).

17.1.19 Make Predictive Call (to be defined)

C → S

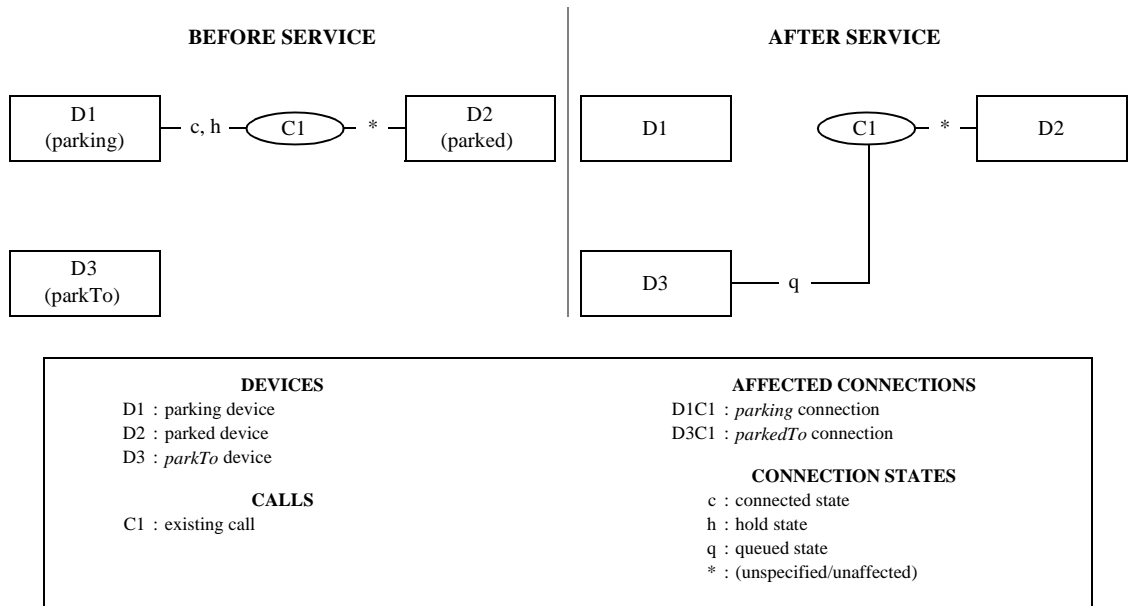
17.1.20 Park Call

C → S

The Park Call service moves a specified call at a device to a specified (parked-to) destination.

The device on whose behalf Park Call was invoked (the parking device) is no longer associated with the call (except when the parking device parks a call back to the parking device).

Figure 17-22 Park Call Service



Note that the parking device and the parkTo device may be the same (when a call is parked to the parking device).

Note that D1C1 can transit to the Queued state for certain Shared Bridged device configurations (see Functional Requirement #6).

17.1.20.1 Service Request

Table 17-103 Park Call—Service Request

Parameter Name	Type	M/O/C	Description
parking	ConnectionID	M	Specifies the connection to be parked.
parkTo	DeviceID	M	Specifies the device to which the call is to be parked (parked-to device). This may be either a station or a Park device.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.20.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.20.2.1 Positive Acknowledgement

Table 17-104 Park Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
parkedTo	ConnectionID	O	Specifies the connection at the parkedTo device.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.20.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.20.3 Operational Model

17.1.20.3.1 Connection State Transitions

Table 17-105 Park Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (parking)	Hold or Connected	Null, Queued (see item #1)
D2C1 (parked)	(Unspecified)	(Unaffected; no transition due to this service).
D3C1 (parkTo)	Null	Queued

1. Connection D1C1 may transit to the Queued state for certain shared bridged device configurations (see functional requirement #5).

17.1.20.3.2 Device-Type Monitoring Event Sequences

Table 17-106 Park Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (parking device)	D1C1	Diverted (see items #1, #2)	Park or Normal
D2 (parked device)	D1C1	Diverted (optional) (see items #1, #2, #4)	Park or Normal
	D3C1	Queued	Park or Normal
D3 (parkTo device)	D3C1	Queued	Park or Normal

17.1.20.3.3 Call-Type Monitoring Event Sequences

Table 17-107 Park Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Diverted (optional) (see items #1, #2, #5)	Park or Normal
	D3C1	Queued	Park or Normal

1. A Connection Cleared event will not be provided for the parking ConnectionID. The Diverted event implies the ConnectionID has gone to Null.
2. The Diverted event is not provided when a call is parked to the parking device (devices D1 and D3 are the same).
3. If the parking device stays offhook and receives busy or blocked tone, the switching function sends a Failed event (event cause of Blocked or Busy) for a call with a different callID (not C1), followed by a Connection Cleared event.

4. The switching function provides the Diverted event for D2 only if it is providing Diverted events for all devices in a call. This is indicated through the capabilities exchange services.
5. For call-type monitoring, the switching function provides the Diverted event for C1 only if it is providing Diverted events for call-type monitors. This is indicated through the capabilities exchange services.
6. If a Bridged event is generated for an Independent Shared Bridged device configuration (see Functional Requirement #5), it is not part of the service completion criteria.

17.1.20.3.4 Functional Requirements

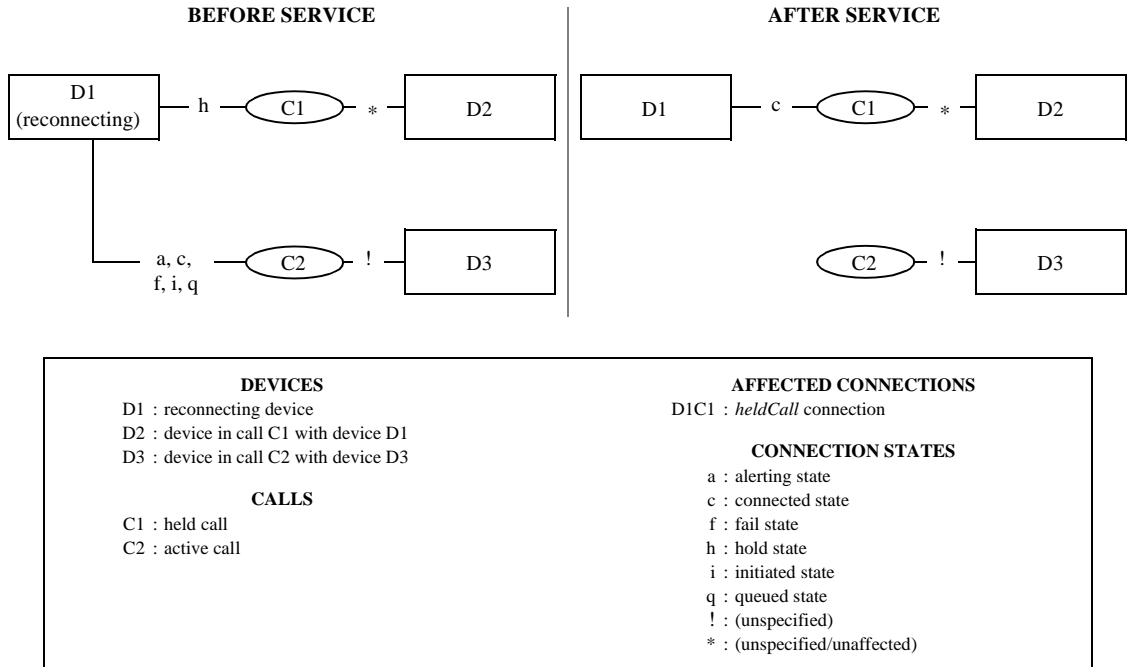
1. As a result of the Park Call service, the call ID associated with this call remains unchanged.
2. A call may be parked at a device which is already involved with the call. In this case the parking device (D1) and the parkTo (D3) are the same. Refer to the Operational Model description for specific requirements for this configuration.
3. The switching function may have a time-out period for the call once it is parked. If the call is not unparked before the time-out period ends, then the parked call may ringback the parking device or another switching defined device such as an attendant console, for example, or the call may be dropped.
4. Features such as Do Not Disturb and Forwarding are honoured at the parkedTo device when the call is being parked.
5. For Shared Bridged device configurations, when a call is parked from an appearance (parking) all appearances will be cleared from the call (i.e., Connection Cleared events with a cause of Normal Clearing), except when the call that is being parked is part of an Independent Shared Bridged device configuration and the appearance that is parking the call is not the last appearance connected into the call. In this case the appearance parking the call will return to the inactive mode (i.e., Bridged event with a cause of Normal) and the other appearances are unaffected. For more information, refer to Annex B.2.3, "Shared-Bridged".
6. This service request does *not* support the use of a null device identifier or a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the ";" character) in it for the parkTo parameter. A complete and valid device identifier shall be provided, otherwise the switching function will reject the service request with a negative acknowledgement.

17.1.21 Reconnect Call

C → S

The Reconnect Call service will clear a specified connection at the reconnecting device and retrieve a specified held connection at the same device.

Figure 17-23 Reconnect Call Service



17.1.21.1 Service Request

Table 17-108 Reconnect Call—Service Request

Parameter Name	Type	M/O/C	Description
activeCall	ConnectionID	M	Specifies the connection to be cleared.
heldCall	ConnectionID	M	Specifies the held connection.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.21.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.21.2.1 Positive Acknowledgement

Table 17-109 Reconnect Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.21.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.21.3 Operational Model

17.1.21.3.1 Connection State Transitions

Table 17-110 Reconnect Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (heldCall)	Hold	Connected
D1C2 (activeCall)	Alerting, Connected, Initiated, Fail, or Queued	Null
D2C1	(Unspecified)	(Unaffected)
D3C2	(Unspecified)	(Unspecified)

17.1.21.3.2 Device-Type Monitoring Event Sequences

Table 17-111 Reconnect Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (reconnecting device)	D1C2	Connection Cleared	Normal Clearing, Call Canceled
	D1C1	Retrieved	Normal
D2 (and other devices in call C1)	D1C1	Retrieved	Normal
D3 (and other device in call C2)	D1C2	Connection Cleared	Normal Clearing, Call Canceled

17.1.21.3.3 Call-Type Monitoring Event Sequences

Table 17-112 Reconnect Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C2	D1C2	Connection Cleared	Normal Clearing, Call Canceled
C1	D1C1	Retrieved	Normal Clearing

17.1.21.3.4 Functional Requirements

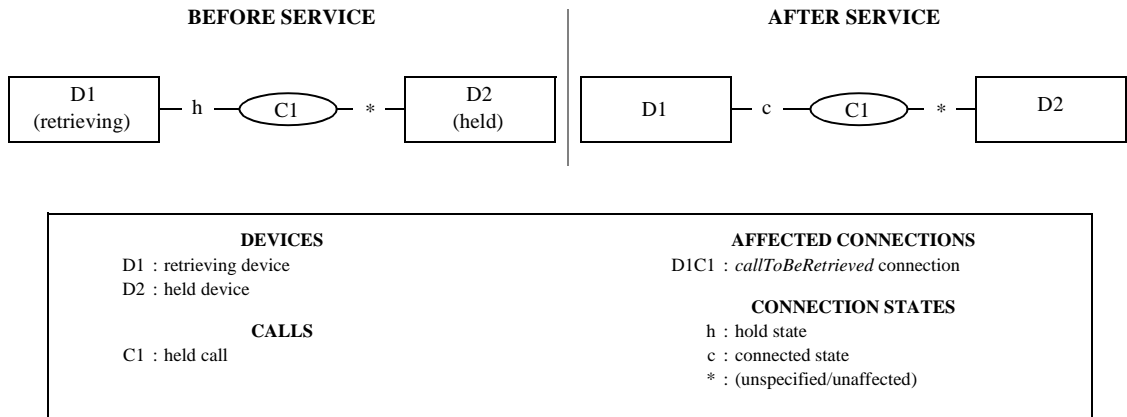
1. This service is a compound service that is equivalent to a computing function issuing a Clear Connection service for the activeCall ConnectionID and then issuing a Retrieve Call service for the heldCall ConnectionID.
2. If all appearances of a shared bridged device configuration are in the hold state and the heldCall parameter contains an appearance's connection ID in the call, then the other appearances in the device configuration will return to the inactive mode (queued state, Bridged events).
3. When the last connected appearance, in a shared bridged device configuration, is cleared as a result of the Reconnect service, all other device configuration appearance associated with the activeCall connection are also cleared from the call (i.e., Connection Cleared events).
4. If the Hold Call, Consultation Call, or the Alternate Call service was used to place the call on hold and the connectionReservation parameter was used to reserve the media stream channel(s) for the held call (e.g., ISDN bearer channel), then the same media stream channel(s) will be used for the call when it becomes reconnected.

17.1.22 Retrieve Call

C → S

The Retrieve Call service connects a specified held connection.

Figure 17-24 Retrieve Call Service



17.1.22.1 Service Request

Table 17-113 Retrieve Call—Service Request

Parameter Name	Type	M/O/C	Description
callToBeRetrieved	ConnectionID	M	Specifies the held connection to be retrieved.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.22.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.22.2.1 Positive Acknowledgement

Table 17-114 Retrieve Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.22.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.22.3 Operational Model

17.1.22.3.1 Connection State Transitions

Table 17-115 Retrieve Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (callToBeRetrieved)	Hold	Connected
D2C1 (and any other connections in call C1)	(Unspecified)	(Unaffected; no transition due to this service).

17.1.22.3.2 Device-Type Monitoring Event Sequences

Table 17-116 Retrieve Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (retrieving device)	D1C1	Retrieved	Normal
D2 (and any other devices in call C1)	D1C1	Retrieved	Normal

17.1.22.3.3 Call-Type Monitoring Event Sequences

Table 17-117 Retrieve Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Retrieved	Normal

17.1.22.3.4 Functional Requirements

1. The Retrieve Call service may only be requested for a connection at a device's physical element if there is no other connected connection already present at the device's physical element. A device's physical elements can only interact with one voice call at a time so if an attempt is made to retrieve a call while another voice call is connected, the switching function sends a negative acknowledgement to the Retrieve Call service request.
2. If the Hold Call, Consultation Call, or the Alternate Call service was used to place the call on hold and the connectionReservation parameter was used to reserve the media stream channel(s) for the held call (e.g., ISDN bearer channel), then the same media stream channel(s) will be used for the call when it becomes reconnected.
3. There may or may not be a media stream channel(s) available for the call on hold. In the case where a media stream channel(s) is not available, the switching function is unable to comply with the service request. In this case the switching function will send either a negative acknowledgement or the Service Completion Failure event, depending upon the acknowledgement model.
4. If all appearances of a shared-bridged or exclusive-bridged device configuration are in the hold state and the callToBeRetrieved parameter contains an appearance's connection ID in the call, then the other appearances in the device configuration will return to the inactive mode (queued state, Bridged events) in the case of shared-bridging or the blocked mode (fail state, failed events) in the case of exclusive bridging.

17.1.23 Single Step Conference Call

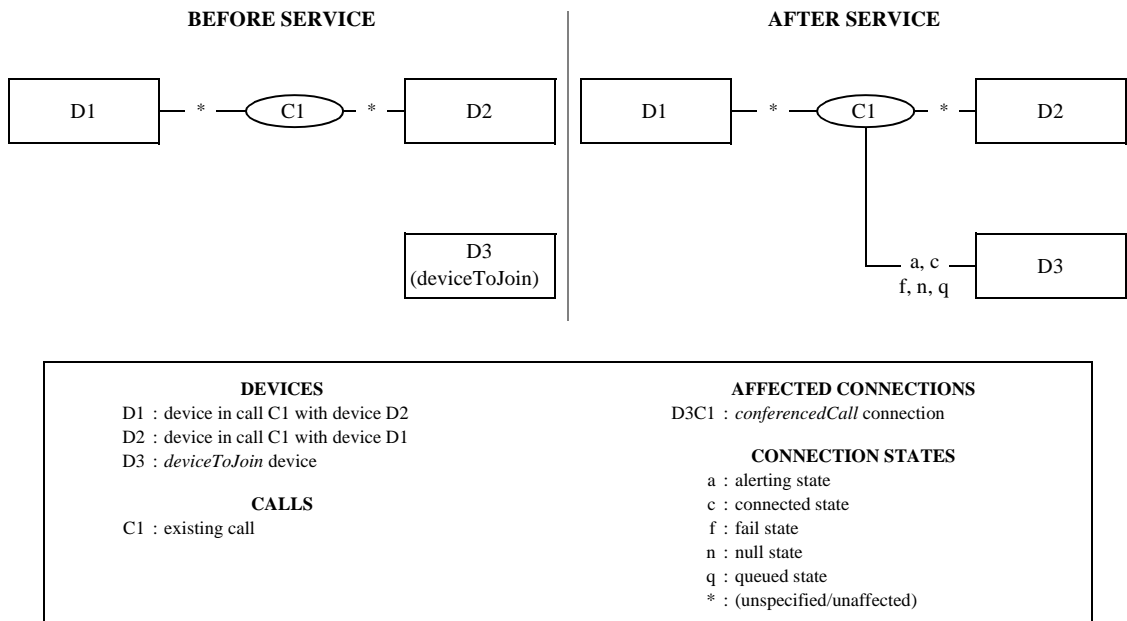
C → S

The Single Step Conference Call joins a new device into an existing call.

This service can be repeated to make n-device conference calls (subject to switching function limits).

This service is distinguished from the Join Call service by the way the device being added to the call perceives the direction of the resulting connection (e.g. alerts). In the case of the Join Call service, the device generates an outgoing connection (e.g. prompts). This affects the parameters associated with the service and the events flowing as a result of the service.

Figure 17-25 Single Step Conference Call Service



Note that the activeCall connection in the service request may be either D1C1 or D2C1.

17.1.23.1 Service Request

Table 17-118 Single Step Conference Call—Service Request

Parameter Name	Type	M/O/C	Description
activeCall	ConnectionID	M	Specifies an existing connection in the call to which a new device is to be added.
deviceToJoin	DeviceID	M	Specifies the device that is to be added to the call.
participationType	Enumerated	O	Specifies the type of participation the added device has in the resulting call. The complete set of possible values is: <ul style="list-style-type: none"> Active (default) - The added device actively participates in the resulting conference call. As a result, the flow direction of the deviceToJoin's connection (i.e., conferencedCall) will be Transmit and Receive. Silent - The added device can listen but cannot actively participate in the resulting conference call. As a result, the flow direction of the deviceToJoin's connection (i.e., conferencedCall) will be Receive.
accountCode	AccountInfo	O	Specifies the account code to associate with the call.
authCode	AuthCode	O	Specifies the authorization code to allow the call.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.

Table 17-118 Single Step Conference Call—Service Request (continued)

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.23.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.23.2.1 Positive Acknowledgement

Table 17-119 Single Step Conference Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
conferencedCall	ConnectionID	M	Specifies the callID of the existing active call and the DeviceID of the joining device.
conferencedCallInfo	ConnectionInformation	O	Specifies the connection information associated with the conferencedCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.23.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.23.3 Operational Model

17.1.23.3.1 Connection State Transitions

Table 17-120 Single Step Conference Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1	(Unspecified)	(Unaffected; no transition due to this service)
D2C1	(Unspecified)	(Unaffected; no transition due to this service)
D3C1 (conferencedCall)	Null	Alerting, Connected, Queued, Fail, or Null (Null if call moves away from D3 [i.e., forwarded]).

17.1.23.3.2 Device-Type Monitoring Event Sequences

Table 17-121 Single Step Conference Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1	D3C1 (conferencedCall)	<i>Note:</i> It is switching function dependent if other events flow before the Conferenced event. (See item #2)	
	D3C1 (conferencedCall)	Conferenced	Silent or Active Participation, or Single Step Conference
	D3C1 (conferencedCall)	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (See item #3)	
D2	D3C1 (conferencedCall)	<i>Note:</i> It is switching function dependent if other events flow before the Conferenced event. See Functional Requirement #2.	
	D3C1 (conferencedCall)	Conferenced	Silent or Active Participation, or Single Step Conference
	D3C1 (conferencedCall)	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (See Item #3)	
D3 (deviceToJoin)	D3C1 (conferencedCall)	<i>Note:</i> It is switching function dependent if other events flow before the Conferenced event. See Functional Requirement #2.	
	D3C1 (conferencedCall)	Conferenced	Silent or Active Participation, or Single Step Conference
	D3C1 (conferencedCall)	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (See Item #3)	

17.1.23.3.3 Call-Type Monitoring Event Sequences

Table 17-122 Single Step Conference Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D3C1	<i>Note:</i> It is switching function dependent if other events flow before the conferenced event. See Functional Requirement #2.	
	D3C1	Conferenced	Silent or Active Participation, or Single Step Conference
	D3C1 (conferencedCall)	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see Item #3)	

1. The event cause is based on the value of the participationType parameter.
 - If it was set to active, then the event cause will be Active Participation.
 - If it was set to silent, then the event cause will be Silent Participation.
 - If the switching function cannot determine the type used, then the event cause will be Single Step Conference.
2. If the deviceToJoin is not in the connected state after the Single Step Conference Call service is executed, then normal call progress events will flow regarding the state of this device after the Conferenced event. Devices in the resulting call will hear the call progress associated with the deviceToJoin.
3. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).

17.1.23.3.4 Functional Requirements

1. The call ID in the ConnectionIDs for the resulting conference call is the same as that in the original active call.
2. It is switching function dependent to determine at which point in the call progress a device is actually conferenced into a call. For example, a device may become part of an existing call before it is actually conferenced into a call. In this case, additional events may flow before the Conferenced event. The callID in the ConnectionIDs for these events shall be the same as that in the original active call and shall be the same as in the Conferenced event that follows.
3. The appearances in a shared bridged device configuration are unaffected by this service.
4. This service request does not support the use of a null device identifier or a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the “;” character) in it for the deviceToJoin parameter. A complete and valid device identifier shall be provided, otherwise the switching function will reject the service request with a negative acknowledgement.
5. For the new destination device (deviceToJoin), all active features for the device are honoured.
6. If this service is used to add a device into a digital data call, the connection for the deviceToJoin (i.e., conferencedCall) will inherit the characteristics of the call with the exception of connection flow direction and the number of channels used. In these cases, the connection flow direction is dependent on the value of participationType and the number of channels is based on what the switching function will use to allow the deviceToJoin to effectively participate in the call.

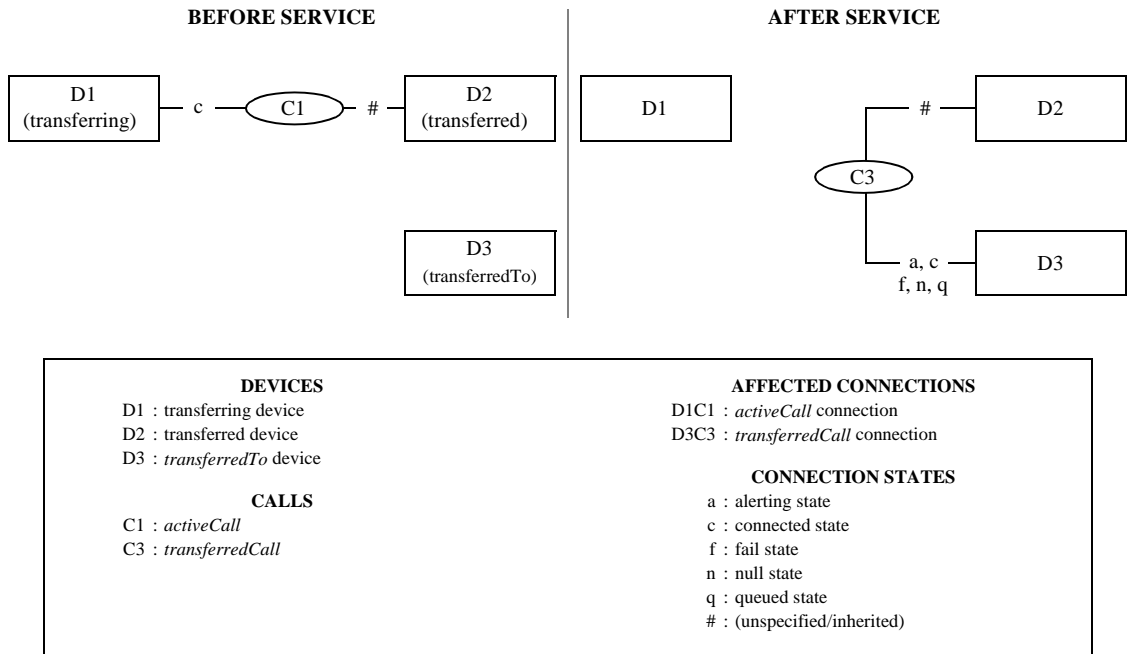
17.1.24 Single Step Transfer Call

C → S

The Single Step Transfer Call service transfers an existing connected connection at a device to another device.

This transfer is performed in a single-step, that is the device doing the transfer does not have to place the existing call on hold before issuing the Single Step Transfer Call service.

Figure 17-26 Single Step Transfer Call Service



17.1.24.1 Service Request

Table 17-123 Single Step Transfer Call—Service Request

Parameter Name	Type	M/O/C	Description
activeCall	ConnectionID	M	Specifies the connected connection in the call to be replaced.
transferredTo	DeviceID	M	Specifies the new called (transferred-to) device.
accountCode	AccountInfo	O	Specifies the account code to associate with the call.
authCode	AuthCode	O	Specifies the authorization code to allow the call.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.24.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.24.2.1 Positive Acknowledgement

Table 17-124 Single Step Transfer Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
transferredCall	ConnectionID	M	Specifies the ConnectionID of the deviceToTransferTo in the transferred call.
connections	ConnectionList	O	Specifies information on each device/connection that is remaining in the call as a result of the service.
transferredCallInfo	ConnectionInformation	O	Specifies the connection information associated with the transferredCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.24.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.24.3 Operational Model

17.1.24.3.1 Connection State Transitions

Table 17-125 Single Step Transfer Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (activeCall)	Connected	Null
D2C1 and any other Connections in the call C1	(Unspecified)	Null
D2C3	Null	(Inherited from the corresponding connection in call C1)
D3C3 (transferredCall)	Null	Alerting, Connected, Queued, Fail, or Null (Null if call moves away from the originally transferred-to device [i.e., forwarded])

17.1.24.3.2 Device-Type Monitoring Event Sequences

Table 17-126 Single Step Transfer Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (transferring)	D1C1	Transferred (see item #3)	Single Step Transfer
D2 (transferred device) (and any other devices in the resulting call)	D1C1	Transferred	Single Step Transfer
	D3C3 (transferredCall)	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #4)	
D3 (transferredTo)	D3C3 (transferredCall)	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #4)	

17.1.24.3.3 Call-Type Monitoring Event Sequences

Table 17-127 Single Step Transfer Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1	D1C1	Transferred (see item #3)	Single Step Transfer
C3	D3C3 (transferredCall)	Events depend on the type of device (ACD, extension, external device, etc.) and features activated at it at the time of this service (Forward, Do Not Disturb, etc.). (see item #4)	

1. If the transferredTo device is not in the connected state after the Single Step Transfer Call service is executed, then normal call progress events will flow regarding the state of this device after the Transferred event.
2. A Connection Cleared event will not be seen for the activeCall ConnectionID; the Transferred event implies that the connection at the transferringDevice has been cleared.
3. If the transferring device stays off-hook and receives busy or blocked tone, the switching function sends a Failed event (event cause of Blocked or Busy) for a call with a different callID (not C1 or C3), followed by a Connection Cleared event.
4. Providing of these events by the switching function does not affect the service completion conditions (i.e., a service is considered complete without these events being provided).
5. If a Bridged event is generated for an Independent Shared Bridged device configuration (see Functional Requirement #5), it is not part of the service completion criteria.

17.1.24.3.4 Functional Requirements

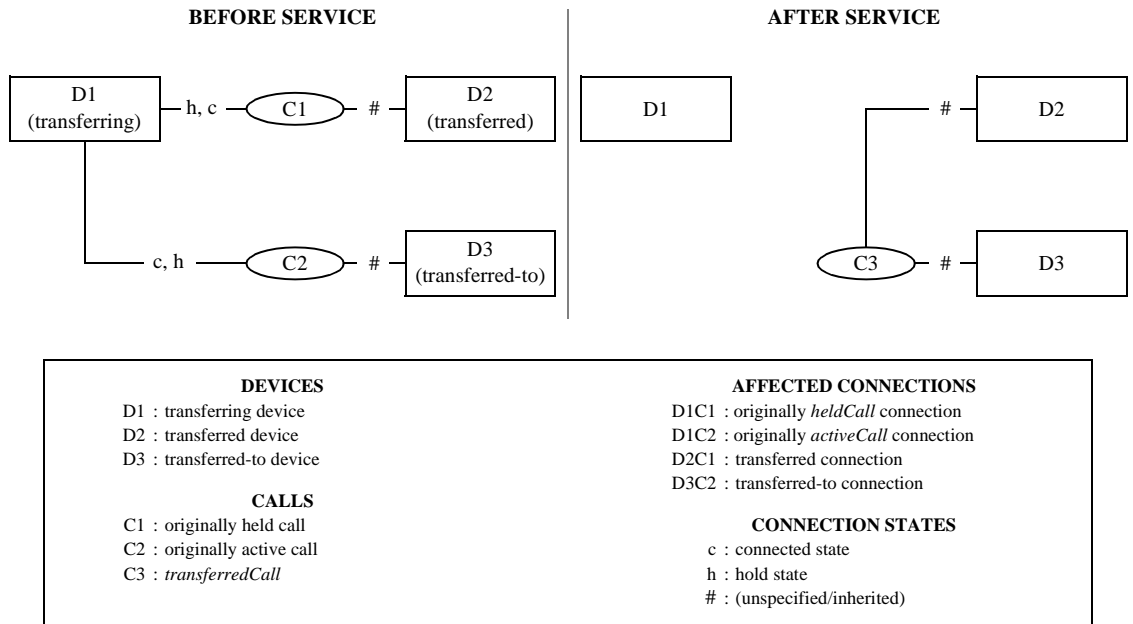
1. The computing function should never assume the reuse of callIDs, although some switching functions may reuse the activeCall callID for the transferredCall callID
2. As a result of this service, the device or devices being transferred may hear call-progress signals for the new device. For example, if the new device answers, the call resumes or if the new device is busy or does not answer, the other devices in the call may receive resulting call progress tones. In either case, the transferring device has left the call and has no further relationship with it.
3. If the Single Step Transfer Call service fails for any reason after the positive acknowledgement is returned, the switching function may return the call to the transferring device. Refer to 6.7.3, “Recall”, on page 39.
4. For the new destination device (deviceToTransferTo), all active features for the device are honoured.
5. Shared Bridged device configurations, when a call is transferred by an appearance (transferring device) all appearances will be cleared from the call (i.e., Connection Cleared events with a cause of Normal Clearing), except when the call that is being transferred is part of an Independent Shared Bridged device configuration and the appearance that is transferring the call is not the last appearance connected into the call. In this case the appearance transferring the call will return to the inactive mode (i.e., Bridged event with a cause of Normal) and the other appearances are unaffected. For more information, refer to Annex B.2.3, “Shared-Bridged”.
6. This service request does not support the use of a null device identifier or a Diallable Digits format (DD) device identifier that has a partial dialling sequence character (i.e., the “;” character) in it for the transferredTo parameter. A complete and valid device identifier shall be provided, otherwise the switching function will reject the service request with a negative acknowledgement.
7. If this service is used to single step transfer a digital data call, the connection for the transferredTo device (i.e., transferredCall) will inherit the characteristics of the call.

17.1.25 Transfer Call

C → S

The Transfer Call service transfers a call held at a device to an active call at the same device. The held and active calls at the transferring device shall be merged into a new call. Also, the Connections of the held and active calls at the transferring device shall become Null and their ConnectionIDs shall be released (i.e., the transferring device is no longer involved with the call).

Figure 17-27 Transfer Call Service



Note that both the D1C1 and D1C2 connections may be held or connected prior to the transfer.

17.1.25.1 Service Request

Table 17-128 Transfer Call—Service Request

Parameter Name	Type	M/O/C	Description
heldCall	ConnectionID	M	Specifies the held connection.
activeCall	ConnectionID	M	Specifies the active connection.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.25.2 Service Response

The capability exchange services describe the type of acknowledgement model (atomic or multi-step) that the switching function supports for this service request.

17.1.25.2.1 Positive Acknowledgement

Table 17-129 Transfer Call—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
transferredCall	ConnectionID	M	Specifies the ConnectionID of the transferred-to device in the resulting call.
connections	ConnectionList	O	Specifies information on each device/connection that is remaining in the call as a result of the service.

Table 17-129 Transfer Call—Positive Acknowledgement (continued)

Parameter Name	Type	M/O/C	Description
transferredCallInfo	ConnectionInformation	O	Specifies the connection information associated with the transferredCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.1.25.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

17.1.25.3 Operational Model

17.1.25.3.1 Connection State Transitions

Table 17-130 Transfer Call—Connection State Transitions

Connection	Initial State (Required)	Final State
D1C1 (heldCall)	Connected, Hold	Null
D1C2 (activeCall)	Connected, Hold	Null
D2C1 and any other connections in call C1	(unspecified)	Null
D3C2 and any other connections in call C2	(unspecified)	Null
D3C3 and any other connections in call C3	Null	(Inherited from the corresponding connection in calls C1 and C2)

17.1.25.3.2 Device-Type Monitoring Event Sequences

Table 17-131 Transfer Call—Device-Type Monitoring Event Sequences

Monitored Device	Connection	Event	Event Cause
D1 (transferring)	(See item #4)	Transferred (see item #3)	Transfer or Normal
D2 (transferred)	(See item #4)	Transferred	Transfer or Normal
D3 (transferred-to)	(See item #4)	Transferred	Transfer or Normal

17.1.25.3.3 Call-Type Monitoring Event Sequences

Table 17-132 Transfer Call—Call-Type Monitoring Event Sequences

Monitored Call	Connection	Event	Event Cause
C1 or C2	(See item #4)	Transferred (see items #1 and #2)	Transfer

1. If the transferred-to device was not in the connected state prior to the Transfer Call service, then normal call progress events will flow after the Transferred event. Devices in the original held call will hear the call progress associated with the connection to the transferred-to device.
2. A Connection Cleared Event will not be provided for the activeCall or heldCall ConnectionIDs. The Transferred event implies connection cleared for the transferring device.
3. If the transferring device stays off-hook and receives busy or blocked tone, the switching function may send a Failed event (event cause of Blocked or Busy) for a different call, followed by a Connection Cleared event.
4. There are multiple connections affected by this service.

5. If a Bridged event is generated for an Independent Shared Bridged device configuration (see Functional Requirement #4), it is not part of the service completion criteria.

17.1.25.3.4 Functional Requirements

1. To get the connections for each of the devices to their initial states, the computing function can either:
 - Use the Consultation Call service to place a call on hold and place a new call.
 - Use the Alternate Call service to place a call on hold and answer an alerting call.

In either of these cases, if the switching function supports the consultOptions parameter in these services, the computing function shall provide this parameter with a value of either “Conference Only” or “Unrestricted”.

Some switching function support a third approach for preparing for the Conference Call service involving the use of a hold service followed by a Make Call.

The fourth approach supported by some switching functions involves two held calls at the same device.

Certain switching functions also support a fifth approach involving two active calls at the same device.

The computing function should use the capabilities exchange services to determine which of these approaches is supported by the switching function.

2. If the computing function uses the Consultation Call service and specifies the consultOptions parameter with a value of Conference, and then attempts to complete the consultation call with a Transfer Call service, it will be rejected with a negative acknowledgement.
3. If the call is not established at the transfer-to device, the switching function may return the call to the transferring device. Refer to 6.7.3, “Recall”, on page 39.
4. For Shared Bridged device configurations, when a call is transferred by an appearance (transferring device) all appearances will be cleared from the call (i.e., Connection Cleared events with a cause of Normal Clearing), except when the call that is being transferred is part of an Independent Shared Bridged device configuration and the appearance that is transferring the call is not the last appearance connected into the call. In this case the appearance transferring the call will return to the inactive mode (i.e., Bridged event with a cause of Normal) and the other appearances are unaffected. For more information, refer to Annex B.2.3, “Shared-Bridged”.
5. The computing function should never assume the reuse of callIDs, although some switching functions may reuse one or the other.

17.2 Events

Table 17-133 Call Control Events Summary

Call Control Event	Description	Pg.
17.2.1 Bridged	Indicates that an appearance at a shared bridged device configuration has been placed into an inactive mode (i.e., queued state).	193
17.2.2 Call Cleared	Indicates that all devices have been removed from an existing call.	195
17.2.3 Conferenced	Indicates that the conferencing device has conferenced itself or another device with an existing call.	198
17.2.4 Connection Cleared	Indicates that a device in a call has disconnected or dropped out from a call.	203
17.2.5 Delivered	Indicates that a call is being presented to a device in either the Ringing or Entering Distribution modes of the alerting state.	207
17.2.6 Digits Dialed	Indicates that a call or feature is being attempted from a device and that a portion of the dialling sequence has been completed.	211
17.2.7 Diverted	Indicates that a call has been diverted from a device.	214
17.2.8 Established	Indicates that a device has answered or has been connected to a call.	218
17.2.9 Failed	Indicates that a call cannot be completed and/or a connection has entered the Fail state.	222
17.2.10 Held	Indicates that an existing call has been put on hold.	227
17.2.11 Network Capabilities Changed	Indicates that a situation occurred during a call's progress in a public or private network that modifies its signalling capability (i.e., inter-networking).	229
17.2.12 Network Reached	Indicates that a call has been connected to an external network using a Network Interface Device (e.g., trunk, CO Line).	232
17.2.13 Offered	Indicates that a call is in a pre-delivery state at a device (prior to ringing indication or delivering ringback, for example).	236
17.2.14 Originated	Indicates that a call is being attempted from a device.	240
17.2.15 Queued	Indicates that a call has been queued.	243
17.2.16 Retrieved	Indicates that a previously held call has been retrieved.	247
17.2.17 Service Initiated	Indicates that a device has gone off-hook for service or is being prompted to go off-hook.	249
17.2.18 Transferred	Indicates that an existing call has been transferred to another device and that the device transferring the call has been dropped from the call.	252

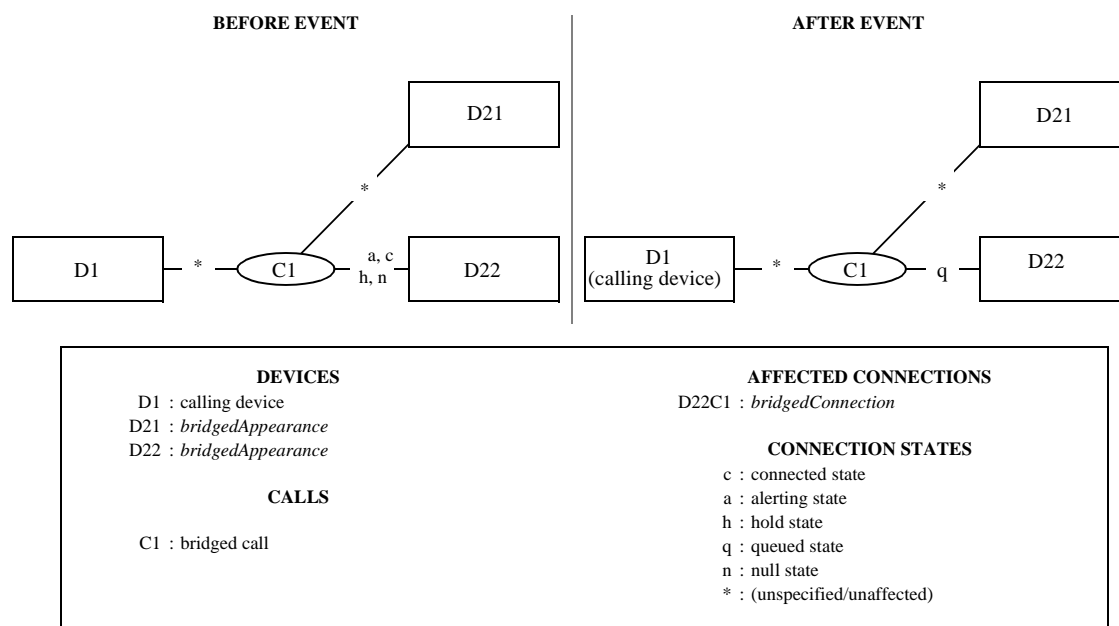
17.2.1 Bridged

The Bridged event indicates that an appearance at a shared bridged device configuration has been placed into an inactive mode (i.e., queued state). See the device configuration section for more details on shared bridged device configurations,

Common situations that generate this event include:

- When the first appearance in a shared bridged device configuration appearance connects into the call at the device (i.e., Answer Call) (manual and service request initiated), the other appearances enter the inactive mode.
- When the first appearance in a shared bridged device configuration leaves a call at the device and at least one appearance is still connected into the call. (i.e., Clear Connection) (manual and service request initiated).
- When the first appearance in a shared bridged device configuration retrieves a call and the other appearances return to the inactive mode.

Figure 17-28 Bridged Event



17.2.1.1 Event Parameters

Table 17-134 Bridged—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
bridgedConnection	ConnectionID	M	Specifies the connection of the appearance that was placed in the inactive mode.
bridgedAppearance	SubjectDeviceID	M	Specifies the appearance which was placed in the inactive mode.
localConnectionInfo	LocalConnectionState	C	<p>Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call.</p> <ul style="list-style-type: none"> • For the bridged appearance at the bridged device: Queued • For the other appearances at the bridged device & all other devices left in the call: (unaffected) <p>This parameter is mandatory for events generated for device-type monitors and shall otherwise not be provided.</p>

Table 17-134 Bridged—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
correlatorData	CorrelatorData	O	Specifies the correlator data associated with the call.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
bridgedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the bridgedConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies the non-standardized information attached to the event.

17.2.1.2 Event Causes

Table 17-135 Bridged—Event Causes

Event Cause	Description	Associated Features
Normal	An appearance has been placed into an inactive mode.	Multi-Appearance

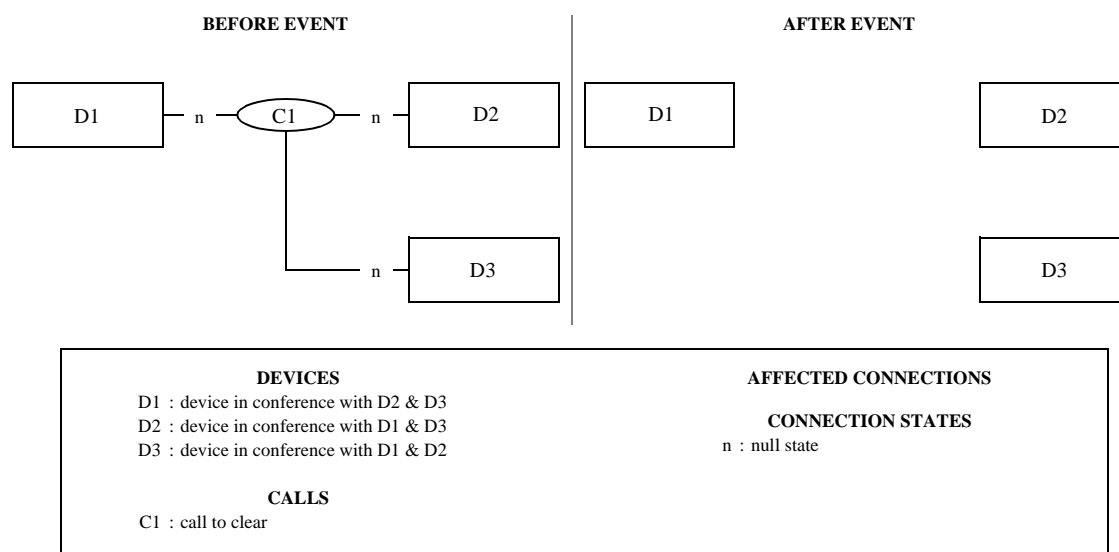
17.2.2 Call Cleared

The Call Cleared event is only provided for calls that are being call-type monitored. This event indicates that a call has been cleared and no longer exists within the switching sub-domain. A call is cleared when there is no longer any device associated with the call.

Common situations that generate this event include:

- After the last remaining device disconnects from the call.
- All devices in a call are immediately disconnected from a call such as when a conference controller dissolves a call.
- The computing function issues a Clear Call service request that is successful.

Figure 17-29 Call Cleared Event



17.2.2.1 Event Parameters

Table 17-136 Call Cleared—Event Parameters

Parameter Name	Type	M/ O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
clearedCall	ConnectionID	M	Specifies the ConnectionID of the cleared call. Note that the DeviceID shall be omitted in this ConnectionID.
correlatorData	CorrelatorData	O	Specifies the correlator data associated with the call.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event. The cause supplied will be the same as the cause supplied on the last Connection Cleared event for the call.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, "MediaCallCharacteristics", on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, "CallCharacteristics", on page 61 for the complete set of possible values.

Table 17-136 Call Cleared—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.2.2 Event Causes

Table 17-137 Call Cleared—Event Causes

Event Cause	Description	Associated Features
Busy	The call was cleared because it reached a busy destination.	Connection Failure
Call Back	The call was cleared after a call back or call back message feature was invoked.	Call Back Call-Related, Call Back Message Call-Related
Call Cancelled	The call was cleared without a device going on-hook (via the Clear Call service, for example).	Clear Connection, Clear Call, Connection Failure
Call Not Answered	The call was cleared because it was not answered before a timer elapsed.	Clear Connection, Clear Call, Connection Failure
Destination Out of Order	The call was cleared because it encountered a destination out of service.	Connection Failure
Do Not Disturb	The call was cleared because the call encountered a destination that has the Do Not Disturb feature set.	Do Not Disturb
Incompatible Destination	The call was cleared because it encountered an incompatible destination.	Connection Failure
Invalid Account Code	The call was cleared because of an invalid account code.	Connection Failure
Invalid Number Format	The call was cleared because of an incorrect dialled number.	Connection Failure
Maintenance	The call was cleared because it encountered a facility or endpoint in a maintenance condition.	Connection Failure
Network Congestion	The call was cleared because it reached a congested network	External Call
Network Not Obtainable	The call was cleared because it could not reach the destination network.	External Call
Network Out of Order	The call was cleared because it encountered a network that is out of order.	Connection Failure
Network Signal	The call was cleared that involved a device that is outside of the switching sub-domain.	Clear Connection, Clear Call, Connection Failure, External Call
Normal Clearing	The call was cleared (a more specific event cause cannot be provided).	Clear Connection, Clear Call, Any feature
Not Available Bearer Service	The call was cleared because it was requested with bearer capability that is currently not available.	Connection Failure
Not Supported Bearer Service	The call was cleared because it was requested with a bearer capability that is currently not supported.	Connection Failure
Number Changed	The call was cleared because the called number has been changed to a new number and the call cannot be completed.	Connection Failure
Number Unallocated	The call was cleared because the called number is not allocated to a subscriber.	Connection Failure
Override	The call was cleared because of an Override (e.g. Intrude) feature.	Connection Failure

Table 17-137 Call Cleared—Event Causes (continued)

Event Cause	Description	Associated Features
Reorder Tone	The call was cleared because the call encountered a reorder condition.	Connection Failure
Resource not Available	The call was cleared because resources were not available.	Connection Failure
Selected Trunk Busy	The call was cleared because the specific selected Network Interface Device (e.g., trunk, CO Line) is busy.	Connection Failure
Trunks Busy	The call was cleared because there was no available Network Interface Device (e.g. trunk, CO Line).	Connection Failure
Unauthorized Bearer Service	The call was cleared because it was requested with an unauthorized bearer capability.	Connection Failure

17.2.2.3 Functional Requirements

1. Connection Cleared events shall be sent for all devices in the call before the Call Cleared event is sent. The Call Cleared event can not be used as a substitute for the appropriate Connection Cleared events.
2. The Call Cleared event is only sent to call-type monitors. It is never sent to device-type monitors.

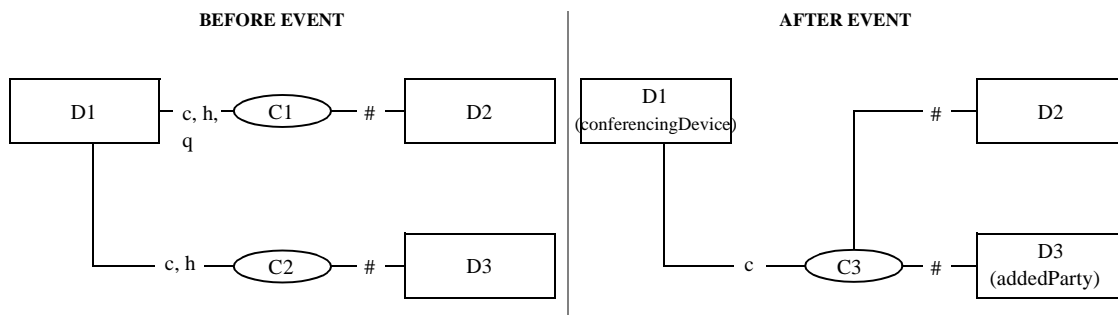
17.2.3 Conferenced

The Conferenced event indicates that the conferencing device has conferenced itself or another device with an existing call and that no devices have been removed from the resulting call.

Common situations that generate this event include:

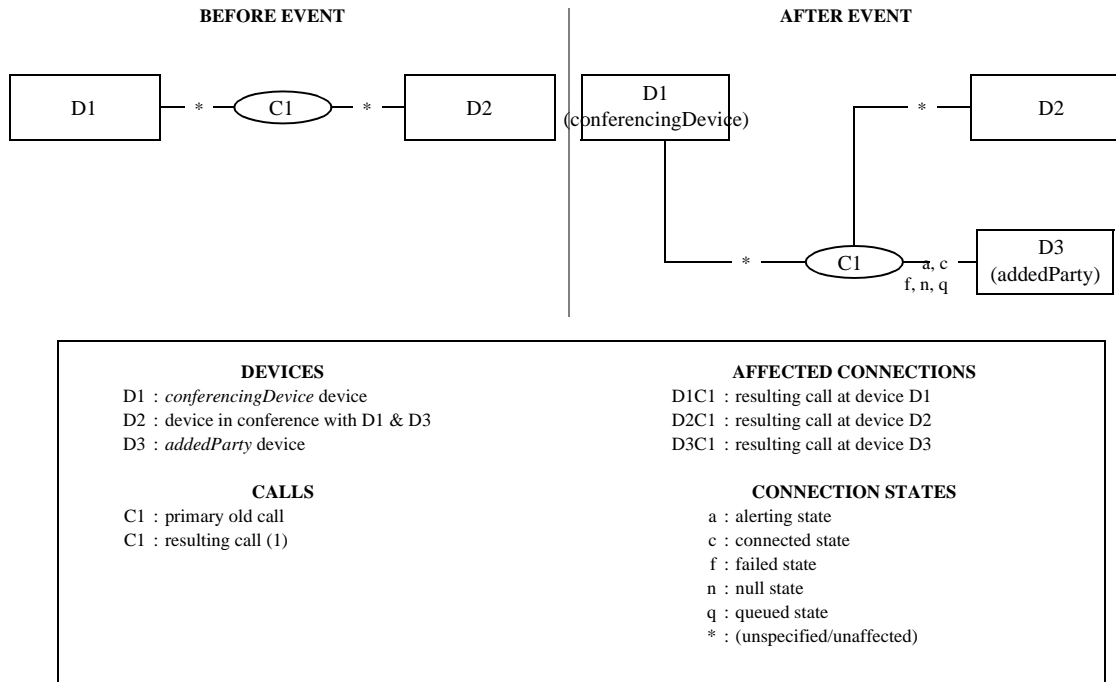
- Two step conferencing situations (manual and service initiated)
- Single step conferencing situations (manual and service initiated)
- Intrude situations (manual and service initiated)
- Join situations (manual and service initiated)

Figure 17-30 Conferenced Event—Case A: Two Step Conferencing



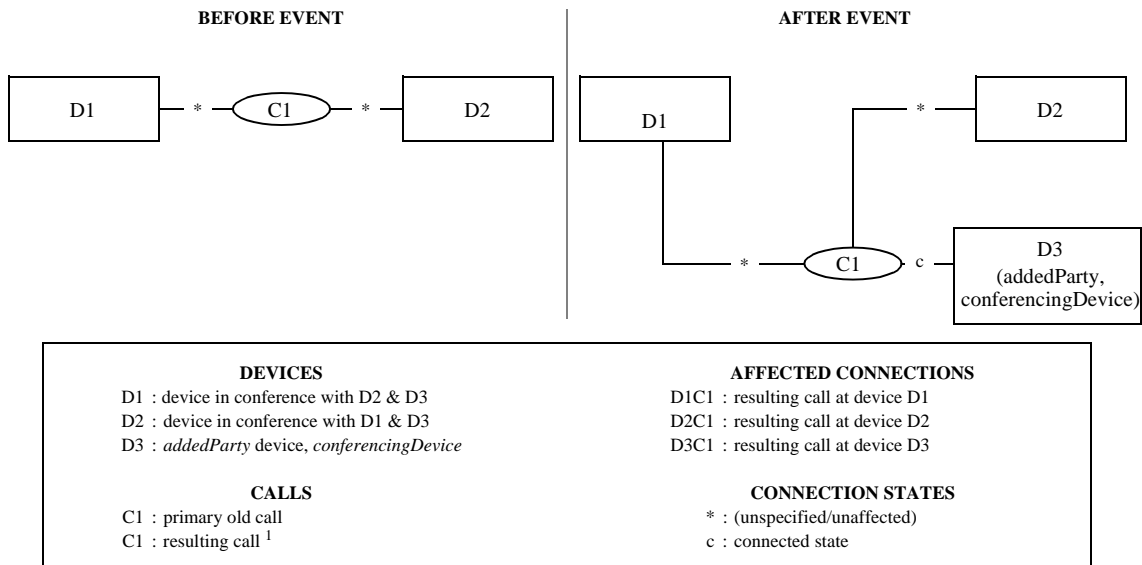
DEVICES	AFFECTED CONNECTIONS
D1 : <i>conferencingDevice</i> device	D1C1 : <i>primaryOldCall</i> connection
D2 : device in conference with D1 and D3	D1C2 : <i>secondaryOldCall</i> connection
D3 : <i>addedParty</i> device	D2C1 : initial call at device D2
	D3C2 : initial call at device D3
	D1C3 : resulting call at device D1
	D2C3 : resulting call at device D2
	D3C3 : resulting call at device D3
CALLS*	CONNECTION STATES
C1 : primary old call	c : connected state
C2 : secondary old call	h : hold state
C3 : resulting call	q : queued state
	# : (unspecified/inherited)
* the primary/secondary old call and the primary/secondary old call connections mentioned in this figure are from the perspective of the conferencingDevice (D1). See Functional Requirement #1.	

Figure 17-31 Conferenced Event—Case B: Single Step Conferencing



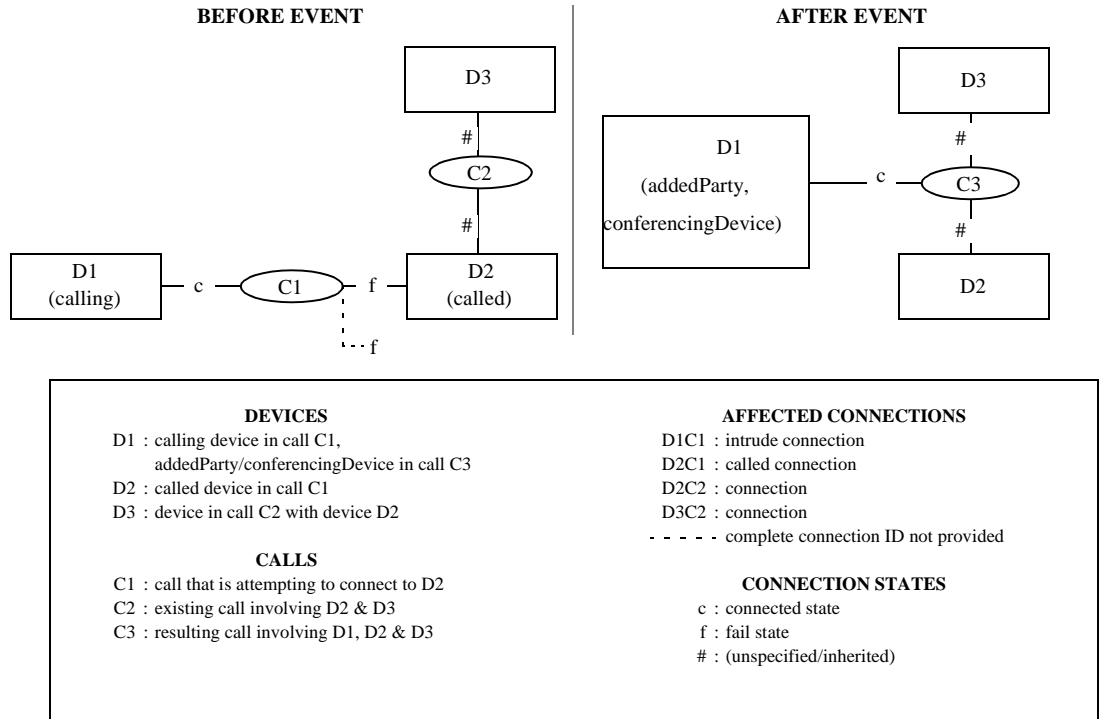
1. Since the Call ID does not change for a Single Step Conference Call, the primary old call and the resulting call are the same.

Figure 17-32 Conferenced Event—Case C: Join Call



1. Since the Call ID does not change for the Join Call service, the primary old call and the resulting call are the same.

Figure 17-33 Conferenced Event - Case D: Intrude Call



Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

Refer to 17.1.16, “Intrude Call”, on page 161 for information on how a connection can transit to the connected state.

17.2.3.1 Event Parameters

Table 17-138 Conferenced—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
primaryOldCall	ConnectionID	M	Specifies the connection of the primary call. See Functional Requirement #1.
secondaryOldCall	ConnectionID	C	Specifies the connection of the secondary call. If the switching function supports the “fixed-view” option (as indicated by the capability exchange services), this parameter is mandatory. If the switching function supports the “local-view” option, this parameter is mandatory if there are two known calls involved with the conference (before the conference is created) from the perspective of the monitored device, otherwise it shall not be provided. See Functional Requirement #1.
conferencingDevice	SubjectDeviceID ¹	M	Specifies the device ID of the conferencing device.
addedParty	SubjectDeviceID ¹	M	Specifies the device ID of the last device added to the call.
conferenceConnections	ConnectionList	M	Specifies information on each device/ConnectionID in the resulting conference call. See Functional Requirement #2.

Table 17-138 Conferenced—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices that are in the conference call: <ul style="list-style-type: none"> • Conferencing device - Connected (or Unaffected for Case B). • Other devices - (See above figures for specific cases of the Conferenced event) This parameter is mandatory for events generated for device-type monitors and otherwise shall not be provided.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

1. Note that SubjectDeviceID refers to a parameter type—not the subject device of the Conferenced event. This parameter type is used to represent the two devices in this event because the two devices are affected by the generation of this event (i.e., the conferencing device and the addedParty device). However, there is only one device which is the subject of the event and that is the conferencing device. For more details on the SubjectDeviceID parameter type, see 12.3.24, “Subject-DeviceID”, on page 79.

17.2.3.2 Event Causes

Table 17-139 Conferenced—Event Causes

Event Cause	Description	Associated Features
Active Participation	The call was conferenced and the added device can actively participate in the call. This feature is typically used to allow intrusion by a supervisor with the ability to speak and listen into an agent call.	Single Step Conference Call, Intrude Call, Join Call
Conference	The call was conferenced because of a two step conference.	Conference Call, Conference
Join Call	The call was conferenced because of a Join Call.	Join Call

Table 17-139 Conferenced—Event Causes (continued)

Event Cause	Description	Associated Features
Network Signal	The call was involved in a conference located outside of the switching sub-domain.	External Call
Normal	The call was conferenced (a more specific event cause cannot be provided).	Conference
Override	The call was conferenced because of an override (e.g., Intrude Call) feature. This event cause may be used when the participation type (active, silent) cannot be provided.	Intrude Call
Silent Participation	<p>The call was conferenced and the added device can silently participate in the call.</p> <p>This feature allows a third party, such as an ACD agent supervisor, to join the call. The joining party can hear the entire conversation, but cannot be heard by either originating party. The feature, sometimes called Silent Intrusion, may provide a tone to one or both parties to indicate that the joining party is listening to the call.</p>	Single Step Conference Call, Intrude Call, Join Call
Single Step Conference Call	The call was conferenced because of a single step conference.	Single Step Conference Call, Intrude Call

17.2.3.3 Functional Requirements

1. The contents of the `primaryOldCall` and the `secondaryOldCall` parameters may be either a “fixed view” or a “local view” of the connections at a device before the conference has been completed. The switching function indicates which view it provides via the `connectionView` parameter in the capability exchange services.
 - **fixed view** - for each conferenced event generated by monitors placed on different devices in a call, the switching function provides the same information in the `primaryOldCall` and the `secondaryOldCall` parameters independent of the `monitorType` (call or device-type monitor) and independent of the role of the device in the conference (`conferencingDevice`, `addedParty`, etc.). The meaning of these parameters for the fixed-view are:
 - `primaryOldCall` - specifies the first call visible at the `conferencingDevice`.
 - `secondaryOldCall` - specifies the second call visible at the `conferencingDevice`.
 - **local view** - for each conferenced event generated by monitors placed on different devices in a call, the switching function provides different information in the `primaryOldCall` and the `secondaryOldCall` parameters that depends upon which call was made visible first, from the perspective of the monitored device. The meaning of these parameters for the local-view are:
 - `primaryOldCall` - specifies the first call visible at the monitored device. For example, for a device-type monitor placed on the `conferencingDevice` (two step conference), this is the first call placed on hold (C1). For a device-type monitor placed on the `addedParty` device, this is the first (and only) call involved in the conference (C2) from the perspective of the monitored device.
 - `secondaryOldCall` - specifies the second call visible at the monitored device. For example, for a device-type monitor placed on the `conferencingDevice` (two step conference), this is the consultation call (C2). For a monitor placed on the `addedParty` device, there is no `secondaryOldCall` parameter.
2. The `conferenceConnections` parameter is a list that contains the new `ConnectionID` and may contain the old `ConnectionID`, the `DeviceID` (values such as ANI, etc.), and for externally located devices the associated `Network Interface DeviceID`.
3. The computing function should never assume the reuse of `callIDs`, although some switching functions may reuse one or the other.
4. There may be other devices present in the resulting conference call which are not part of the conference.

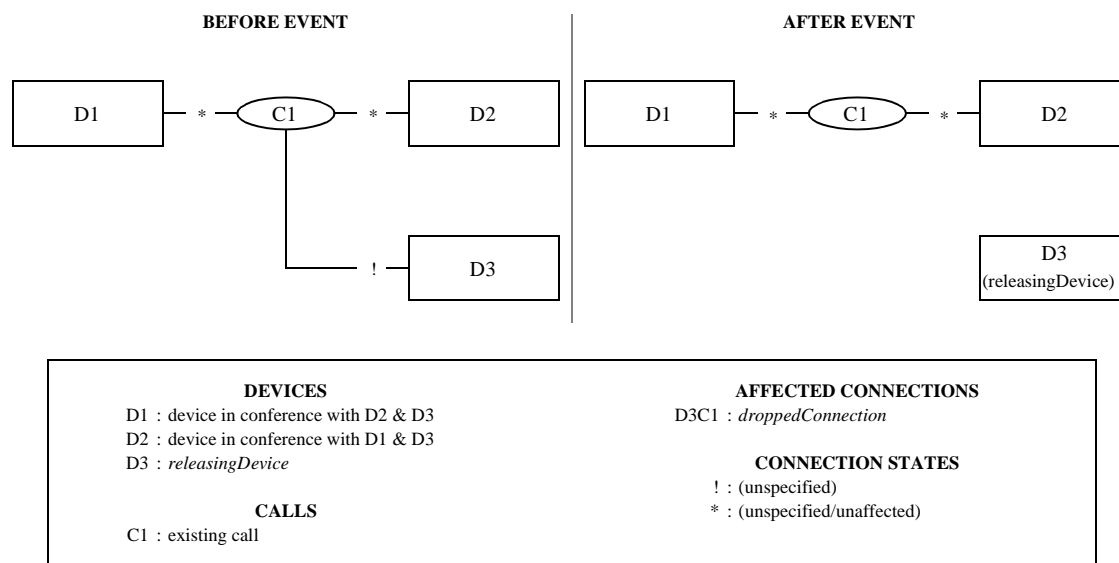
17.2.4 Connection Cleared

The Connection Cleared event indicates that a single device has disconnected or dropped out of a call.

Common situations that generate this event include:

- A user manually terminates the call (by going on-hook, for example).
- The Clear Connection service is successfully invoked.
- Connection clears as a result of some other service's operation.

Figure 17-34 Connection Cleared Event



17.2.4.1 Event Parameters

Table 17-140 Connection Cleared—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
droppedConnection	ConnectionID	M	Specifies the connection of the device that was dropped from the call.
releasingDevice	SubjectDeviceID	M	Specifies the device that dropped from the call.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the clearing device: Null • For the other devices left in the call: (unaffected) This parameter is mandatory for events generated for device-type monitors and otherwise shall not be provided.
correlatorData	CorrelatorData	O	Specifies the correlator data associated with the call.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
chargingInfo	ChargingInfo	O	Specifies a total value of charging or currency units for the device that dropped from the call.
cause	EventCause	M	Specifies the reason for the event.

Table 17-140 Connection Cleared—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
droppedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the droppedConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.4.2 Event Causes

Table 17-141 Connection Cleared—Event Causes

Event Cause	Description	Associated Features
Alert Time Expired	The connection was cleared because a timer associated with the Make Predictive call expired.	Make Predictive Call
Busy	The connection was cleared because the call reached a busy destination.	Connection Failure
Call Back	The connection was cleared after a Call Back or a Call Back Message feature has been invoked.	Call Back Call-Related, Call Back Message Call-Related
Call Cancelled	The connection was cleared without a device going on-hook (via the Clear Call service, for example).	Clear Connection, Clear Call, Connection Failure
Call Not Answered	The connection was cleared because the call was not answered before a timer elapsed.	Clear Connection, Clear Call, Connection Failure
Destination Detected	The connection was cleared because the call encountered a specific destination.	Make Predictive Call
Destination Not Obtainable	The connection was cleared because the call could not reach the destination.	Connection Failure
Destination Out of Order	The connection was cleared because the call encountered a destination that is out of service.	Connection Failure
Do Not Disturb	The connection was cleared because the call encountered a destination that has the Do Not Disturb feature set.	Do Not Disturb
Incompatible Destination	The connection was cleared because the call encountered an incompatible destination.	Connection Failure
Invalid Account Code	The connection was cleared because of an invalid account code.	Connection Failure
Invalid Number Format	The connection was cleared because of an incorrect dialled number.	Connection Failure

Table 17-141 Connection Cleared—Event Causes (continued)

Event Cause	Description	Associated Features
Maintenance	The connection was cleared because it encountered a facility or endpoint in a maintenance condition.	Connection Failure
Network Congestion	The connection was cleared because the call reached a congested network.	External Call
Network Not Obtainable	The connection was cleared because the call could not reach the destination network.	External Call
Network Out of Order	The connection was cleared because the call encountered a network that is out of order.	Connection Failure
Network Signal	The device located outside the switching sub-domain has dropped from the call.	Clear Connection, Clear Call, Connection Failure, External Call
Normal Clearing	The connection was cleared (a more specific event cause cannot be provided).	Clear Connection, Clear Call, Any feature
Not Available Bearer Service	The connection was cleared because the call was requested with a bearer capability that is currently not available.	Connection Failure
Not Supported Bearer Service	The connection was cleared because the call was requested with a bearer capability that is currently not supported.	Connection Failure
Number Changed	The connection was cleared because the called number has been changed to a new number and the call cannot be completed.	Connection Failure
Number Unallocated	The connection was cleared because the called number is not allocated to a subscriber.	Connection Failure
Override	The connection was cleared because of an override (e.g., Intrude Call) feature.	Connection Failure
Queue Cleared	A call is queued in multiple ACD queues and the connection associated with one of the ACD queues was cleared because the call was distributed to an available agent from another ACD queue.	ACD
Reorder Tone	The connection was cleared because the call encountered a reorder condition. When this occurs, the network usually provides Reorder Tone to indicate that a request (call, feature, or supplementary service) was not recognizable. This condition typically results when a user dials a number that is not valid or attempts to obtain a service that is not enabled for that user or device.	Connection Failure
Resource Not Available	The connection was cleared because of resources not available.	Connection Failure
Selected Trunk Busy	The connection was cleared because the specific selected Network Interface Device (e.g., trunk, CO Line) is busy.	Connection Failure
Trunks Busy	The connection was cleared because there was no available Network Interface Device (e.g., trunk, CO Line).	Connection Failure
Unauthorized Bearer Service	The connection was cleared because the call was requested with an unauthorized bearer capability.	Connection Failure

17.2.4.3 Functional Requirements

1. A Connection Cleared event will be generated for appearances of a shared bridged device configuration under the following conditions.

- The call is ended.
- The appearance permanently drops from the call through some feature/service.
- The call moves away from the device configuration and none of the appearances are connected into the call.
- The call moves away from the device configuration and the device configuration is an appearance interdependent shared bridged device configuration.

The event cause will be Normal Clearing, the droppedConnection will be the connection identifier associated with the appearance and the releasingDevice will depend on the type of referencing model that is supported by the switching function (refer to 6.1.7, “Referencing Devices, Elements, Appearances and Device Configurations”, on page 28).

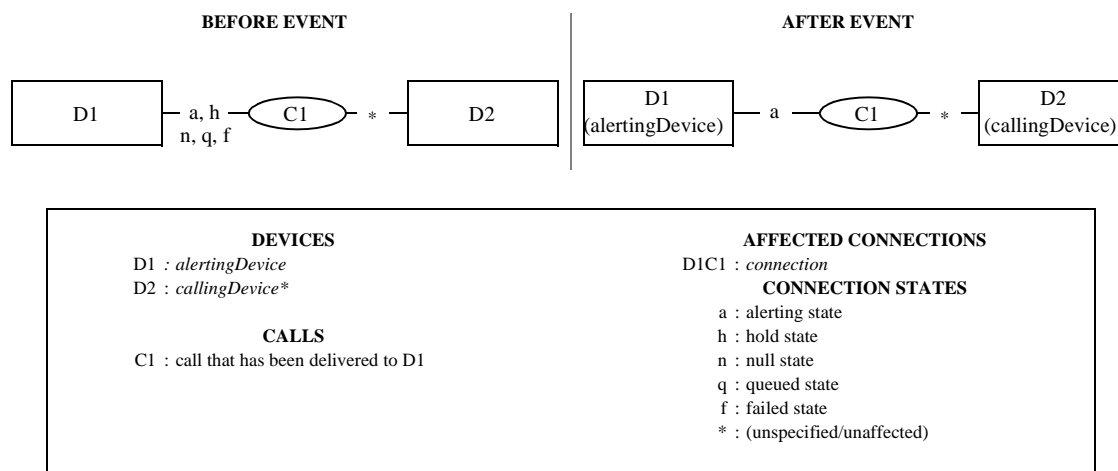
17.2.5 Delivered

The Delivered event indicates that a call is being presented to a device in either the Ringing or Entering Distribution modes of the alerting state.

Common situations that generate this event include:

- A call has been assigned to a device and that device is alerting.
- A call has been assigned to a distribution device such as an ACD, routing device, or hunt group.

Figure 17-35 Delivered Event



*There are some situations where D1 can be the calling device (e.g. Make Predictive Call).

17.2.5.1 Event Parameters

Table 17-142 Delivered—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
connection	ConnectionID	M	Specifies the connection that is alerting.
alertingDevice	SubjectDeviceID	M	Specifies the device that is alerting.
callingDevice	CallingDeviceID	M	Specifies the calling device.
calledDevice	CalledDeviceID	M	Specifies the originally called device.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected from device.
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from. If this parameter is supported, it shall be provided if there is an originating NID associated with the call, otherwise it shall not be provided. See Functional Requirement #4.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the alerting device: Alerting • For the calling device: (unaffected) This parameter is mandatory for events generated for device-type monitors and otherwise shall not be provided.

Table 17-142 Delivered—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
connectionInfo	ConnectionInformation	O	Specifies the connection information associated with the alerting connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.5.2 Event Causes

Table 17-143 Delivered—Event Causes

Event Cause	Description	Associated Features
ACD Busy	The call was delivered to an ACD device that has currently no available agents.	ACD, Consultation Call

Table 17-143 Delivered—Event Causes (continued)

Event Cause	Description	Associated Features
ACD Forward	The call was delivered to a new device and is no longer queued at the previous ACD device.	ACD
ACD Saturated	The call was delivered to an ACD device that has currently no internal resources available (including agents).	ACD, Consultation Call
Call Back	The call was delivered to a device because of a previously set call back feature.	Call Back Call-Related
Call Forward	The call was delivered to a device after it was forwarded (any type of forwarding).	Call Forwarding
Call Forward—Busy	The call was delivered to a device after it was forwarded because of a busy condition.	Call Forwarding
Call Forward—Immediate	The call was delivered to a device after it was forwarded (forwarding on all conditions).	Call Forwarding
Call Forward—No Answer	The call was delivered to a device after it was forwarded because of a no answer condition.	Call Forwarding
Distributed	The call was delivered to a device because the call has moved from the distribution mechanism (ACD, Hunt) to an available device associated with the distribution mechanism.	ACD, Routeing Services
Entering Distribution	The call was delivered to a distribution mechanism (ACD, Hunt) for the purpose of being distributed.	ACD, Routeing Services
Key Operation	The call was delivered to a device that has an appearance in a call appearance, bridged, or hybrid device configuration.	Multiple Appearance
Multiple Alerting	The call that is alerting the subject device is also alerting other devices.	Multiple Alerting
Network Signal	The call was delivered to a device that is outside of the switching sub-domain.	External Calls
New Call	The call was not redirected.	Any feature
No Agent Available	The call was delivered to a device because there were no available devices in a group (ACD).	ACD, Forwarding
Normal	The call was delivered to a device (a more specific cause cannot be provided).	Any feature
Overflow	The call was delivered to a device after it overflowed a queue, group, or target.	ACD
Override	The call was delivered to a device as a result of an override (e.g., Intrude Call) feature.	Intrude Call
Recall	The call was delivered to a device as part of the recall feature (e.g., due to a time-out associated with a feature that failed to complete).	Recall
Redirected	The call was delivered to a device after it was diverted from or deflected to this device.	Deflect Call, Divert Call
Remains in Queue	The call was delivered to a device while the calling device's position in the queue is retained. For example, the call could be delivered to a VRU or other automated answering equipment while the calling device retains its position in the ACD queue.	ACD, Queuing, Single Step Conference, Join Call
Resources not available	The call was delivered because some resources were not available (ACD, forwarding).	ACD, Forwarding

Table 17-143 Delivered—Event Causes (continued)

Event Cause	Description	Associated Features
Single Step Transfer	The call alerted the device due to a Single Step Transfer service.	Single Step Transfer
Single Step Conference	The call alerted the device due to a Single Step Conference service.	Single Step Conference
Timeout	The event was generated because a trunk timer expired. It is not generated as the result of a particular network event or condition.	External calls

17.2.5.3 Functional Requirements

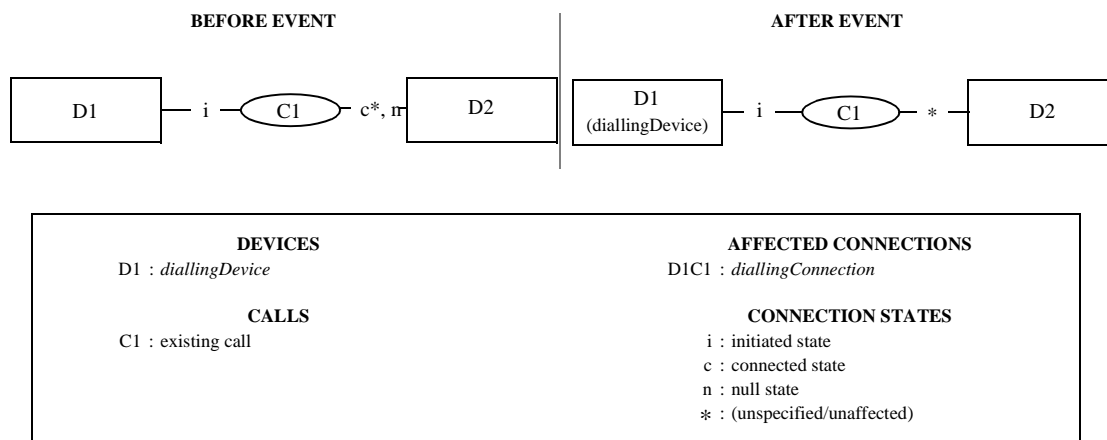
1. The Delivered event should not be used to determine when a device is physically ringing. (The Ringer Status event should be used for this.)
2. When a call is delivered to a bridged device configuration, multiple Delivered events are sent (one for each appearance of the device configuration). Each event will contain the following type of information:
 - The event for the appearance which is delivered the call first will have the appropriate event cause on why the call is being delivered to the device configuration. All subsequent events for the other appearances will have an event cause of Key Operation
 - The connection parameter will be the connection identifier of the specific appearance.
 - The alertingDevice will depend on the type of referencing model that is supported by the switching function (refer to 6.1.7, “Referencing Devices, Elements, Appearances and Device Configurations”, on page 28).
 - The calledDevice or associatedCalledDevice will contain the device identifier associated with the logical element of the device.
3. When observing a call or a device in a call, and the call diverts from a device in the call, if the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services option), the computing function shall use the alertingDevice, calledDevice, lastRedirectionDevice, and cause parameters to properly track the progress of the call as the result of the redirection. See 6.7.6, “Tracking a Diverted Call”, on page 41 for more information.
4. If there is more than one calling device (i.e. a conference calling back to a device), then the originatingNIDConnection parameter will not be present.
5. For external incoming calls, the contents of the networkCallingDeviceID and the networkCalledDeviceID parameters shall not change as long as the associatedCallingDevice remains in the call. This differs from the callingDeviceID and the calledDeviceID parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

17.2.6 Digits Dialed

The Digits Dialed event indicates that a call or feature is being attempted from a device and that a portion of the dialling sequence has been completed. It implies that only part of the input activity is complete, and that more of the dialling sequence needs to be supplied before the entire input activity is complete.

After the entire dialling sequence is complete, the Originated event will be generated in the case where a call is being attempted. The last Digits Dialed event received prior to receiving the Originated event will report the last digits dialed to complete the dialling sequence.

Figure 17-36 Digits Dialed Event



Note that the *connected state when D2 is a NID.

17.2.6.1 Event Parameters

Table 17-144 Digits Dialed—Event Parameters

Parameter Name	Type	M/ O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
diallingConnection	ConnectionID	M	Specifies the connection at which the digits were dialed.
diallingDevice	SubjectDeviceID	M	Specifies the device at which the digits were dialed.
diallingSequence	DeviceID	M	Specifies the sequence of digits that was dialed.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the device dialling the digits: initiated. This parameter is mandatory for events generated for device-type monitors and otherwise shall not be provided.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.

Table 17-144 Digits Dialed—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
diallingConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the diallingConnection connection. If this parameter is not present, then the connection information is switching function specific.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.2.6.2 Event Causes

Table 17-145 Digits Dialed—Event Causes

Event Cause	Description	Associated Features
Conference	Digits were dialed as part of a consultation for the purpose of a conference.	Consultation Call with consultOptions of ConferenceOnly
Consultation	Digits were dialed as part of a consultation.	Consultation Call
Network Dialling	Digits were dialed after a call has left the switching sub-domain.	External Calls
Normal	Digits were dialed (a more specific event cause cannot be provided).	Any feature, Dial Digits, Make Call
Transfer	Digits were dialed as part of a consultation for the purpose of a transfer.	Consultation Call with consultOptions of TransferOnly

17.2.6.3 Functional Requirements

1. This event will only be generated when the computing function has a device-type or call-type monitoring applied to a device that is initiating a call.
2. The grouping and number of digits reported in each event is switching function dependent.
3. The first Digits Dialed event includes the first digit specified in the Make Call or Consultation Call service which started the dialling sequence. If no digits were specified in the initiating Make Call or Consultation Call service (in the case where a null string was passed in the service request), the diallingSequence parameter in the Digits Dialed event contains a null string.

4. The digit sequence reported by the sequence of Digits Dialed events reflects the digits actually dialed through manual interaction or by CSTA service requests. It will not reflect any filtering or manipulation of the digit sequence by the switching function
5. If a Network Reached event is received prior to the receipt of an originated event, it may not be possible to determine the complete dialling sequence.
6. The switching function may have a timeout period for multi-stage dialling. If the dialling sequence does not complete prior to this timeout, it may either abort the call or attempt to use the digits already dialed and signal that dialling is complete with an originated event.

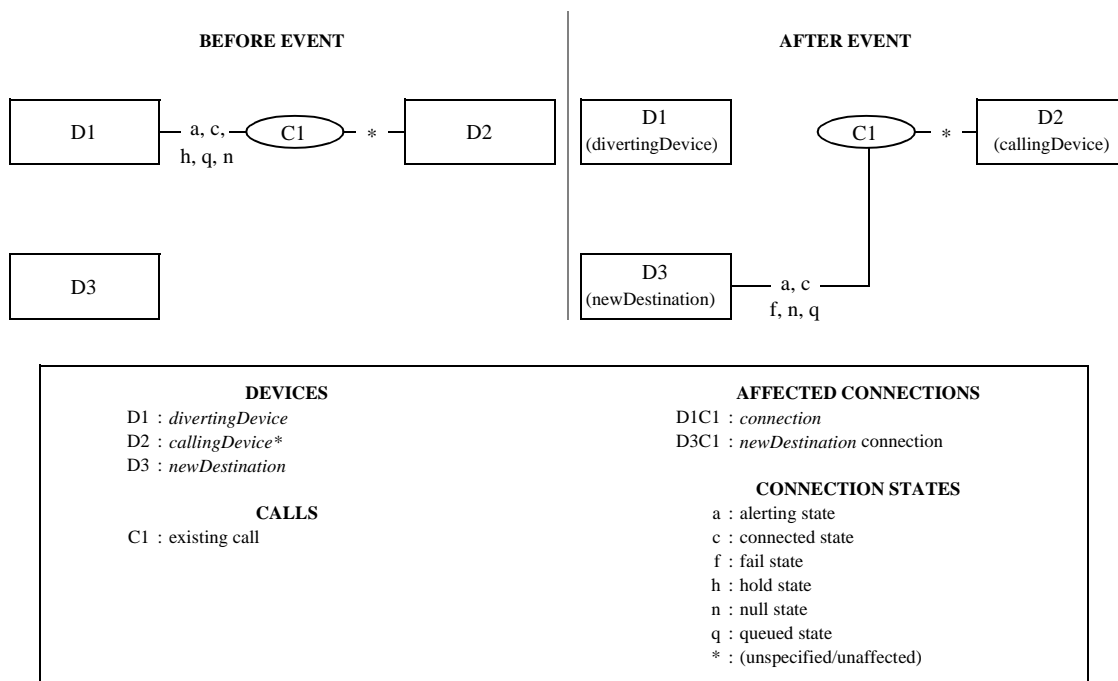
17.2.7 Diverted

The Diverted event indicates that a call has been diverted from a device and that the call is no longer present at the device.

Common situations that generate this event include:

- A call leaves a device that has some type of forwarding feature activated. Examples are Ring No Answer, Forward Immediate (depends upon the forwarding feature model supported by the switching function. See 6.7.1, “Forwarding”, on page 36), Recall, etc.
- A call leaves an ACD/Split/Hunt group device to be redirected to an agent, an extension, another ACD/Split/Hunt Group device, or to an offsite destination.
- A call leaves an ACD queue and is redirected to either an agent or an extension.
- A divert (Deflect, Pick, etc.) is successfully invoked.

Figure 17-37 Diverted Event



*There are some situations where D1 can also be the Calling Device (e.g. Make Predictive Call).

17.2.7.1 Event Parameters

Table 17-146 Diverted—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
connection	ConnectionID	M	Specifies the connection that was diverted.
divertingDevice	SubjectDeviceID	M	Specifies the device from which the call was diverted.
newDestination	SubjectDeviceID	M	Specifies the device to which the call was diverted.

Table 17-146 Diverted—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
callingDevice	CallingDeviceID	C	Specifies a device remaining in the call with the newDestination device. This parameter shall be provided in the case of Immediate Forwarding, where forwarding is triggered after the call is delivered to a device (see Function Requirement #3). Otherwise the parameter is optional.
calledDevice	CalledDeviceID	C	Specifies the originally called device. This parameter shall be provided in the case of Immediate Forwarding, where forwarding is triggered after the call is delivered to a device (see Function Requirement #3). Otherwise the parameter is optional.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected from device. Note that this is not the divertingDevice that caused this event, but the device that performed a redirection towards the current divertingDevice just before the current redirection.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the diverting device: Null • For the other devices left in the call: (unaffected) This parameter is mandatory for events generated for device-type monitors and otherwise shall not be provided.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
connectionInfo	ConnectionInformation	O	Specifies the connection information associated with the connection. If this parameter is not present, then the connection information is switching function specific.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.

Table 17-146 Diverted—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.7.2 Event Causes

Table 17-147 Diverted—Event Causes

Event Cause	Description	Associated Features
ACD Forwarding	The call was diverted from one ACD device to a new ACD device and is no longer queued at the previous ACD device.	ACD
Call Forward	The call was diverted from a device after it was forwarded from another device (any type of forwarding).	Call Forwarding
Call Forward—Busy	The call was diverted from a device after it was forwarded (forwarding on a busy condition).	Call Forwarding
Call Forward—Immediate	The call was diverted from a device after it was forwarded (forwarding on all conditions).	Call Forwarding
Call Forward—No Answer	The call was diverted from a device after it was forwarded because of a no answer condition.	Call Forwarding
Call Not Answered	The call was diverted from a device because it was not answered before a timer elapsed.	Recall
Call Pickup	The call was diverted from a device because of the pickup feature.	Directed Pickup Call, Group Pickup Call
Distributed	The call was diverted from a device because it was distributed by a distribution group (ACD or Hunt group).	ACD
Do Not Disturb	The call was diverted from a device because the device had the Do Not Disturb feature set.	Call Forwarding
No Agent Available	The call was diverted from a device because there were no available devices in a group (ACD).	ACD
No Resources Available	The call was diverted from a device because there were no resources available to process it.	ACD
Normal	The call was diverted from a device (a more specific cause cannot be provided).	Any feature
Overflow	The call was diverted from a device after it overflowed a queue, group, or target.	ACD

Table 17-147 Diverted—Event Causes (continued)

Event Cause	Description	Associated Features
Park	The park feature has been invoked to either park or un-park a call at a device.	Park Call
Recall	The call was diverted from a device as part of a recall feature (e.g., due to a time-out associated with a feature that failed to complete).	Recall
Redirected	The call was diverted from a device because of a deflect or divert feature.	Deflect Call, Divert Call

17.2.7.3 Functional Requirements

1. Based upon the switching function (as indicated through the capabilities exchange services), the switching function may send the Diverted event to either:
 - Only the divertingDevice (D1) for device-type monitors, and not for call-type monitors, or
 - To all devices in a call (device-type monitors) and for call-type monitors.
2. When observing a call or a device in a call, and the call previously diverts from a device in the call, if the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services option), the computing function shall use the divertingDevice, calledDevice, lastRedirectionDevice, and cause parameters to properly track the progress of the call as a result of the previous redirection. See 6.7.6, “Tracking a Diverted Call”, on page 41 for more information.
3. In the case of Immediate Forwarding, where forwarding is triggered *after* the call is delivered to a device (as indicated by the capability exchange services), the Diverted event for the Diverting device shall contain a cause of Immediate Forwarding and a localConnectionInfo of Null. Since this Diverted event is the first and last event generated for the call at the Diverting device, it is an example of a Null to Null connection state transition.
4. For external incoming calls, the contents of the networkCallingDeviceID and the networkCalledDeviceID parameters shall not change as long as the associatedCallingDevice remains in the call. This differs from the callingDeviceID and the calledDeviceID parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

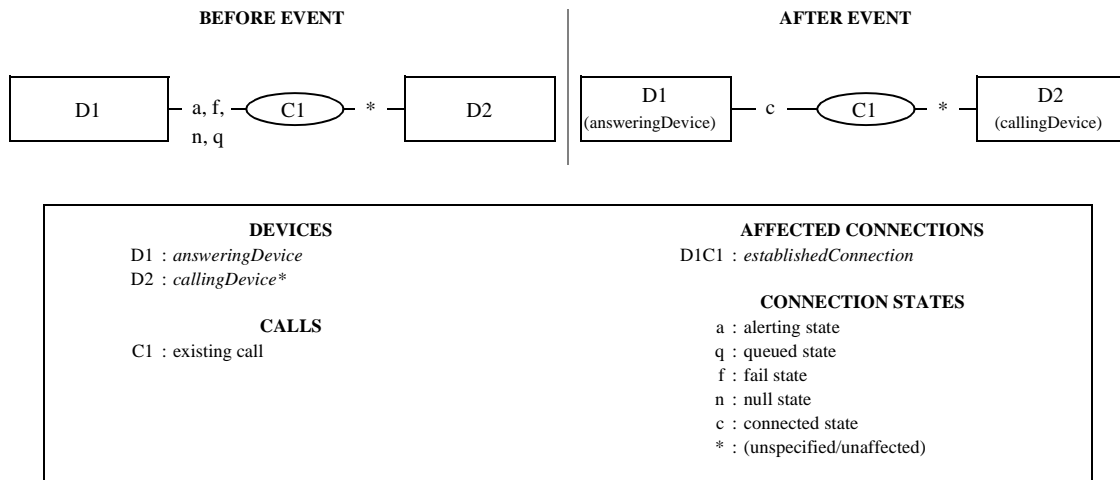
17.2.8 Established

The Established event indicates that a call has been answered at a device or that a call has been connected to a device.

Common situations that generate this event include:

- A call has been answered at a device (e.g., a user has manually gone off-hook).
- The Answer Call service has been successfully invoked.
- A call has been picked by another device.

Figure 17-38 Established Event



*There may be some situations where D1 can be the calling device (e.g. Make Predictive Call).

17.2.8.1 Event Parameters

Table 17-148 Established—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
establishedConnection	ConnectionID	M	Specifies the connection that was connected.
answeringDevice	SubjectDeviceID	M	Specifies the device that connected into the call.
callingDevice	CallingDeviceID	M	Specifies the calling device.
calledDevice	CalledDeviceID	M	Specifies the originally called device.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected from device.
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from. If this parameter is supported, it shall be provided if there is an originating NID associated with the call, otherwise it shall not be provided. See Functional Requirement #4.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the answering device: Connected • For the calling device: (unaffected - never Null) This parameter is mandatory for events generated for device-type monitors and otherwise shall not be provided.

Table 17-148 Established—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
establishedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the establishedConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.8.2 Event Causes

Table 17-149 Established—Event Causes

Event Cause	Description	Associated Features
ACD Forward	The call was established at a device (immediately, before being delivered) and is no longer queued at the previous ACD device.	ACD

Table 17-149 Established—Event Causes (continued)

Event Cause	Description	Associated Features
Alternate	The call was reestablished or answered at a device due to the alternate feature.	Alternate Call
Call Back	The call was established at a device (immediately, before being delivered) because of a previously set call back feature.	Call Back Call-Related
Call Forward	The call was established at a device (immediately, before being delivered) after it was forwarded (any type of forwarding).	Call Forwarding
Call Forward—Busy	The call was established at a device (immediately, before being delivered) after it was forwarded because of a busy condition.	Call Forwarding
Call Forward—Immediate	The call was established at a device (immediately, before being delivered) after it was forwarded (forwarding on all conditions).	Call Forwarding
Call Forward—No Answer	The call was established at a device (immediately, before being delivered) after it was forwarded because of a no answer condition.	Call Forwarding
Call Pickup	The call was established at a device via a pickup feature.	Directed Pickup Call, Group Pickup Call
Distributed	The call was distributed, (and was established immediately, before being delivered) to a device because the call has moved from the distribution mechanism (ACD, Hunt) to an available device associated with the distribution mechanism	ACD, Routeing Services
Intrude	The call was established at a device because the Intrude Call service was executed successfully.	Intrude Call
Key Operation	The call was established at a device that has an appearance in a call appearance, bridged, or hybrid device configuration.	Multiple Appearance
Network Signal	The call was established at a device that is outside of the switching sub-domain.	External Calls
New Call	The call was not redirected.	Any Feature
No Agent Available	The call was established immediately, before being delivered, at a device because there were no available devices in a group (ACD)	ACD, Forwarding
Normal	The call was established at a device (a more specific cause cannot be provided).	Any feature
Overflow	The call was established at a device (immediately, before being delivered) after it overflowed a queue, group, or target.	ACD
Override	The call was established at a device as a result of an override (e.g., Single Step Conference Call, Intrude Call) feature.	Intrude Call
Recall	The call was established at a device as part of the recall feature (e.g., due to a time-out associated with a feature that failed to complete) the call was established immediately (e.g. auto answer), before being delivered.	Recall
Redirected	The call was established at a device after it was previously diverted from or deflected to this device the call was established immediately, before being delivered.	Deflect Call, Divert Call

Table 17-149 Established—Event Causes (continued)

Event Cause	Description	Associated Features
Resources Not Available	The call was established at a device (immediately, before being delivered) because some resources were not available (ACD, forwarding).	ACD, Forwarding
Remains in Queue	The call was established at a device and the calling device's position in the ACD queue is retained and the call will be routed to an agent when one becomes available. For example, the call could be established at a VRU or other automated answering equipment while the calling device retains its position in the ACD queue.	ACD
Single Step Conference	The call was established at a device (immediately, before being delivered) due to the Single Step Conference feature.	Single Step Conference
Single Step Transfer	The call was established at a device (immediately, before being delivered) due to the Single Step Transfer feature.	Single Step Transfer
Timeout	The event was generated because a trunk timer expired. It is not generated as the result of a network event or condition.	External Call

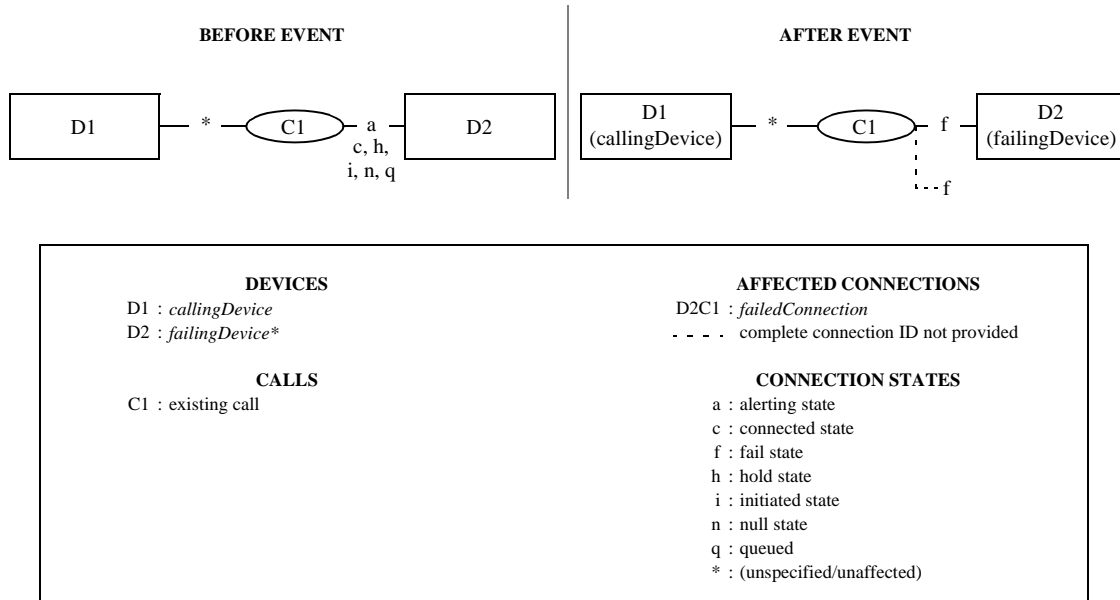
17.2.8.3 Functional Requirements

1. A computing function will not see an Established event after a Service Initiated event (e.g. because of Make Call with prompting) indicating that the calling device is connected in the call. Instead, the computing function will see an Originated event indicating connection into the call.
2. When an appearance from a bridged device configuration connects into a call, an Established event is sent. The event will contain the following type of information:
 - The event cause will be Normal.
 - The establishedConnection will be the connection identifier of the specific appearance.
 - The answeringDevice will depend on the type of referencing model that is supported by the switching function (refer to 6.1.7, “Referencing Devices, Elements, Appearances and Device Configurations”, on page 28).
 - The calledDevice or associatedCalledDevice will contain the device identifier associated with the logical element of the device.
3. When observing a call or a device in a call, and the call diverts from a device in the call, if the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services option), the computing function shall use the answeringDevice, calledDevice, lastRedirectionDevice, and cause (eventCause parameter type) parameters to properly track the progress of the call as a result of the redirection. See 6.7.6, “Tracking a Diverted Call”, on page 41 for more information.
4. If there is more than one calling device (i.e. a conference calling back to a device), then the originatingNIDConnection parameter will not be present.
5. For external incoming calls, the contents of the networkCallingDeviceID and the networkCalledDeviceID parameters shall not change as long as the associatedCallingDevice remains in the call. This differs from the callingDeviceID and the calledDeviceID parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

17.2.9 Failed

The Failed event indicates that a call cannot be completed or a connection has entered the Fail state for any of the reasons described in the Event Causes table on page 223.

Figure 17-39 Failed Event



Refer to 6.7.2, “Connection Failure”, on page 38 for a complete description of partial connection IDs.

In some situations D2 can be the calling device.

17.2.9.1 Event Parameters

Table 17-150 Failed—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
failedConnection	ConnectionID	M	Specifies the connection that failed. (See Functional Requirement #3 at the end of this section.)
failingDevice	SubjectDeviceID	M	Specifies the device that failed.
callingDevice	CallingDeviceID	M	Specifies the calling device.
calledDevice	CalledDeviceID	M	Specifies the originally called device.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected-from device.
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from. If this parameter is supported, it shall be provided if there is an originating NID associated with the call, otherwise it shall not be provided.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> For the failing device: Fail For the other devices left in the call: (unaffected) This parameter is mandatory for events generated for device-type monitors and otherwise shall not be provided.

Table 17-150 Failed—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the otherDevice if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
failedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the failedConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.2.9.2 Event Causes

Table 17-151 Failed—Event Causes

Event Cause	Description	Associated Features
Blocked	The call failed after a device has disconnected from a call leaving one other device remaining in the call.	Connection Failure

Table 17-151 Failed—Event Causes (continued)

Event Cause	Description	Associated Features
Busy	The call failed after it encountered a busy or unavailable device.	Connection Failure
Call Cancelled	The call failed before the associated device has gone on-hook.	Connection Failure
Call Not Answered	The call failed because it was not answered before a timer expired.	Connection Failure
Destination Not Obtainable	The call failed because it could not reach the destination.	Connection Failure
Destination Out of Order	The call failed because it encountered a destination out of service.	Connection Failure
Do Not Disturb	The call failed because it encountered a device that has the do not disturb feature set.	Do Not Disturb, Call Forwarding
Incompatible Destination	The call failed because it encountered an incompatible destination.	Connection Failure
Invalid Account Code	The call failed because of an invalid account code.	Connection Failure
Invalid Number Format	The call failed because the dialled number is incorrect	Connection Failure
Key Operation in Use	The call failed because an appearance is associated with an exclusive bridged device configuration and that the appearance is disabled until the call is terminated or until the call moves away from the device.	Multiple Appearance
Lockout	The call failed because it encountered an inter-digit time-out while dialling.	Connection Failure
Maintenance	The call failed because it encountered a facility or endpoint in a maintenance condition.	Connection Failure
Network Congestion	The call failed because it encountered a congested network. In some circumstances, this event cause indicates that the user is listening to a special signal tone from a network. The tone may be accompanied by a voiced statement similar to "All circuits are busy..."	Connection Failure
Network Not Obtainable	The call failed because it could not reach a destination network.	Connection Failure
Network Out of Order	The call failed because it encountered a network that is out of order.	Connection Failure
Network Signal	The call failed because it encountered a problem after it left the switching sub-domain.	External Calls
Normal	Normal cause for event.	Any feature
Not Available Bearer Service	The call failed because it was requested with a bearer capability that is currently not available.	Connection Failure
Not Supported Bearer Service	The call failed because it was requested with a bearer capability that is currently not supported.	Connection Failure
Number Changed	The call failed because the called number has been changed to a new number and the call cannot be completed.	Connection Failure
Number Unallocated	The call failed because the called number is not allocated to a subscriber.	Connection Failure

Table 17-151 Failed—Event Causes (continued)

Event Cause	Description	Associated Features
Reorder Tone	<p>The call failed because it encountered a reorder condition.</p> <p>When this occurs, the network usually provides Reorder Tone to indicate that a request (call, feature, or supplementary service) was not recognizable. This condition typically results when a user dials a number that is not valid or attempts to obtain a service that is not enabled for that user or device.</p>	Connection Failure
Resources Not Available	The call failed because resources were not available.	Connection Failure
Selected Trunk Busy	The call failed because a specific selected Network Interface Device (e.g., trunk, CO line) is busy.	Connection Failure
Trunks Busy	The call failed because there is no available Network Interface Device (e.g., trunk, CO line).	Connection Failure
Unauthorized Bearer Service	The call failed because it was requested with an unauthorized bearer capability.	Connection Failure

17.2.9.3 Functional Requirements

1. With an exclusive bridged device configuration, a Failed event is generated for each appearance that does not enter the call. Each event will contain the following type of information:
 - The event cause will be Key Operation In Use.
 - The failedConnection will be the connection identifier of the specific appearance.
 - The failingDevice will depend on the type of referencing model that is supported by the switching function (refer to 6.1.7, “Referencing Devices, Elements, Appearances and Device Configurations”, on page 28).
 - The calledDevice or associatedCalledDevice will contain the device identifier associated with the logical element of the device.
2. When observing a call or a device in a call, and the call diverts from a device in the call, if the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services option), the computing function shall use the failingDevice, calledDevice, lastRedirectionDevice, and cause (EventCause parameter type) parameters to properly track the progress of the call as a result of the redirection. See 6.7.6, “Tracking a Diverted Call”, on page 41 for more information.
3. The content of the failedConnection parameter, depending on whether or not the Failed event is reported to the failing device’s monitor, will contain one of the following types of connection identifiers (use the capabilities exchange services to determine which approach is supported by a particular switching function):
 - a. A complete connection identifier (i.e., an identifier that contains both a device identifier and call identifier). This indicates that a connection is made with the failing device, and that the Failed event will be reported to all active device-type monitors associated with the call, as well as all call-type monitors associated with the call.
 - b. A call identifier only connection identifier (i.e., an identifier that contains a valid call identifier and no device identifier). This indicates that a connection is not made with the failing device, and that the Failed event will only be reported to the active device and call-type monitors associated with the devices that were in the call prior to the failure (i.e., if a device-type monitor was on the failing device, then the Failed event is not reported).

- c. A complete connection identifier (i.e., an identifier that contains both a device identifier and call identifier), not being reported for monitors on the failing device. This indicates that the Failed event will only be reported to the active device and call-type monitors associated with the devices that were in the call prior to the failure (i.e., if a device-type monitor was on the failing device, then the Failed event is not reported).
4. For external incoming calls, the contents of the `networkCallingDeviceID` and the `networkCalledDeviceID` parameters shall not change as long as the `associatedCallingDevice` remains in the call. This differs from the `callingDeviceID` and the `calledDeviceID` parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

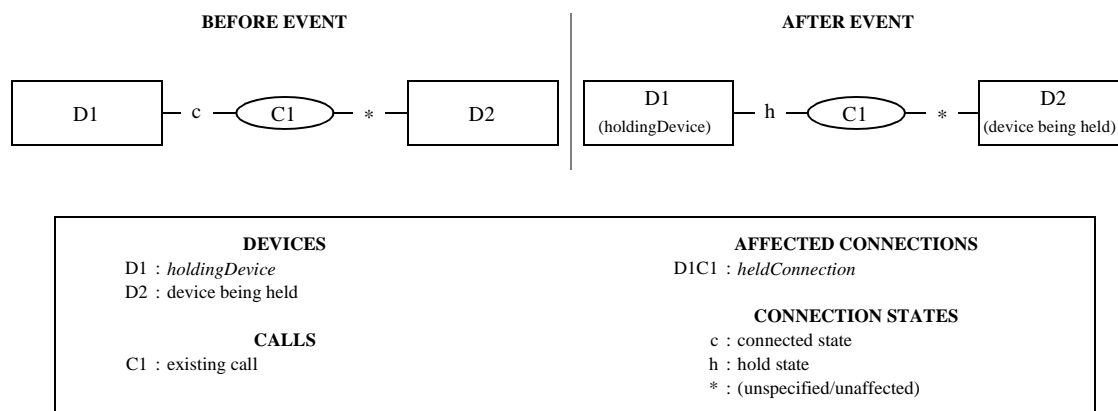
17.2.10 Held

The Held event indicates that a call has been placed on hold.

Common situations that generate this event include:

- Consultation situations (manual and service initiated).
- Hold situations (manual and service initiated).

Figure 17-40 Held Event



17.2.10.1 Event Parameters

Table 17-152 Held—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
heldConnection	ConnectionID	M	Specifies the connection at which the hold was activated.
holdingDevice	SubjectDeviceID	M	Specifies the device at which hold was activated.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the holding device: Hold • For the other devices left in the call: (unaffected) This parameter is mandatory for events generated for device-type monitors.
correlatorData	CorrelatorData	O	Specifies the correlator data associated with the call.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.

Table 17-152 Held—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
heldConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the heldConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies on-standardized information.

17.2.10.2 Event Causes

Table 17-153 Held —Event Causes

Event Cause	Description	Associated Features
Alternate	The call was held at a device as part of an alternate feature.	Alternate Call
Conference	The call was held at a device as a part of a consultation with the intended purpose of establishing a conference.	Consultation Call with a consultOptions of ConferenceOnly
Consultation	The call was held at a device as part of a consultation.	Consultation Call
Intrude	The call was held at a device because the Intrude Call service was executed successfully.	Intrude Call
Maintenance	The call was placed on hold at a device that entered maintenance conditions (e.g. mobile handset out of coverage)	Maintenance
Network Signal	The call was held by a device that is outside of the switching sub-domain.	External Call
Normal	The call was held at a device (a more specific cause cannot be provided).	Any feature
Recall	The call was held at a device due to the activation of the recall feature.	Recall
Suspend	The call was temporarily placed on hold at a device that went onhook.	Call Clearing
Transfer	The call was held at a device as part of a transfer feature.	Consultation Call with a consultOptions of Transfer Only

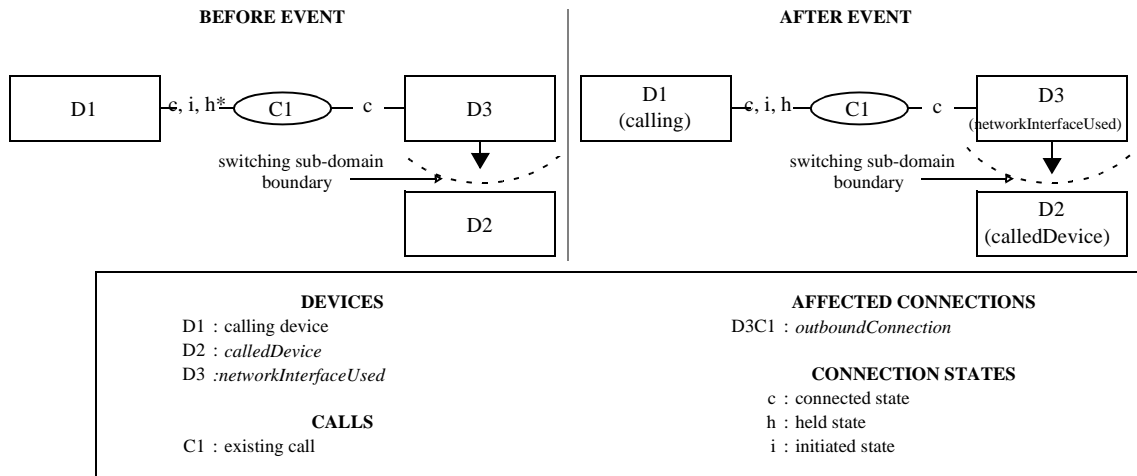
17.2.10.3 Functional Requirements

17.2.11 Network Capabilities Changed

The Network Capabilities Changed event indicates that a situation occurred during a call's progress in a public or private network that modifies its signalling capability (i.e., inter-networking). It does not indicate a change in the connection state of the Network Interface Device (e.g., trunk, CO Line) through which the call has accessed the network.

This event shall always be preceded by a Network Reached event that included the networkCapability parameter. The event may be repeated according to the situations encountered in the network.

Figure 17-41 Network Capabilities Changed Event



Note that the * held state when an external outgoing call in progress is placed held.

17.2.11.1 Event Parameters

Table 17-154 Network Capabilities Changed—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
outboundConnection	ConnectionID	M	Specifies the outbound connection of the network interface device.
networkInterfaceUsed	SubjectDeviceID	M	Specifies the Network Interface Device (e.g., trunk, CO Line) that was selected.
calledDevice	CalledDeviceID	M	Specifies the destination device.

Table 17-154 Network Capabilities Changed—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
progressIndicator	Structure	M	<p>Specifies an interworking situation encountered in the public or private network outside the switching sub-domain. As a consequence of entering the network, the level of event reporting associated with this call may be reduced. This parameter consists of the following components:</p> <ol style="list-style-type: none"> 1. progressLocation (M) - It may be one of the following: <ul style="list-style-type: none"> • User • Private network serving the local user • Public network serving the local user • Transit network • Public network serving the remote user • Private network serving the remote user • Local interface controlled by the signalling link • International Network • Network beyond interworking point • Other 2. progressDescription (M) - It may be one of the following: <ul style="list-style-type: none"> • ISDN Progress Description - This information is derived from ETSI ETS 300 182: 1993. • QSIG Progress Description - This information is derived from ECMA-143. • Other
localConnectionInfo	LocalConnectionState	C	<p>Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call.</p> <ul style="list-style-type: none"> • For the network interface device: Connected • For the other devices left in the call: (Unaffected) <p>This parameter is mandatory for events generated for device-type monitors.</p>
correlatorData	CorrelatorData	C	<p>Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.</p>
userData	UserData	C	<p>Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.</p>
networkCapability	Structure	O	<p>Specifies the type of network reached and the Call Control events supported by the network. It includes the following components:</p> <ol style="list-style-type: none"> 1. networkType (M) - The complete set of possible values is: <ul style="list-style-type: none"> • ISDN public • Non-ISDN public • ISDN private • Non-ISDN private • Other 2. eventsProvided (O) - This is a bitmap of all of the Call Control events defined in this Standard.
cause	EventCause	M	<p>Specifies the reason for the event.</p>

Table 17-154 Network Capabilities Changed—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, "MediaCallCharacteristics", on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, "CallCharacteristics", on page 61 for the complete set of possible values.
outboundConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the outboundConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

17.2.11.2 Event Causes

Table 17-155 Network Capabilities Changed—Event Causes

Event Cause	Description	Associated Features
Network Signal	The call encountered an interworking situation.	External Call

17.2.12 Network Reached

The Network Reached event indicates that a call has cut through the switching sub-domain boundary to another network; that is, has reached and engaged a Network Interface Device (e.g., trunk, CO Line). This event indicates that there may be a reduced level of event reporting and possibly no additional device feedback, except connection/call clearing, provided for this device in the call due to a lack of network signalling. The level of signalling provided by the network may be indicated by the networkCapability parameter.

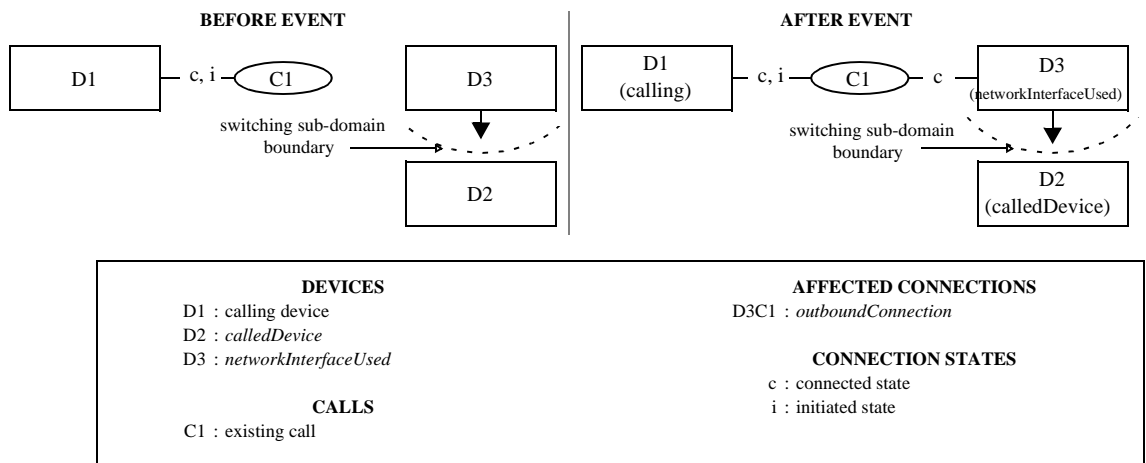
Additionally, the computing function should assume that it cannot directly manipulate the far-end device associated with the Network Interface Device.

This event is never sent for calls made to devices that are within the switching sub-domain. This event indicates that a connection with a Network Interface Device has reached the connected state, and that further events for that connection refer to the state of the endpoint which the Network Interface Device is associated.

A common situation that generates this event includes:

- An outgoing call has cut-through at a network interface device and further call progress information, such as the Delivered and Established events, may not be available.

Figure 17-42 Network Reached Event



17.2.12.1 Event Parameters

Table 17-156 Network Reached—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
outboundConnection	ConnectionID	M	Specifies the outbound connection associated with the call that is leaving the switching sub-domain.
networkInterfaceUsed	SubjectDeviceID	M	Specifies the Network Interface Device that was selected.
callingDevice	CallingDeviceID	M	Specifies the calling device.
calledDevice	CalledDeviceID	M	Specifies the originally called device.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected from device.

Table 17-156 Network Reached—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from. If this parameter is supported, it shall be provided if there is an originating NID associated with the call, otherwise it shall not be provided.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the Network Interface Device: Connected • For the other devices left in the call: (unaffected) This parameter is mandatory for events generated for device-type monitors.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
networkCapability	Structure	O	Specifies the type of network reached and the Call Control events supported by the network. It includes the following components: <ol style="list-style-type: none"> 1. networkType (M) - The complete set of possible values is: <ul style="list-style-type: none"> • ISDN public • Non-ISDN public • ISDN private • Non-ISDN private • Other 2. eventsProvided (O) - This is a bitmap of the Call Control events defined in this Standard.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, "MediaCallCharacteristics", on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, "CallCharacteristics", on page 61 for the complete set of possible values.
outboundConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the outboundConnection connection. If this parameter is not present, then the connection information is switching function specific.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls (that, in this case are also outgoing calls).

Table 17-156 Network Reached—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls (that, in this case are also outgoing calls).
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call (that, in this case is also an outgoing). This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.12.2 Event Causes

Table 17-157 Network Reached—Event Causes

Event Cause	Description	Associated Features
ACD Forward	The call left the switching sub-domain because it was forwarded from one ACD device to another ACD device and is no longer present at the previous ACD device.	ACD
Distributed	The call left the switching sub-domain after it was distributed by an ACD or hunt group.	ACD
Call Forward	The call left the switching sub-domain after it was forwarded (any type of forwarding).	Forwarding
Call Forward—Busy	The call left the switching sub-domain after it was forwarded because of a busy condition.	Forwarding
Call Forward—Immediate	The call left the switching sub-domain after it was forwarded (forwarding on all conditions).	Forwarding
Call Forward—No Answer	The call left the switching sub-domain after it was forwarded because of a no answer condition.	Forwarding
No Agents Available	The call left the switching sub-domain after it was diverted from a device because there were no available devices in a group (ACD).	ACD, distribution
No Resources Available	The call left the switching sub-domain after it was diverted from a device because there were no resources available to process it.	ACD, distribution
Normal	The call left the switching sub-domain (a more specific event cause cannot be provided).	Any feature
Overflow	The call left the switching sub-domain after it overflowed a queue, group, or target.	ACD
Redirected	The call left the switching sub-domain after it was diverted or deflected.	Deflect Call, Divert Call
Transfer	The call was transferred to a destination outside the switching sub-domain.	Transfer

17.2.12.3 Functional Requirements

1. When observing a call or a device in a call, and the call diverts from a device in the call, if the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services option), the computing function shall use the

networkInterfaceUsed, calledDevice, lastRedirectionDevice, and cause (EventCause parameter type) parameters to properly track the progress of the call as a result of the redirection. See 6.7.6, "Tracking a Diverted Call", on page 41 for more information.

2. After a call has cut through the switching sub-domain boundary to another network (Network Reached event generated), all subsequent events reported for the endpoint to which the Network Interface Device (e.g., trunk, CO Line) is associated shall include a cause parameter with a cause of Network Signal or a more specific cause representing network information.
3. For external incoming calls, the contents of the networkCallingDeviceID and the networkCalledDeviceID parameters shall not change as long as the associatedCallingDevice remains in the call. This differs from the callingDeviceID and the calledDeviceID parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

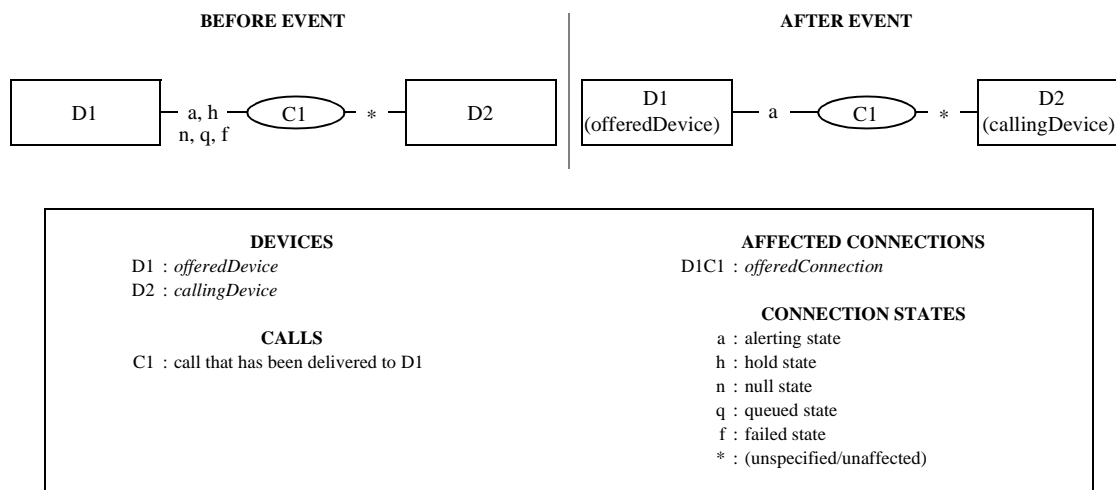
17.2.13 Offered

The Offered event indicates that the connection is in the Offered mode of the alerting state. This indicates that a call is in a pre-delivery state.

In this pre-delivery state, the opportunity exists for a computing function to issue one of a set of supported services (e.g., Accept Call, Clear Connection (“reject”), Deflect Call) or an ISDN device to accept or reject the call. From the calling side perspective, the call is not delivered at the called device. As a consequence, delivery information such as Ringback indication and/or Network signalling is not provided. For example, the device makes no ringing sounds while in the Offered mode of the Alerting state.

The connection may transit to the Ringing mode of the alerting state after the call is accepted (See 17.1.1, “Accept Call”, on page 116) or after a time out period, for example.

Figure 17-43 Offered Event



17.2.13.1 Event Parameters

Table 17-158 Offered—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
offeredConnection	ConnectionID	M	Specifies the connection that is alerting.
offeredDevice	SubjectDeviceID	M	Specifies the device that is alerting.
callingDevice	CallingDeviceID	M	Specifies the calling device.
calledDevice	CalledDeviceID	M	Specifies the originally called device.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected from device.
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from. If this parameter is supported, it shall be provided if there is an originating NID associated with the call, otherwise it shall not be provided. See Functional Requirement #3.

Table 17-158 Offered—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the alerting device: Alerting • For the calling device: (unaffected) This parameter is mandatory for events generated for device-type monitors.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
offeredConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the offeredConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.13.2 Event Causes

Table 17-159 Offered—Event Causes

Event Cause	Description	Associated Features
ACD Busy	The call was offered to an ACD device that has currently no available agents.	ACD, Consultation Call
ACD Forward	The call was offered to a new device and is no longer queued at the previous ACD device.	ACD
ACD Saturated	The call was offered to an ACD device that has currently no internal resources available (including agents).	ACD, Consultation Call
Call Back	The call was offered to a device because of a previously set call back feature.	Call Back Call-Related
Call Forward	The call was offered to a device after it was forwarded (any type of forwarding).	Call Forwarding
Call Forward—Busy	The call was offered to a device after it was forwarded because of a busy condition.	Call Forwarding
Call Forward—Immediate	The call was offered to a device after it was forwarded (forwarding on all conditions).	Call Forwarding
Call Forward—No Answer	The call was offered to a device after it was forwarded because of a no answer condition.	Call Forwarding
Distributed	The call was offered to a device because the call has moved from the distribution mechanism (ACD, Hunt) to an available device associated with the distribution mechanism.	ACD, Routeing Services
Entering Distribution	The call was offered to a distribution mechanism (ACD, Hunt) for the purpose of being distributed.	ACD, Routeing Services
Key Operation	The call was offered to a device that has an appearance in a call appearance, bridged, or hybrid device configuration.	Multiple Appearance
Multiple Alerting	The call that is alerting the subject device is also alerting other devices.	Multiple Alerting
Network Signal	The call was offered to a device that is outside of the switching sub-domain.	External Calls
New Call	The call was not redirected.	Any feature
No Agent Available	The call was offered to a device because there were no available devices in a group (ACD).	ACD, Forwarding
Normal	The call was offered to a device (a more specific cause cannot be provided).	Any feature
Overflow	The call was offered to a device after it overflowed a queue, group, or target.	ACD
Override	The call was offered to a device as a result of an override (e.g., Intrude Call) feature.	Intrude Call
Recall	The call was offered to a device as part of the recall feature (e.g., due to a time-out associated with a feature that failed to complete).	Recall
Redirected	The call was offered to a device after it was diverted from or deflected to this device.	Deflect Call, Divert Call
Remains in Queue	The call was offered to a device while the calling device's position in the queue is retained. For example, the call could be offered to a VRU or other automated answering equipment while the calling device retains its position in the ACD queue.	ACD, Queuing, Single Step Conference, Join Call

Table 17-159 Offered—Event Causes (continued)

Event Cause	Description	Associated Features
Resources not available	The call was offered because some resources were not available (ACD, forwarding).	ACD, Forwarding
Single Step Transfer	The call alerted the device due to a Single Step Transfer service.	Single Step Transfer
Single Step Conference	The call alerted the device due to a Single Step Conference service.	Single Step Conference
Timeout	The event was generated because a trunk timer expired. It is not generated as the result of a particular network event or condition.	External calls

17.2.13.3 Functional Requirements

1. Switching function implementations vary in their support for call control services while the Offered mode of the alerting state. The computing function shall determine supported services for Offered mode through the capabilities exchange services.
2. The switching function may support an Offered mode time-out. For example, if no service is applied to the connection within an switching function-specific period, it will transit from the Offered mode to the Ringing mode of the alerting state.
3. If there is more than one calling device (i.e. a conference calling back to a device), then the originatingNIDConnection parameter will not be present.
4. When a call is offered to a bridged device configuration, multiple Offered events are sent (one for each appearance of the device configuration). Each event will contain the following type of information:
 - The event for the appearance which is offered the call first will have the appropriate event cause on why the call is being offered to the device configuration. All subsequent events for the other appearances will have an event cause of Key Operation.
 - The offeredConnection will be the connection identifier of the specific appearance.
 - The offeredDevice will depend on the type of referencing model that is supported by the switching function (refer to 6.1.7, “Referencing Devices, Elements, Appearances and Device Configurations”, on page 28).
 - The calledDevice or associatedCalledDevice will contain the device identifier associated with the logical element of the device.
5. When observing a call or a device in a call, and the call diverts from a device in the call, if the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services option), the computing function shall use the offeredDevice, calledDevice, lastRedirectionDevice, and cause (eventCause parameter type) parameters to properly track the progress of the call as a result of the redirection. See 6.7.6, “Tracking a Diverted Call”, on page 41 for more information.
6. For external incoming calls, the contents of the networkCallingDeviceID and the networkCalledDeviceID parameters shall not change as long as the associatedCallingDevice remains in the call. This differs from the callingDeviceID and the calledDeviceID parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

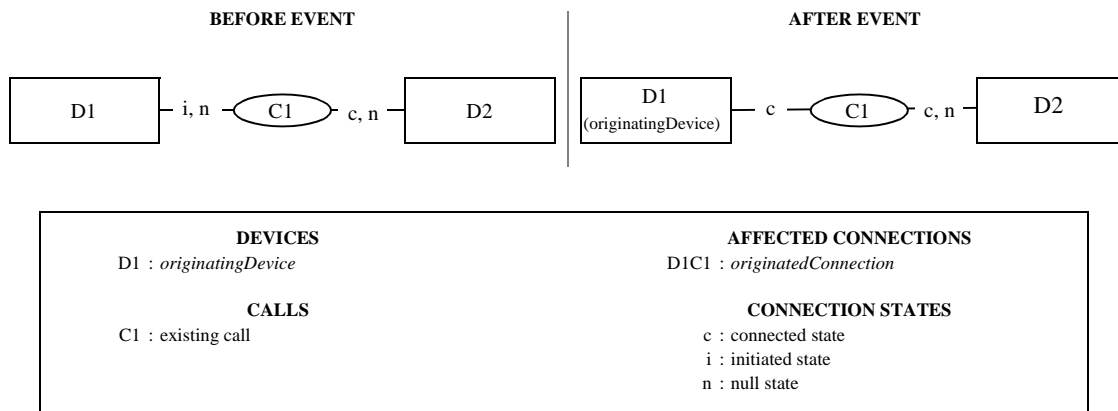
17.2.14 Originated

The Originated event indicates that a call is being attempted from a device. It implies that input activity for the call is complete and that a call (rather than a feature) has been requested.

Common situations that generate this event when the switch has originated a call at the originating device are:

- Due to the execution of the Make Call service.
- Due to the execution of the Consultation Call service.
- After a user has completed manually dialling a number.
- When an external incoming call originates from a Network Interface Device (e.g., trunk, CO line)

Figure 17-44 Originated Event



Note that the connected state for D2C1 is when D2 is a NID.

17.2.14.1 Event Parameters

Table 17-160 Originated—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
originatedConnection	ConnectionID	M	Specifies the connection at which the call originated.
callingDevice	SubjectDeviceID	M	Specifies the calling device.
calledDevice	CalledDeviceID	M	Specifies the originally called device.
originatingDevice	DeviceID	O	Specifies the originating device. The originating device represents a device that originated a call in the switching sub-domain on behalf of the calling device (e.g., a group of stations). The originating device shall not represent a Network Interface Device. This parameter may be provided for external outgoing and internal calls when the originating device is different from the calling device. This parameter shall not be provided for external incoming calls or when the originating device is identical to the calling device.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the device initiating the call: Connected This parameter is mandatory for events generated for device-type monitors.

Table 17-160 Originated—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
originatedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the originatedConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.14.2 Event Causes

Table 17-161 Originated—Event Causes

Event Cause	Description	Associated Features
Call Back	A call was originated from a device because a previously set call back feature.	Call Back Call-Related, Call Back Non-Call-Related
Conference	A call was originated from a device as part of consultation with the intended purpose of establishing a conference.	Consultation Call with a consultOptions of ConferenceOnly

Table 17-161 Originated—Event Causes (continued)

Event Cause	Description	Associated Features
Consultation	A call was originated from a device as part of a consultation call.	Consultation Call
Make Call	A call was originated from a device as the result of the Make Call service.	Make Call (without prompting)
New Call	A call was originated.	Any feature
Normal	A call was originated from a device (a more specific event cause cannot be provided).	Any feature, Make Call
Transfer	A call was originated from a device as part of a consultation for the purpose of transferring a call.	Consultation Call with a consultOptions of TransferOnly

17.2.14.3 Functional Requirements

1. This event will only be generated when the computing function has a device-type or call-type monitoring applied to a device that is initiating a call.
2. This event will only be generated on an incoming call from a device outside the switching sub-domain when the device is a Network Interface Device that can have monitoring applied to it.
3. For external incoming calls, the contents of the networkCallingDeviceID and the networkCalledDeviceID parameters shall not change as long as the associatedCallingDevice remains in the call. This differs from the callingDeviceID and the calledDeviceID parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

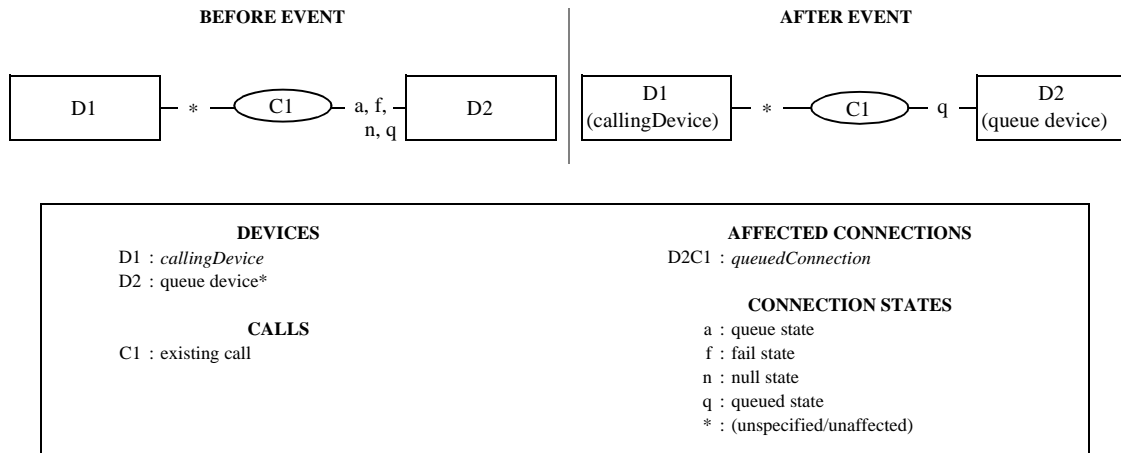
17.2.15 Queued

The Queued event indicates that a call has been queued.

Common situations that generate this event include:

- A call is queued at an ACD or hunt group device.
- A call is queued (camped on or parked, for example) at a device.

Figure 17-45 Queued Event



*There are some situations where D2 can be the calling device.

17.2.15.1 Event Parameters

Table 17-162 Queued—Event Parameters

Parameter Name	Type	M/ O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
queuedConnection	ConnectionID	M	Specifies the queued connection.
queue	SubjectDeviceID	M	Specifies the queued device.
callingDevice	CallingDeviceID	M	Specifies the calling device.
calledDevice	CalledDeviceID	M	Specifies the originally called device.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected from device.
numberQueued	Value	O	Specifies the total number of calls in queue, including this call.
callsInFront	Value	O	Specifies the number of calls ahead of the call when it was enqueued (this call is not counted in this number).
localConnectionInfo	LocalConnectionState	C	<p>Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call.</p> <ul style="list-style-type: none"> • For the queuing device: Queued • For the other devices left in the call: (unaffected) <p>This parameter is mandatory for events generated for device-type monitors.</p>
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.

Table 17-162 Queued—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
queuedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the queuedConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specified non-standardized information.

17.2.15.2 Event Causes

Table 17-163 Queued—Event Causes

Event Cause	Description	Associated Features
Busy	The call was queued at a called device because the device was active on another call and unable to accept this call.	Make Call, Consultation Call, Deflect Call, Single-Step Conference Call, Single-Step Transfer Call
Call Forward	The call was immediately queued to a device after it was forwarded.	Call Forwarding

Table 17-163 Queued—Event Causes (continued)

Event Cause	Description	Associated Features
Call Forward—Busy	The call was immediately queued to a device after it was forwarded because of a busy condition.	Call Forwarding
Call Forward—Immediate	The call was immediately queued to a device after it was forwarded (forwarding on all conditions).	Call Forwarding
Call Forward—No Answer	The call was immediately queued to a device after it was forwarded because of a no answer condition.	Call Forwarding
Camp On	The call was queued to a device because of the camp on feature.	Camp On Call
Camp On Trunks	The call was queued to a trunk because of the camp on feature.	External Call, Camp On Call
Distribution Delay	The distribution of the call was delayed at a distribution device for distribution device specific reasons (e.g. preconnect announcement).	ACD, queueing
Do Not Disturb	The call was queued at a device after it encountered a device with Do Not Disturb feature set.	Forwarding
Multiple Queueing	The call that is queued is also queued at other devices.	Multiple Queueing
Network Congestion	The call was queued because the call encountered a congested network.	External Call
Network Not Obtainable	The call could not reach a destination network.	External Call
Network Signal	The call was queued to a device outside of the switching sub-domain.	External Call
No Available Agents	The call was queued to a device because there were no available agents.	ACD
Normal	The call was queued to a device (a more specific cause cannot be provided).	Any feature
Overflow	The call was queued to a device after it overflowed a queue, group, or target.	ACD, queueing
Park	The call was queued because of the park feature.	Park Call
Recall	The call was queued to a device as part of the recall feature (e.g., due to a time-out associated with a feature that failed to complete).	Recall
Redirected	The call was queued to a device after it was diverted from or deflected to this device.	Deflect Call, Divert Call
Remains in Queue	The call was queued to a device while the calling device's position in the queue is retained. For example, the call could be queued to a VRU or other automated answering equipment while the calling device retains its position in the ACD queue.	ACD, Queueing, Single Step Conference, Join Call
Resources Not Available	The call was queued to a device because resources were not available.	ACD
Trunks Busy	The call was queued to a device because there is no available Network Interface Device (e.g., Trunk, CO line).	External Call

17.2.15.3 Functional Requirements

1. When observing a call or a device in a call, and the call diverts from a device in the call, if the switching function does not provide the Diverted event for all devices in a call or for call-type monitors (as indicated through the capabilities exchange services option), the computing function shall use the queue, calledDevice, lastRedirectionDevice, and cause (eventCause parameter type) parameters to properly track the progress of the call as a result of the redirection. See 6.7.6, “Tracking a Diverted Call”, on page 41 for more information.

2. For external incoming calls, the contents of the `networkCallingDeviceID` and the `networkCalledDeviceID` parameters shall not change as long as the `associatedCallingDevice` remains in the call. This differs from the `callingDeviceID` and the `calledDeviceID` parameters in that these parameters may change as a result of features such as transfer, forwarding, and conferencing, etc.

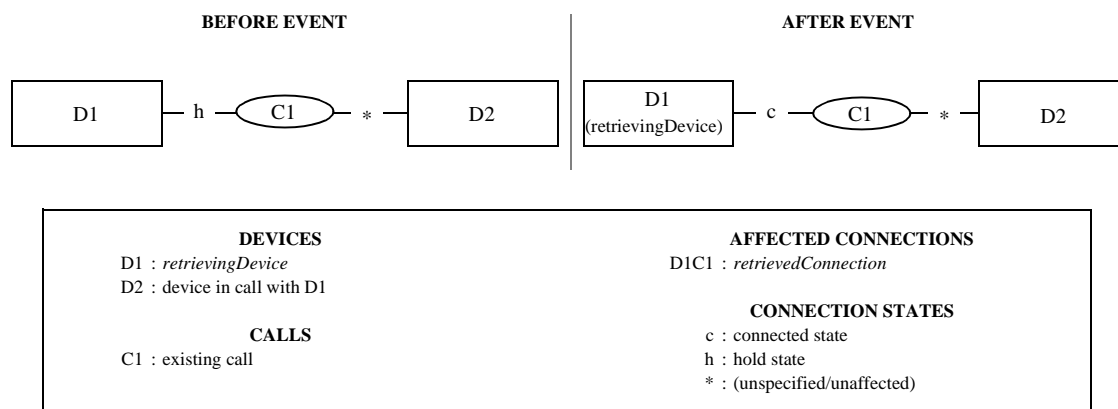
17.2.16 Retrieved

The Retrieved event indicates that a previously held call has been retrieved.

Common situations that generate this event include:

- When a held call is retrieved through the phone using features such as Retrieve, Alternate, etc.
- When a held call is retrieved during the successful execution of the Alternate Call, Reconnect Call, or the Retrieve Call service.

Figure 17-46 Retrieved Event



17.2.16.1 Event Parameters

Table 17-164 Retrieved—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
retrievedConnection	ConnectionID	M	Specifies the connection at which hold was deactivated.
retrievingDevice	SubjectDeviceID	M	Specifies the device at which hold was deactivated.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • For the retrieving device: Connected • For the other devices left in the call: (unaffected) This parameter is mandatory for events generated for device-type monitors.
correlatorData	CorrelatorData	O	Specifies the correlator data associated with the call.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.

Table 17-164 Retrieved—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
retrievedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the retrievedConnection connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.16.2 Event Causes

Table 17-165 Retrieved—Event Causes

Event Cause	Description	Associated Features
Alternate	The call was retrieved at a device as part of the alternate feature.	Alternate Call
Network Signal	The call was retrieved at a device that is outside of the switching sub-domain.	External Call
Normal	The call was retrieved at a device (a more specific cause cannot be provided).	Any feature, Alternate Call, Reconnect Call, Retrieve Call
Recall	The call was retrieved at a device because a hold timeout expired.	Recall

17.2.16.3 Functional Requirements

1. This event is only generated for the Alternate feature when the device is alternating between a connection that is in the Connected state and a connection that is in the Hold state. Note that this event shall not be generated when the Alternate feature is used to answer an alerting call (the Established event is generated in this case). Refer to 17.1.2, “Alternate Call”, on page 118 for more information.

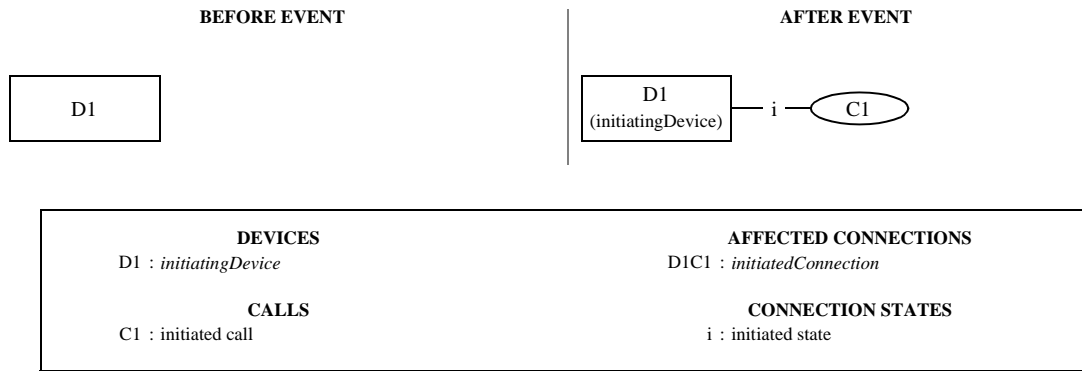
17.2.17 Service Initiated

The Service Initiated event indicates that a telephony service has been initiated at a monitored device. The switching function typically generates this event when “dial-tone” is being provided. This event indicates that either a call may be originated or a feature may be invoked. This event also may indicate that a device is prompting a user.

Common situations that generate this event include:

- When a service or feature prompts a user to take a phone off-hook and that phone is not able to do so without manual intervention.
- A Make Call or Consultation Call service has been invoked and the originating device is initiating a new call that is associated with the originating device.
- When manually invoking any feature at a device for an existing call that requires a new call to be created to input the feature.
- When a device is taken off-hook manually.
- When an incoming call arrives on a monitored Network Interface Device (e.g. trunk, CO line).

Figure 17-47 Service Initiated Event



17.2.17.1 Event Parameters

Table 17-166 Service Initiated—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
initiatedConnection	ConnectionID	M	Specifies the connection at which service was initiated.
initiatingDevice	SubjectDeviceID	M	Specifies the initiating device.
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> • When the device is initiating a service of some form: Initiated This parameter is mandatory for events generated for device-type monitors.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call.
cause	EventCause	M	Specifies the reason for the event.

Table 17-166 Service Initiated—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5, “CallCharacteristics”, on page 61 for the complete set of possible values.
initiatedConnectionInfo	ConnectionInformation	O	Specifies the connection information associated with the initiatedConnection connection. If this parameter is not present, then the connection information is switching function specific.
networkCallingDevice	NetworkCallingDeviceID	O	Specifies the original calling device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
networkCalledDevice	NetworkCalledDeviceID	O	Specifies the original called device information provided by the network for external incoming calls. This parameter is provided only for external incoming calls.
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

17.2.17.2 Event Causes

Table 17-167 Service Initiated—Event Causes

Event Cause	Description	Associated Features
Active Participation	The telephony service was initiated at a device and the device is being prompted to go off-hook.	Join Call with a participationType of Active (prompting)
Call Back	The telephony service was initiated at a device and the device is being prompted to go off-hook because of a previously set call back feature.	Call Back Call-Related, Call Back Non-Call-Related
Call Pickup	The telephony service was initiated at a device and the device is being prompted to go off-hook.	Directed Pickup, Group Pickup (prompting)
Conference	The telephony service was initiated at a device as part of a consultation with the intended purpose of establishing a conference.	Consultation Call with a consultOptions of ConferenceOnly
Consultation	The telephony service was initiated at a device as part of a consultation.	Consultation Call
Make Call	The telephony service was initiated at a device and the device is being prompted to go off-hook.	Make Call (prompting)
New Call	The telephony service was initiated for establishing a connection with another device.	Any feature, Make Call

Table 17-167 Service Initiated—Event Causes (continued)

Event Cause	Description	Associated Features
Normal	A more specific cause cannot be provided.	Any feature
Silent Participation	The telephony service was initiated at a device and the device is being prompted to go off-hook.	Join Call with a participationType of Silent (prompting)
Join Call	The telephony service was initiated at a device and the device is being prompted to go off-hook	Join Call (prompting)
Transfer	The telephony service was initiated at a device as part of a consultation for the purpose of transferring a call.	Consultation Call with a consultOptions of TransferOnly

17.2.17.3 Functional Requirements

1. Some CSTA services (Make Call, Call Back, Pickup, Join Call) may require to prompt the user of the targeted device in order to take that device off-hook. In this case a Service Initiated event is generated containing the appropriate cause code (Make Call, Call Back, Call Pickup, Join Call). The implementation of this prompting mechanism is switching function specific (display flashing, ring pattern, lamp blinking, etc.).
2. It is switching function dependent as to whether this event is provided to the computing function. For those switching functions that cannot detect prompting, or a device going off-hook and being provided dialtone, the first event seen by the computing function, for the calling device, will be the Originated event.
3. When prompting a call appearance, bridged, or hybrid device configuration as a result of a Make Call service request, only one of the appearances (initiatedConnection) will be prompted.
4. When prompting a call appearance, bridged, or hybrid device configuration as a result of a Call Back service request or Recall feature, only the appearance (initiatedConnection) that issued the Call Back service request or that was the Recall device, will be prompted.
5. For an external incoming call, this event is generated at the Network Interface Device (when this device can have monitoring applied to it).
6. It is not required to send the Service Initiated event for functional (e.g. en-bloc BRI) terminals nor is it required to be sent for calls that are set up without receiving dial tone or other prompting, like CSTA calls initiated with the Make Call service from a hands-free telephone.

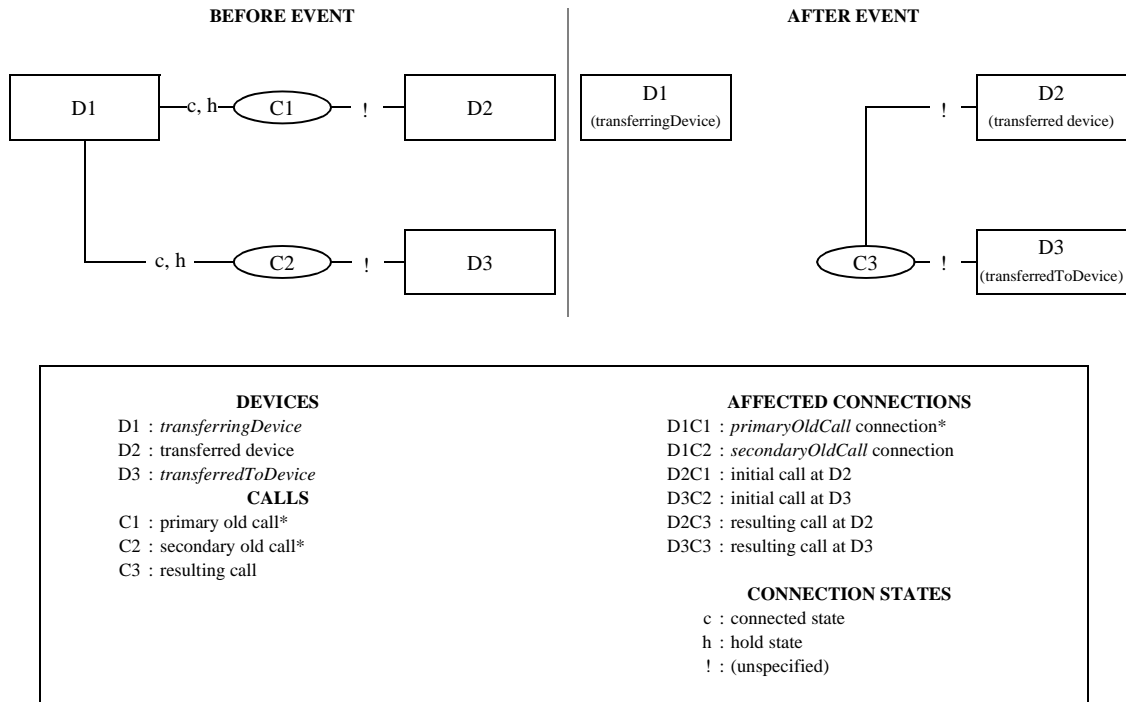
17.2.18 Transferred

The Transferred event indicates that an existing call has been transferred to another device and the transferring device has been dropped from the call. The transferring device does not appear in any future events for the call.

Common situations that generate this event include:

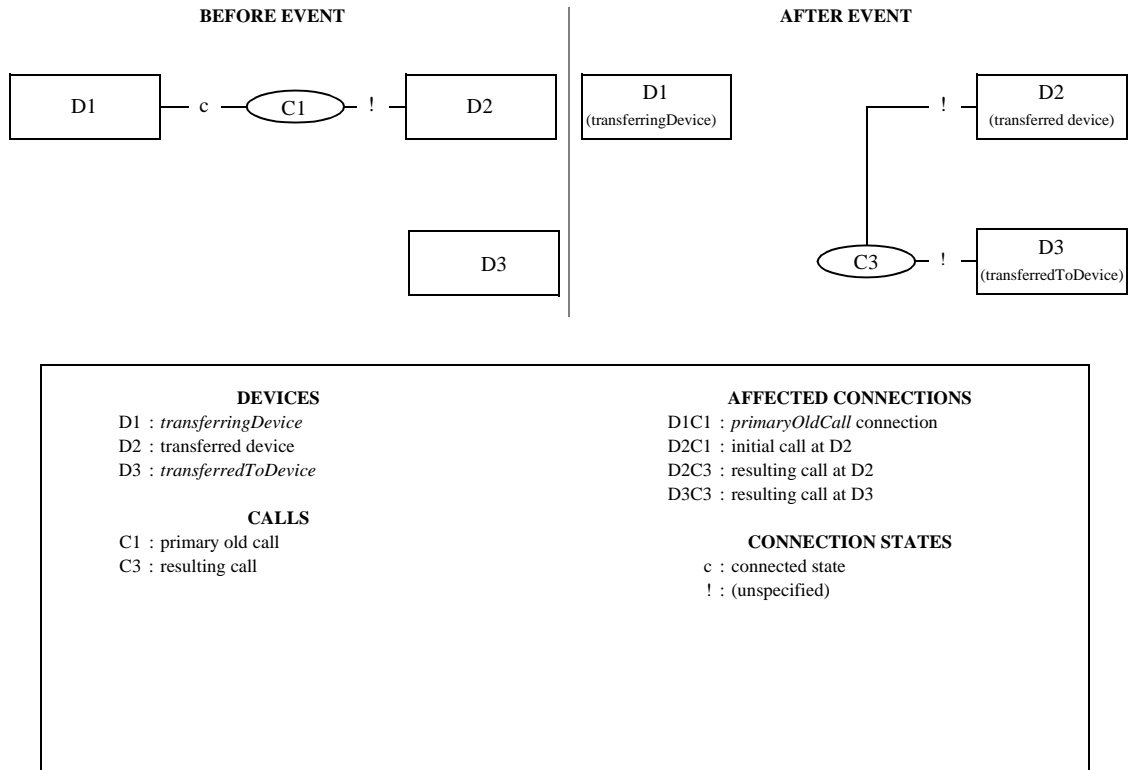
- Two step transferring situations (manual and service initiated).
- Single step transferring situations (manual and service initiated).

Figure 17-48 Transferred Event (Case A: Two Step Transfer)



* the primary/secondary old call and the primary/secondary old call connections mentioned in this figure are from the perspective of the transferringDevice (D1). See Functional Requirement #1.

Figure 17-49 Transferred Event (Case B: Single Step Transfer)



17.2.18.1 Event Parameters

Table 17-168 Transferred—Event Parameters

Parameter Name	Type	M/O/C	Description
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
primaryOldCall	ConnectionID	M	Specifies the connection of the primary call. See Functional Requirement #1.
secondaryOldCall	ConnectionID	C	Specifies the connection of the secondary call. If the switching function supports the “fixed-view” option (as indicated by the capability exchange services), this parameter is mandatory. If the switching function supports the “local-view” option, this parameter is mandatory if there are two known calls involved with the transfer (before the transfer is created) from the perspective of the monitored device, otherwise it shall not be provided. See Functional Requirement #1.
transferringDevice	SubjectDeviceID ¹	M	Specifies the device that transferred the call.
transferredToDevice	SubjectDeviceID ¹	M	Specifies the transferred to device.
transferredConnections	ConnectionList	M	Specifies information on each device/ConnectionID in the resulting transferred call that are known by the switching function.

Table 17-168 Transferred—Event Parameters (continued)

Parameter Name	Type	M/O/C	Description
localConnectionInfo	LocalConnectionState	C	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. The following are the states of the devices in the call. <ul style="list-style-type: none"> For the transferring device (any Connection IDs associated with the transfer, i.e., this event should be used for both single and multi-step transfers.): Null For the other devices associated with the transfer: (unaffected) This parameter is mandatory for events generated for device-type monitors.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call and shall otherwise not be provided.
userData	UserData	C	Specifies user information that is related to the call. This parameter is mandatory if user data is sent and the parameter is supported, otherwise it shall not be included.
chargingInfo	ChargingInfo	O	Specifies a total value of charging or currency units for the device that transferred the call.
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection. This parameter is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services), otherwise this parameter is optional.
mediaCallCharacteristics	MediaCallCharacteristics	O	Specifies the media class and media characteristics of the call. See 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete description.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5 for the complete set of possible values.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

- Note that SubjectDeviceID refers to a parameter type—not the subject device of the Transferred event. This parameter type is used to represent the two devices in this event because the two devices are affected by the generation of this event (i.e., the transferring device and the transferred device). However, there is only one subject device of the event and that is the transferring device. For more details on the SubjectDeviceID parameter type, see 12.3.24, “SubjectDeviceID”, on page 79.

17.2.18.2 Event Causes

Table 17-169 Transferred—Event Causes

Event Cause	Description	Associated Features
Network Signal	The call was involved in a transfer located outside of the switching sub-domain.	External Call
Normal	The call was transferred (a more specific event cause cannot be provided).	Transfer
Single Step Transfer	The call was transferred because of a single step transfer.	Single Step Transfer Call
Transfer	The call was transferred because of a two step transfer.	Transfer Call, Two Step Transfer

17.2.18.3 Functional Requirements

1. The contents of the `primaryOldCall` and the `secondaryOldCall` parameters may contain either a “fixed view” or a “local view” of the connections at a device before the transfer has been completed. The switching function indicates which view it provides via the `connectionView` parameter in the capability exchange services.
 - fixed view - for each transferred event generated by monitors placed on different devices in a call, the switching function provides the same information in the `primaryOldCall` and the `secondaryOldCall` parameters independent of the `monitorType` (call or device-type monitor) and independent of the role of the device in the conference (`conferencingDevice`, `addedParty`, etc.). The meaning of these parameters for the fixed-view are:
 - `primaryOldCall` - specifies the first call visible at the `transferringDevice`.
 - `secondaryOldCall` - specifies the second call visible at the `transferringDevice`.
 - local view - for each transferred event generated by monitors placed on different devices in a call, the switching function provides different information in the `primaryOldCall` and the `secondaryOldCall` parameters that depends upon which call was made visible first, from the perspective of the monitored device. The meaning of these parameters for the local-view are:
 - `primaryOldCall` - specifies the first call visible at the monitored device. For example, for a device-type monitor placed on the `transferringDevice` (two step transfer), this is the first call placed on hold (C1). For a device-type monitor placed on the `transferredTo` device, this is the first (and only) call involved in call C2 from the perspective of the monitored device.
 - `secondaryOldCall` - specifies the second call visible at the monitored device. For example, for a device-type monitor placed on the `transferringDevice` (two step transfer), this is call C2. For a monitor placed on the `transferredTo` device, there is no `secondaryOldCall` parameter.
2. The `transferredConnections` parameter is a list that contains the new `ConnectionID`, old `ConnectionID`, `DeviceID` (values such as ANI, etc.), and for externally located devices the associated `Network Interface DeviceID`. Refer to 12.2.8, “`ConnectionList`”, on page 62, for a description of which components are optional and mandatory.
3. The computing function should never assume the reuse of callIDs, although some switching functions may reuse one or the other.

18 Call Associated Features

This section describes the Call Associated Feature services and events (non-Call Control related) including descriptions of:

- Call Associated Feature services
- Call Associated Feature events

The services in this section assume that there is an existing call.

Refer to 6.1.4, “Call”, on page 21 and 6.1.5, “Connection”, on page 24 for information on the identifiers used to reference calls and connections.

18.1 Services

Table 18-1 Call Associated Feature Services Summary

Call Associated Feature Service	Description	Pg.
18.1.1 Associate Data (to be defined)	Associates correlator data, account code, and/or authorisation code information with a specified call.	256
18.1.2 Cancel Telephony Tones (to be defined)	Cancels a telephony tone that is being generated on a specified connection.	256
18.1.3 Generate Digits (to be defined)	Generates DTMF tones or rotary pulses on behalf of a connection in a call.	256
18.1.4 Generate Telephony Tones (to be defined)	Generates a specified telephony tone on behalf of a connection in a call.	256
18.1.5 Send User Information (to be defined)	Sends user-to-user information from the specified connection to the other device in the call.	256
18.1.6 Start DTMF Digits Collection (to be defined)	Supports allocation of resources for detecting DTMF digits.	256
18.1.7 Start Telephony Tones Collection (to be defined)	Supports allocation of resources for detecting telephony tones.	256
18.1.8 Stop DTMF Digits Collection (to be defined)	Releases the resources allocated with the Start DTMF Digits Collection service.	256
18.1.9 Stop Telephony Tones Collection (to be defined)	Releases the resources allocated with the Start Telephony Tones Collection service.	256

18.1.1 Associate Data (to be defined)	C → S
18.1.2 Cancel Telephony Tones (to be defined)	C → S
18.1.3 Generate Digits (to be defined)	C → S
18.1.4 Generate Telephony Tones (to be defined)	C → S
18.1.5 Send User Information (to be defined)	C → S
18.1.6 Start DTMF Digits Collection (to be defined)	C → S
18.1.7 Start Telephony Tones Collection (to be defined)	C → S
18.1.8 Stop DTMF Digits Collection (to be defined)	C → S
18.1.9 Stop Telephony Tones Collection (to be defined)	C → S

18.2 Events

Table 18-2 Call Associated Feature Events Summary

Call Associated Feature Event	Description	Pg.
18.2.1 Call Information (to be defined)	Indicates that call associated information has been collected for a call.	257
18.2.2 Charging (to be defined)	Indicates that new charging information has arrived for a device involved in a call.	257
18.2.3 DTMF Digits Detected (to be defined)	Indicates that DTMF digits have been detected or that collection of DTMF digits has stopped.	257
18.2.4 Telephony Tones Detected (to be defined)	Indicates that telephony tones have been detected or that collection of telephony tones has stopped.	257
18.2.5 Service Completion Failure (to be defined)	Indicates that a previous multi-step computing function initiated service request has failed before that service's successful completion criteria was satisfied.	257

- 18.2.1 Call Information (to be defined)**
- 18.2.2 Charging (to be defined)**
- 18.2.3 DTMF Digits Detected (to be defined)**
- 18.2.4 Telephony Tones Detected (to be defined)**
- 18.2.5 Service Completion Failure (to be defined)**

19 Media Attachment Services & Events

This clause describes services and events which enable access to the media stream of a call through attachment and detachment of media services to the call.

19.1 Services

Table 19-1 Media Attachment Services Summary

Media Attachment Service	Description	Pg.
19.1.1 Attach Media Service (to be defined)	Attaches a media service to a call.	258
19.1.2 Detach Media Service (to be defined)	Detaches a media service from a call.	258

19.1.1 Attach Media Service (to be defined)

C → S

19.1.2 Detach Media Service (to be defined)

C → S

19.2 Events

Table 19-2 Media Attachment Events Summary

Media Attachment Event	Description	Pg.
19.2.1 Media Attached (to be defined)	Indicates that a media service has been attached to a call.	258
19.2.2 Media Detached (to be defined)	Indicates that a media service has been detached from a call.	258

19.2.1 Media Attached (to be defined)

19.2.2 Media Detached (to be defined)

20 Routeing Services

This section specifies two types of Routeing services:

- Route Registration services
- Call Routeing services

NOTE

This clause describes Routeing services between the Switching Function and the Computing Function.

20.1 Registration Services

Table 20-1 Route Registration Services Summary

Route Registration Service	Description	Pg.
20.1.1 Route Register	Registers the computing function as a routeing server for a specified routeing device or for the entire switching function.	260
20.1.2 Route Register Abort	Specifies that the switching function has terminated a routeing server registration.	262
20.1.3 Route Register Cancel	Unregisters the computing function as a routeing server.	263

20.1.1 Route Register

C → S

The Route Register service is used to register the computing function as a routing server for a specific routing device or as a routing server for all routing devices within the switching sub-domain. The computing function may be required to register for routing services before it can receive any route requests for a routing device from the switching function. A computing function may register to be the routing server for more than one routing device.

20.1.1.1 Service Request

Table 20-2 Route Register—Service Request

Parameter Name	Type	M/O/C	Description
routingDevice	DeviceID	C	Specifies the routing device for which the computing function requests to be the routing server. This parameter is mandatory if the switching function does not support the option of registering for all routing devices in the switching sub-domain. Otherwise, the parameter is optional and if not present, indicates the registration is to be for all routing devices in the switching sub-domain. See Functional Requirement #2.
requestedRoutingMediaClass	Bitmap	O	Specifies the media classes of calls that are being requested to be routed. Refer to the mediaClass component in 12.2.4, “MediaCallCharacteristics”, on page 60 for the complete set of possible values. Note that multiple bits may be set. If this parameter is not provided (or if the parameter is not supported by the switching function), it is switching function dependent which types of media calls are routed.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.1.1.2 Service Response

This service follows the atomic acknowledgement model for this service request.

20.1.1.2.1 Positive Acknowledgement

Table 20-3 Route Register—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
routeRegisterReqID	RouteRegisterReqID	M	Specifies the routing registration request identifier for this registration.
actualRouteingMediaClass	Bitmap	C	This parameter specifies the actual media classes of calls that are routed by the switching function for routing registration. The actual media classes of calls routed may be the same or a subset of what was requested on the service request. If this parameter is supported by the switching function, it may be omitted if the requested and actual routing media class parameters are the same otherwise it shall be provided. If the parameter is not supported by the switching function, then the switching function does not filter media classes for calls for specific routing registrations. The capability exchange services indicates the media classes of calls that can be routed.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.1.1.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.1.1.3 Operational Model

20.1.1.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.1.1.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.1.1.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.1.1.3.4 Functional Requirements

1. The routeRegisterReqID parameter returned in the positive acknowledgement is used to identify the registration over which routing requests will be sent. All routing dialogues for the routing device occur over this routing registration. The routeRegisterReqID is also used when cancelling the routing registration.
2. If the routingDevice parameter on the service request is not provided and this option is supported, then the registration request is for all routing devices within the switching sub-domain. If the switching function sends a positive acknowledgement to this request, the routing server will receive route requests generated for all routing devices within the switching sub-domain. Some switching function implementations may not support the capability to register for all routing devices with a single Route Register request (i.e., with the routingDevice parameter not provided), in which case the switching function will send a negative acknowledgement to the Route Register request. The capabilities exchange services can be used by the computing function to determine if registering for all routing devices within the switching sub-domain is supported.
3. The number of simultaneous registrations allowed for the same routing device is switching function dependent. Some switching functions may limit this number to one, in which case only one registration per routing device is allowed. When the limit is reached, subsequent route register requests for the same routing device will result in negative acknowledgements from the switching function.

20.1.2 Route Register Abort

S → C

This service is used by the switching function to asynchronously cancel an active routeing registration. This service invalidates a current routeing registration. There is no positive acknowledgement defined for this service.

20.1.2.1 Service Request

Table 20-4 Route Register Abort—Service Request

Parameter Name	Type	M/O/C	Description
routeRegisterReqID	RouteRegisterReqID	M	Specifies the routeing registration request identifier for the routeing registration that was aborted.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.1.2.2 Service Response

There are no service completion conditions for this service.

20.1.2.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service.

20.1.2.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.1.2.3 Operational Model

20.1.2.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.1.2.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.1.2.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.1.2.3.4 Functional Requirements

1. The switching function may issue this service at any time when it can no longer maintain the routeing registration (e.g., when the associated routeing device goes out of service).

20.1.3 Route Register Cancel

C → S

The Route Register Cancel service is used to cancel a previous route registration. This request terminates the routing registration and the computing function receives no further routing requests for that routing registration once it receives the positive acknowledgement to the Route Register Cancel request.

20.1.3.1 Service Request

Table 20-5 Route Register Cancel—Service Request

Parameter Name	Type	M/O/C	Description
routeRegisterReqID	RouteRegisterReqID	M	Specifies the routing registration request identifier for which the routing registration is to be cancelled.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.1.3.2 Service Response

This service follows the atomic acknowledgement model for this service request.

20.1.3.2.1 Positive Acknowledgement

Table 20-6 Route Register Cancel—Positive Acknowledgement

Parameter Name	Type	M/O/C	Description
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.1.3.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.1.3.3 Operational Model

20.1.3.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.1.3.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.1.3.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.1.3.3.4 Functional Requirements

1. The computing function shall continue to process outstanding routing requests from the routing device until it receives a positive acknowledgement for the Route Register Cancel service request. The switching function will not send any further route requests for a registration once it has sent the positive acknowledgement.

20.2 Services

Table 20-7 Call Routeing Services Summary

Routeing Services	Description	Pg.
20.2.1 Re-Route	This service requests an alternate destination from the one provided by a previous Route Select service and based on previous information provided for the call.	265
20.2.2 Route End	This service ends a routeing dialogue.	266
20.2.3 Route Reject	This service is sent to the switching function during a routeing dialogue to indicate that a call should be returned to the network for alternate routeing.	268
20.2.4 Route Request	This service requests that the computing function provides a destination for a call. To aid in the selection of a destination, the service request includes the current destination and may include additional information.	270
20.2.5 Route Select	This service is used by the computing function to provide the destination requested by a previous Route Request or Re-Route request.	272
20.2.6 Route Used	This service provides the actual destination for a call that has been routed using the Route Select service with its optional parameter that requests the route that was used.	274

20.2.1 Re-Route

S → C

The Re-Route service requests an alternate destination from the one provided by a previous Route Select service and based on previous information provided for the call.

20.2.1.1 Service Request

Table 20-8 Re-Route—Service Request

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	RouteingCrossRefID	M	Specifies the cross reference identifier associated with the routeing dialogue.
routeRegisterReqID	RouteRegisterReqID	C	Specifies the route register request identifier associated with the route registration for this routeing dialogue. See Functional Requirement #2.
replyTimeout	Value	O	Specifies the amount of time (in milliseconds) that the switching function will wait for a reply from the computing function, before it proceeds with default routeing for the call. If the parameter is not present or the value is 0, the amount of time is switching function specific.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call (and if the parameter is supported).
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.2.1.2 Service Response

There are no service completion conditions for this service.

20.2.1.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service.

20.2.1.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.2.1.3 Operational Model

20.2.1.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.2.1.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.1.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.1.3.4 Functional Requirements

1. The requested route is sent from the computing function to the switching function by the Route Select service. The switching function shall use the routeingCrossRefID and routeRegisterReqID (if supported) parameters from the initial Route Request service to link this service to the others that are used to provide a route.
2. The routeRegisterReqID parameter is mandatory if the switching function supports route registration, and shall not be provided otherwise.
3. If the number of remaining retries (remainRetries parameter) is not provided by the computing function in the Route Select service, the computing function should be prepared to respond to all Re-Route service requests or terminate the routeing dialogue by using the Route End service when it cannot provide additional destinations.

20.2.2 Route End

C ↔ S

The Route End service ends a routing dialogue. This service is bi-directional (i.e., it may be invoked by the switching function or the computing function).

The computing function can use the Route End service when it cannot provide a route for a call. Typically, this can occur if:

- The computing function receives a valid routing request for a call without sufficient call information and it cannot determine a routing destination.
- The computing function has already provided all available destinations for a call and no more alternate destinations are available.
- The computing function does not have access to a database necessary to route the call.

In these cases, the computing function uses the Route End service to inform the switching function that it cannot provide a route for the call in question. The Route End service request will terminate the routing dialogue (routingCrossRefID) for the call. The Route End request does not clear the call. The switching function will continue to process the call using whatever default routing algorithm is available (i.e., in a switching function specific way).

The switching function uses the Route End service when it ends a routing dialogue. Typically, this can occur if:

- The call associated with the routing cross reference identifier has been successfully routed. This may occur when the computing function has sent a Route Select service request and the switching function has successfully routed the call.
- The calling party has abandoned a call associated with the routing cross reference identifier.
- The switching function timeout for a route request response has expired. This may occur if the computing function did not respond to a Route Request or Re-Route Request service within a switching function defined period.
- The switching function has ended a routing dialogue due to internal resource (or other) problems.

20.2.2.1 Service Request

Table 20-9 Route End—Service Request

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	RoutingCrossRefID	M	Specifies the cross reference identifier associated with the routing dialogue.
routeRegisterReqID	RouteRegisterReqID	C	Specifies the route register request identifier associated with the route registration for this routing dialogue. See Functional Requirement #3.
errorValue	ErrorValue	O	Specifies the reason for the route end request (see 12.2.11, “ErrorValue”, on page 65 for more information on this parameter).
correlatorData	CorrelatorData	C	Specifies correlator data. See Functional Requirement #4.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.2.2.2 Service Response

There are no service completion conditions for this service.

20.2.2.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service.

20.2.2.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, "ErrorValue", on page 65.

20.2.2.3 Operational Model

20.2.2.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.2.2.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.2.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.2.3.4 Functional Requirements

1. The computing function can use this service to respond to either a Route Request or a Re-Route service request.
2. The routeingCrossRefID and routeRegisterReqID (if supported) parameters from the initial Route Request service shall be used to link this service to the others that are used to provide a route.
3. The routeRegisterReqID parameter is mandatory if the switching function supports route registration, and shall not be provided otherwise.
4. The correlatorData parameter meaning is dependent upon the direction of the service request.
 - if the Route End service request is sent from the Switching Function to the Computing Function then the parameter specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call (and if the parameter is supported).
 - if the Route End service request is sent from the Computing Function to the Switching Function then the parameter specifies the correlator data to associate with the call.

20.2.3 Route Reject

C → S

The Route Reject service request is sent to the switching function during a routeing dialogue to indicate that a call should be returned to the originating network (the network from where the call entered the switching sub-domain where the routeing request was issued from) for alternate routing.

20.2.3.1 Service Request

Table 20-10 Route Reject—Service Request

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	RouteingCrossRefID	M	Specifies the cross reference identifier associated with the routeing dialogue.
routeRegisterReqID	RouteRegisterReqID	C	Specifies the route register request identifier associated with the route registration for this routeing dialogue. See Functional Requirement #3.
rejectCause	Enumerated	O	Specifies the reason why call should be returned to the originating network for alternate routing. The complete set of possible values is: <ul style="list-style-type: none"> • BusyOverflow - all possible destinations for the call are busy or unavailable and there is no queueing mechanism (ACD, for example) available. • QueueTimeOverflow - all possible destinations for the call are busy or unavailable and the estimated holding time before an agent is able to answer the call is too long. • CapacityOverflow - all possible destinations are busy or unavailable and the call cannot be queued because the system is already at capacity. • CalanderOverflow - all possible calendar based destinations for the call are unavailable because of the time of the day or the day of the week. • UnknownOverflow - the computing function is unable to be more specific. See Functional Requirement #4.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.2.3.2 Service Response

There are no service completion conditions for this service.

20.2.3.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service.

20.2.3.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.2.3.3 Operational Model

20.2.3.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.2.3.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.3.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.3.3.4 Functional Requirements

1. The computing function should issue the Route Reject service only in response to a Route Request or Re-Route Request from the switching function.

2. The `routeingCrossRefID` and `routeRegisterReqID` (if supported) parameters from the initial Route Request service shall be used to link this service to the others that are used to provide a route.
3. The `routeRegisterReqID` parameter is mandatory if the switching function supports route registration, and shall not be provided otherwise.
4. The `rejectCause` parameter provides additional information why the computing function is requesting that the switching function return the call to the originating network (the network from where the call entered the switching sub-domain where the routeing request was issued from) for alternate routeing. This information can be used by the switching function and/or passed to the originating network via network signalling protocols.

20.2.4 Route Request

S → C

The Route Request service requests that the computing function provide a destination for a call. To aid in the selection of a destination, the service request includes the current destination and may include additional information.

20.2.4.1 Service Request

Table 20-11 Route Request—Service Request

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	RouteingCrossRefID	M	Specifies the cross reference identifier associated with the routeing dialogue.
routeRegisterReqID	RouteRegisterReqID	C	Specifies the route register request identifier associated with the route registration for this routeing dialogue. See Functional Requirement #3.
currentRoute	CalledDeviceID	M	Specifies the current destination of the call for which a route is requested.
callingDevice	CallingDeviceID	O	Specifies the originator of the call.
routeingDevice	SubjectDeviceID	O	Specifies the device that initiated the Route Request service.
routedCall	ConnectionID	C	Specifies the ConnectionID of the call. This parameter is mandatory if the route request is call related. If the request is not call-related, then this parameter shall not be provided.
routeSelAlgorithm	Enumerated	O	Specifies the type of routeing algorithm requested. The complete set of possible values is: <ul style="list-style-type: none"> • ACD • Emergency • Least Cost • Normal • User Defined
associatedCallingDevice	AssociatedCallingDeviceID	C	Specifies the Network Interface Device associated with the calling device if the call is an external incoming call. This parameter is mandatory for all external incoming calls and shall not be provided otherwise.
associatedCalledDevice	AssociatedCalledDeviceID	C	For outgoing external calls, this parameter specifies the Network Interface Device associated with the originally called device. For incoming external calls, this parameter specifies a device within the switching sub-domain associated with the originally called device. This parameter is mandatory for all external outgoing calls and it is optional for external incoming calls.
priority	Boolean	O	Specifies the call priority. This may affect the selection of alternative routes. The complete set of possible values is: <ul style="list-style-type: none"> • True - Priority call. • False - Non-priority call.
replyTimeout	Value	O	Specifies the amount of time (in milliseconds) that the switching function will wait for a reply from the computing function, before it proceeds with default routeing for the call. If the parameter is not present or the value is 0, the amount of time is switching function specific.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call (and if the parameter is supported).

Table 20-11 Route Request—Service Request (continued)

Parameter Name	Type	M/O/C	Description
mediaCallCharacteristics	MediaCallCharacteristics	O	This specifies the media class (voice, digital data, etc.) and characteristics of the call. If this parameter is not present (and the parameter is supported) then the call is a voice call.
callCharacteristics	CallCharacteristics	O	Specifies the high level characteristics (ACD call, Priority call, etc.) associated with the call. See 12.2.5 for complete the set of possible values.
routedCallInfo	ConnectionInformation	O	Specifies the connection information associated with the routedCall connection. If this parameter is not present, then the connection information is switching function specific.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.2.4.2 Service Response

There are no service completion conditions for this service.

20.2.4.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service.

20.2.4.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.2.4.3 Operational Model

20.2.4.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.2.4.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.4.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.4.3.4 Functional Requirements

1. The requested route is sent from the computing function to the switching function by the Route Select service.
2. The switching function generates the routeingCrossRefID to link this service to the others that are used to provide a route.
3. The routeRegisterReqID parameter is mandatory if the switching function supports route registration, and shall not be provided otherwise. If the routeRegisterReqID parameter is not present, then the routeingCrossRefID shall be unique across the entire switching sub-domain.

20.2.5 Route Select

C → S

The Route Select service is used by the computing function to provide the destination requested by a previous Route Request or Re-Route service.

20.2.5.1 Service Request

Table 20-12 Route Select—Service Request

Parameter Name	Type	M/O/C	Description
CrossRefIdentifier	RouteingCrossRefID	M	Specifies the cross reference identifier associated with the routeing dialogue.
routeRegisterReqID	RouteRegisterReqID	C	Specifies the route register request identifier associated with the route registration for this routeing dialogue. See Functional Requirement #3.
routeSelected	DeviceID	M	Specifies the primary selected destination of the call for which a route was requested.
alternateRoutes	List of DeviceIDs	O	Specifies the list of alternate destinations (in priority order) which are to be used sequentially for routeing the call if the primary selected destination initially (i.e., the routeSelected parameter) or a previous entry in the list is not valid or available.
remainRetries	Enumerated	O	Specifies either the number of alternative routes remaining or the reason why the computing function is not providing it. This parameter may be one of the following: <ul style="list-style-type: none"> noListAvailable - indicates that the computing does not maintain a fixed list of alternate routes. noCountAvailable - indicates that the computing function does not maintain or cannot provide a count of the remaining routes to try. retryCount (Integer) - indicates the actual number of alternative routes remaining. See Functional Requirement #4 for more information on this parameter.
routeUsedReq	Boolean	O	Specifies whether the switching function should issue the Route Used service when it has accepted a route. <ul style="list-style-type: none"> True - Route Used service should be issued. False - Route Used service should not be issued.
correlatorData	CorrelatorData	O	Specifies the correlator data to associate with the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.2.5.2 Service Response

There are no service completion conditions for this service.

20.2.5.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service.

20.2.5.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.2.5.3 Operational Model

20.2.5.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.2.5.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.5.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.5.3.4 Functional Requirements

1. The computing function shall issue a Route Select service request only in response to a Route request or Re-Route request from the switching function.
2. The routeingCrossRefID and routeRegisterReqID (if supported) parameters from the initial Route Request service shall be used to link this service to the others that are used to provide a route.
3. The routeRegisterReqID parameter is mandatory if the switching function supports route registration, and shall not be provided otherwise.
4. If the computing function is unable to supply an actual count (retryCount) of the number of alternative routes in the remainRetries parameter, it can provide one of two other options in the parameter:
 - noListAvailable. This indicates a specific reason why the retry count is not being provided - that the computing function does not maintain a fixed list of alternate routes. For example, the alternative routes may be calculated algorithmically or by progressive database searches (i.e. returns the next entry matched to the search criterion).
 - noCountAvailable. This indicates a more general reason why the retry count is not being provided - that the computing function does not maintain or cannot provide a count of the remaining routes to try. For example, the design of the database does not enable a count of the entries left to be provided or it does not maintain a fixed list of routes.

20.2.6 Route Used

S → C

The Route Used service provides the actual destination for a call that has been routed using the Route Select service with its optional parameter that requests the route that was used.

20.2.6.1 Service Request

Table 20-13 Route Used—Service Request

Parameter Name	Type	M/O/C	Description
crossRefIdentifier	RouteingCrossRefID	M	Specifies the cross reference identifier associated with the routeing dialogue.
routeRegisterReqID	RouteRegisterReqID	C	Specifies the route register request identifier associated with the route registration for this routeing dialogue. See Functional Requirement #4.
routeUsed	CalledDeviceID	M	Specifies the actual destination of the call for which a route was requested.
callingDevice	CallingDeviceID	O	Specifies the originator of the call.
domain	Boolean	O	Specifies whether the resolved destination is within the switching sub-domain. The complete set of possible values is: <ul style="list-style-type: none"> • True - Resolved destination is within the switching sub-domain. • False - The call has been routed outside the switching sub-domain.
correlatorData	CorrelatorData	C	Specifies the correlator data associated with the call. This parameter is mandatory if there is correlator data associated with the call (and if the parameter is supported).
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Specifies non-standardized information.

20.2.6.2 Service Response

There are no service completion conditions for this service.

20.2.6.2.1 Positive Acknowledgement

There is no positive acknowledgement associated with this service.

20.2.6.2.2 Negative Acknowledgement

The negative acknowledgement error codes are described in 12.2.11, “ErrorValue”, on page 65.

20.2.6.3 Operational Model

20.2.6.3.1 Connection State Transitions

There are no connection state changes due to this service.

20.2.6.3.2 Device-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.6.3.3 Call-Type Monitoring Event Sequences

There are no events generated as a result of this service.

20.2.6.3.4 Functional Requirements

1. This service is used to inform the computing function of the actual route that was used by the switching function. This route could be different than the route provided by the computing function with the Route Select service because the route could have been altered via interactions in the switching function with features such as Forwarding or ACD routeing, for example.
2. The Route Used service should be completed via a Route End service sent by either the computing or switching functions.
3. The routeingCrossRefID and routeRegisterReqID (if supported) parameters from the initial Route Request service shall be used to link this service to the others that are used to provide a route.

4. The routeRegisterReqID parameter is mandatory if the switching function supports route registration, and shall not be provided otherwise.

21 Physical Device Features

This section describes the feature capabilities as related to the device’s physical element including descriptions of:

- Physical Device Feature services
- Physical Device Feature events

General Functional Requirement

The device identifier supplied on each Physical Device Feature service shall be the device identifier associated with the device’s physical element. Otherwise, the request is rejected with a negative acknowledgement.

21.1 Services

Table 21-1 Physical Device Feature Services Summary

Physical Device Feature Service	Description	Pg.
21.1.1 Button Press (to be defined)	Simulates the activation of a specified button on a device.	277
21.1.2 Get Auditory Apparatus Information (to be defined)	Get information on one or all auditory apparatuses at a specified device.	277
21.1.3 Get Button Information (to be defined)	Get the button information for either a specified button or all buttons on a device.	277
21.1.4 Get Display (to be defined)	Get a snapshot of the contents of the physical device element’s display.	277
21.1.5 Get Hookswitch Status (to be defined)	Get the hookswitch status of a specified device.	277
21.1.6 Get Lamp Information (to be defined)	Get the lamp information for either a specified lamp or all lamps on a device.	277
21.1.7 Get Lamp Mode (to be defined)	Get the lamp mode status of a specified button on a device.	277
21.1.8 Get Message Waiting Indicator (to be defined)	Get the message waiting status at a specified device.	277
21.1.9 Get Microphone Gain (to be defined)	Get the microphone gain setting at a specified device.	277
21.1.10 Get Microphone Mute (to be defined)	Get the microphone mute status at a specified device.	277
21.1.11 Get Ringer Status (to be defined)	Get the ringer status (ringing/not ringing, ring count, ring pattern, ring volume) of one or all ringers associated with a specified physical device element.	277
21.1.12 Get Speaker Mute (to be defined)	Get the speaker status of a specified device.	277
21.1.13 Get Speaker Volume (to be defined)	Get the speaker volume setting of a specified device.	277
21.1.14 Set Button Information (to be defined)	Set the button information of a specified button on a device.	277
21.1.15 Set Display (to be defined)	Set the display on a specified physical device element.	277
21.1.16 Set Hookswitch Status (to be defined)	Set the hookswitch status of a specified device.	277
21.1.17 Set Lamp Mode (to be defined)	Set the lamp mode status of a specified button on a device.	277
21.1.18 Set Message Waiting Indicator (to be defined)	Set the message waiting status of a specified device.	277
21.1.19 Set Microphone Gain (to be defined)	Set the microphone gain setting of a specified device.	277
21.1.20 Set Microphone Mute (to be defined)	Set the microphone mute status of a specified device.	277
21.1.21 Set Ringer Status (to be defined)	Set the specified ringer to ring or not to ring. May also be used to set the ring pattern and ring volume of a ringer at a specified physical device element.	277

Table 21-1 Physical Device Feature Services Summary (continued)

Physical Device Feature Service	Description	Pg.
21.1.22 Set Speaker Mute (to be defined)	Set the speaker mute status of a specified device.	277
21.1.23 Set Speaker Volume (to be defined)	Set the speaker volume setting of a specified device.	277

21.1.1	Button Press (to be defined)	C → S
21.1.2	Get Auditory Apparatus Information (to be defined)	C → S
21.1.3	Get Button Information (to be defined)	C → S
21.1.4	Get Display (to be defined)	C → S
21.1.5	Get Hookswitch Status (to be defined)	C → S
21.1.6	Get Lamp Information (to be defined)	C → S
21.1.7	Get Lamp Mode (to be defined)	C → S
21.1.8	Get Message Waiting Indicator (to be defined)	C → S
21.1.9	Get Microphone Gain (to be defined)	C → S
21.1.10	Get Microphone Mute (to be defined)	C → S
21.1.11	Get Ringer Status (to be defined)	C → S
21.1.12	Get Speaker Mute (to be defined)	C → S
21.1.13	Get Speaker Volume (to be defined)	C → S
21.1.14	Set Button Information (to be defined)	C → S
21.1.15	Set Display (to be defined)	C → S
21.1.16	Set Hookswitch Status (to be defined)	C → S
21.1.17	Set Lamp Mode (to be defined)	C → S
21.1.18	Set Message Waiting Indicator (to be defined)	C → S
21.1.19	Set Microphone Gain (to be defined)	C → S
21.1.20	Set Microphone Mute (to be defined)	C → S
21.1.21	Set Ringer Status (to be defined)	C → S
21.1.22	Set Speaker Mute (to be defined)	C → S
21.1.23	Set Speaker Volume (to be defined)	C → S

21.2 Events

Table 21-2 Physical Device Feature Event Summary

Physical Device Feature Event	Description	Pg.
21.2.1 Button Information (to be defined)	The information associated with a button on a device has changed.	278
21.2.2 Button Press (to be defined)	A button has been pressed.	278
21.2.3 Display Updated (to be defined)	The contents of a physical device element's display has changed.	278
21.2.4 Hookswitch (to be defined)	A hookswitch status has changed.	278
21.2.5 Lamp Mode (to be defined)	The lamp mode status of a particular lamp has changed.	278
21.2.6 Message Waiting (to be defined)	The message waiting status has changed.	278
21.2.7 Microphone Gain (to be defined)	The microphone gain setting has changed for one of the hookswitches.	278
21.2.8 Microphone Mute (to be defined)	The microphone mute status has changed for one of the hookswitches.	278
21.2.9 Ringer Status (to be defined)	The ringer attribute associated with a physical element of a device has changed.	278
21.2.10 Speaker Mute (to be defined)	The speaker mute status has changed for one of the hookswitches.	278
21.2.11 Speaker Volume (to be defined)	The speaker volume setting has changed for one of the hookswitches.	278

21.2.1 Button Information (to be defined)	C → S
21.2.2 Button Press (to be defined)	C → S
21.2.3 Display Updated (to be defined)	C → S
21.2.4 Hookswitch (to be defined)	C → S
21.2.5 Lamp Mode (to be defined)	C → S
21.2.6 Message Waiting (to be defined)	C → S
21.2.7 Microphone Gain (to be defined)	C → S
21.2.8 Microphone Mute (to be defined)	C → S
21.2.9 Ringer Status (to be defined)	C → S
21.2.10 Speaker Mute (to be defined)	C → S
21.2.11 Speaker Volume (to be defined)	C → S

22 Logical Device Features

This clause describes the feature capabilities as related to a logical device. It including the specification of:

- Logical Device Feature services
- Logical Device Feature events

22.1 Services

Table 22-1 Logical Device Feature Services Summary

Logical Device Feature Service	Description	Pg.
22.1.1 Call Back Non-Call-Related (to be defined)	Requests that the switching function originate a call back call between two devices.	280
22.1.2 Call Back Message Non-Call-Related (to be defined)	Requests that the switching function leave a pre-defined message requesting that the called device call the calling device.	280
22.1.3 Cancel Call Back (to be defined)	Cancels a previous (or all) Call Back feature at a device.	280
22.1.4 Cancel Call Back Message (to be defined)	Cancels a previous (or all) Call Back Message feature at a device.	280
22.1.5 Get Agent Status (to be defined)	Get the agent status of a specified device.	280
22.1.6 Get Auto Answer (to be defined)	Get the auto-answer status of a specified device	280
22.1.7 Get Caller ID Status (to be defined)	Get the Caller ID status of a specified device.	280
22.1.8 Get Do Not Disturb (to be defined)	Get the do not disturb status of a specified device.	280
22.1.9 Get Forwarding (to be defined)	Get the forwarding status of a specified device.	280
22.1.10 Get Last Number Dialed (to be defined)	Get the last number dialed at a specified device.	280
22.1.11 Get Routeing Mode (to be defined)	Get the routeing mode at a specified device.	280
22.1.12 Set Agent Status (to be defined)	Set the agent state of a specified device.	280
22.1.13 Set Auto Answer (to be defined)	Set the auto-answer status of a specified device	280
22.1.14 Set Caller ID Status (to be defined)	Set the Caller ID Status at the specified device.	280
22.1.15 Set Do Not Disturb (to be defined)	Set the do not disturb status of a specified device.	280
22.1.16 Set Forwarding (to be defined)	Set the forwarding status of a specified device.	280
22.1.17 Set Routeing Mode (to be defined)	Set the routeing mode of a specified device.	280

22.1.1	Call Back Non-Call-Related (to be defined)	C → S
22.1.2	Call Back Message Non-Call-Related (to be defined)	C → S
22.1.3	Cancel Call Back (to be defined)	C → S
22.1.4	Cancel Call Back Message (to be defined)	C → S
22.1.5	Get Agent Status (to be defined)	C → S
22.1.6	Get Auto Answer (to be defined)	C → S
22.1.7	Get Caller ID Status (to be defined)	C → S
22.1.8	Get Do Not Disturb (to be defined)	C → S
22.1.9	Get Forwarding (to be defined)	C → S
22.1.10	Get Last Number Dialed (to be defined)	C → S
22.1.11	Get Routeing Mode (to be defined)	C → S
22.1.12	Set Agent Status (to be defined)	C → S
22.1.13	Set Auto Answer (to be defined)	C → S
22.1.14	Set Caller ID Status (to be defined)	C → S
22.1.15	Set Do Not Disturb (to be defined)	C → S
22.1.16	Set Forwarding (to be defined)	C → S
22.1.17	Set Routeing Mode (to be defined)	C → S

22.2 Events

Table 22-2 Logical Device Feature Event Summary

Logical Device Feature Event	Description	Pg.
22.2.1 Agent Busy (to be defined)	An agent is occupied with serving an ACD call.	281
22.2.2 Agent Logged Off (to be defined)	An agent has logged off of an ACD group.	281
22.2.3 Agent Logged On (to be defined)	An agent has logged on to an ACD group.	281
22.2.4 Agent Not Ready (to be defined)	An agent is unavailable and cannot receive incoming ACD calls.	281
22.2.5 Agent Ready (to be defined)	An agent is available for an ACD call.	281
22.2.6 Agent Working After Call (to be defined)	An agent is involved with after call work and cannot receive ACD calls.	281
22.2.7 Auto Answer (to be defined)	The auto-answer status has changed.	281
22.2.8 Call Back (to be defined)	The call back feature status has changed.	281
22.2.9 Call Back Message (to be defined)	The call back message status has changed.	281
22.2.10 Caller ID Status (to be defined)	The Caller ID status has been changed for a device.	281
22.2.11 Do Not Disturb (to be defined)	The do not disturb status has changed.	281
22.2.12 Forwarding (to be defined)	The forwarding status has changed.	281
22.2.13 Routeing Mode (to be defined)	The routeing mode status has changed.	281

- 22.2.1 Agent Busy (to be defined)**
- 22.2.2 Agent Logged Off (to be defined)** C → S
- 22.2.3 Agent Logged On (to be defined)** C → S
- 22.2.4 Agent Not Ready (to be defined)** C → S
- 22.2.5 Agent Ready (to be defined)** C → S
- 22.2.6 Agent Working After Call (to be defined)** C → S
- 22.2.7 Auto Answer (to be defined)** C → S
- 22.2.8 Call Back (to be defined)** C → S
- 22.2.9 Call Back Message (to be defined)** C → S
- 22.2.10 Caller ID Status (to be defined)** C → S
- 22.2.11 Do Not Disturb (to be defined)** C → S
- 22.2.12 Forwarding (to be defined)** C → S
- 22.2.13 Routeing Mode (to be defined)** C → S

23 Device Maintenance Events

23.1 Events

Table 23-1 Device Maintenance Events Summary

Device Maintenance Event	Description	Pg.
23.1.1 Back In Service (to be defined)	Indicates that the device has returned into service and once again operates normally in the switching function	282
23.1.2 Device Capabilities Changed (to be defined)	Indicates that the device level information has changed.	282
23.1.3 Out Of Service (to be defined)	Indicates that the device has entered a maintenance state (i.e., has been taken out of service) and can no longer accept calls or service requests (call control, call associated, physical, and logical)	282

- 23.1.1 **Back In Service (to be defined)** C → S
- 23.1.2 **Device Capabilities Changed (to be defined)** C → S
- 23.1.3 **Out Of Service (to be defined)** C → S

24 I/O Services

This section defines two types of I/O services:

- I/O Registration services
- I/O services

24.1 Registration Services

Table 24-1 I/O Registration Services Summary

I/O Registration Service	Description	Pg.
24.1.1 I/O Register (to be defined)	Registers the computing function as an I/O server for a specified device or for the entire switching function.	283
24.1.2 I/O Register Abort (to be defined)	Specifies that the switching function has terminated an I/O server registration.	283
24.1.3 I/O Register Cancel (to be defined)	Unregisters the computing function as an I/O server.	283

24.1.1 I/O Register (to be defined)

C → S

24.1.2 I/O Register Abort (to be defined)

S → C

24.1.3 I/O Register Cancel (to be defined)

C → S

24.2 I/O Services

Table 24-2 I/O Services Summary

I/O Registration Service	Description	Pg.
24.2.1 Data Path Resumed (to be defined)	The Data Path Resumed service provides information that a previously suspended Data Path has been resumed.	284
24.2.2 Data Path Suspended (to be defined)	The Data Path Suspended service provides information that a Data Path has been suspended.	284
24.2.3 Fast Data (to be defined)	The Fast Data service starts a Data Path for only the duration of sending one data message.	284
24.2.4 Resume Data Path (to be defined)	The Resume Data Path requests the Switching Function to resume a currently suspended Data Path.	284
24.2.5 Send Broadcast Data (to be defined)	The Send Broadcast Data service writes to all open Data Paths for a given application association and Data Path type.	284
24.2.6 Send Data (to be defined)	The Send Data service writes data to a specified Data Path.	284
24.2.7 Send Multicast Data (to be defined)	The Send Multicast Data service writes to multiple Data Paths.	284
24.2.8 Start Data Path (to be defined)	The Start Data Path service starts a Data Path on the specified object.	284
24.2.9 Stop Data Path (to be defined)	The Stop Data Path service terminates an existing Data Path.	284
24.2.10 Suspend Data Path (to be defined)	The Suspend Data Path suspends a specified Data Path but does not destroy the Data Path.	284

- 24.2.1 **Data Path Resumed (to be defined)** S → C
- 24.2.2 **Data Path Suspended (to be defined)** S → C
- 24.2.3 **Fast Data (to be defined)** C ↔ S
- 24.2.4 **Resume Data Path (to be defined)** C → S
- 24.2.5 **Send Broadcast Data (to be defined)** C → S
- 24.2.6 **Send Data (to be defined)** S → C
- 24.2.7 **Send Multicast Data (to be defined)** C → S
- 24.2.8 **Start Data Path (to be defined)** C → S
- 24.2.9 **Stop Data Path (to be defined)** S → C
- 24.2.10 **Suspend Data Path (to be defined)** S ↔ C

25 Voice Unit Services & Events

This section specifies Voice Unit services and events.

25.1 Services

Table 25-1 Voice Services Summary

Voice Service	Description	Pg.
25.1.1 Concatenate Message (to be defined)	The Concatenate Message service combines multiple messages, in the sequence provided, into a single resulting message.	285
25.1.2 Delete Message (to be defined)	The Delete Message service deletes a specified voice message.	285
25.1.3 Play Message (to be defined)	The Play Message service causes the Voice Unit to start playing a voice message on a particular Connection.	285
25.1.4 Query Voice Attribute (to be defined)	The Query Voice Attribute service accesses and reports the current value of a specified voice attribute for a specified Message.	285
25.1.5 Record Message (to be defined)	The Record Message service starts recording a voice message from a specified connection.	285
25.1.6 Reposition (to be defined)	The Reposition service moves the current position pointer forward or backward a specified number of milliseconds in a message.	285
25.1.7 Resume (to be defined)	The Resume service restarts the playing or recording of a previously suspended message at the current position.	285
25.1.8 Review (to be defined)	The Review service plays a portion of a voice message during a recording session.	285
25.1.9 Set Voice Attribute (to be defined)	The Set Voice Attribute service sets a voice attribute for a specified connection and message.	285
25.1.10 Stop (to be defined)	The Stop service stops playing or recording of a message and resets the position pointer to the beginning of the message.	285
25.1.11 Suspend (to be defined)	The Suspend service temporarily stops the playing or recording of the current message and sets the current position in the message for later use.	285
25.1.12 Synthesize Message (to be defined)	The Synthesize Message service constructs a voice message from a text message.	285

25.1.1 Concatenate Message (to be defined)	C → S
25.1.2 Delete Message (to be defined)	C → S
25.1.3 Play Message (to be defined)	C → S
25.1.4 Query Voice Attribute (to be defined)	C → S
25.1.5 Record Message (to be defined)	C → S
25.1.6 Reposition (to be defined)	C → S
25.1.7 Resume (to be defined)	C → S
25.1.8 Review (to be defined)	C → S
25.1.9 Set Voice Attribute (to be defined)	C → S
25.1.10 Stop (to be defined)	C → S
25.1.11 Suspend (to be defined)	C → S
25.1.12 Synthesize Message (to be defined)	C → S

25.2 Events

Table 25-2 Voice Events Summary

Call Control Event	Description	Pg.
25.2.1 Play (to be defined)	The Play event indicates that a message is being played.	286
25.2.2 Record (to be defined)	The Record event indicates that a message is being recorded.	286
25.2.3 Review (to be defined)	The Review event indicates that a message is being reviewed.	286
25.2.4 Stop (to be defined)	The Stop event indicates that a message is not being used for play or record.	286
25.2.5 Suspend Play (to be defined)	The Suspend Play event indicates that a message is suspended in play.	286
25.2.6 Suspend Record (to be defined)	The Suspend Record event indicates that a message is suspended during record.	286
25.2.7 Voice Attribute Changed (to be defined)	The Voice Attribute changed event indicates that one or more attributes of a message has changed.	286

- 25.2.1 Play (to be defined)**
- 25.2.2 Record (to be defined)**
- 25.2.3 Review (to be defined)**
- 25.2.4 Stop (to be defined)**
- 25.2.5 Suspend Play (to be defined)**
- 25.2.6 Suspend Record (to be defined)**
- 25.2.7 Voice Attribute Changed (to be defined)**

26 Call Detail Record (CDR) Services

This section specifies the Call Detail Record (CDR) services.

26.1 Services

Table 26-1 CDR Services Summary

Voice Service	Description	Pg.
26.1.1 Get Call Detail Records (to be defined)	The Get Call Detail Records Service, requested by the switch, provides notification that a "Send Stored Call Detail Records" Service request shall be sent from the computer to the switch to initiate the en-bloc transfer of Call Detail Recording Reports.	287
26.1.2 Call Detail Recording Report (to be defined)	The Call Detail Recording Report service is used to provide information about call details and is sent from the server (switch) to the client (computer).	287
26.1.3 Send Stored Call Detail Records (to be defined)	The Send Stored Call Detail Records service initiates the en-bloc transfer of Call Detail Recording Reports from the server (switch) to the client (computer) for calls recorded at the end of each call.	287
26.1.4 Start Call Detail Records Transmission (to be defined)	The Start Call Detail Records Transmission Service starts the transmission of Call Detail Records.	287
26.1.5 Stop Call Detail Records Transmission (to be defined)	The Stop Call Detail Records Transmission Service is used to cancel a previously initiated Start Call Transmission service.	285

26.1.1 Get Call Detail Records (to be defined)

26.1.2 Call Detail Recording Report (to be defined)

26.1.3 Send Stored Call Detail Records (to be defined)

26.1.4 Start Call Detail Records Transmission (to be defined)

26.1.5 Stop Call Detail Records Transmission (to be defined)

27 Vendor Specific Extensions Services & Events

27.1 Registration Services

Table 27-1 Escape Registration Services Summary

Escape Registration Service	Description	Pg.
27.1.1 Escape Register (to be defined)	Register the computing function for escape services.	288
27.1.2 Escape Register Abort (to be defined)	Indicates the switching function has terminated an escape service registration.	288
27.1.3 Escape Register Cancel (to be defined)	Unregisters the computing function for escape services.	288

27.1.1 **Escape Register (to be defined)** C → S

27.1.2 **Escape Register Abort (to be defined)** S → C

27.1.3 **Escape Register Cancel (to be defined)** C → S

27.2 Services

Table 27-2 Escape Services Summary

Escape Service	Description	Pg.
27.2.1 Escape (to be defined)	Request for vendor-specific extended service (i.e., enhanced service in addition to those defined in this specification).	288
27.2.2 Private Data Version (to be defined)	Provides the switching function with the negotiated version for Private Data.	288

27.2.1 **Escape (to be defined)** C ↔ S

27.2.2 **Private Data Version (to be defined)** C → S

27.3 Events

Table 27-3 Private Events Summary

Private Event	Description	Pg.
27.3.1 Private Event (to be defined)	Unsolicited vendor-specific extended information event. The type of monitors (i.e., device-type or call-type) this event is reported on is switching function implementation specific.	288

27.3.1 **Private Event (to be defined)** S → C

Annex A
(informative)

Connection State Transition Examples

This annex contains examples that describes the connection state transitions specified in 6.1.5, “Connection”.

Table A-1 Connection State Transition Call Flow Examples


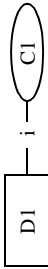
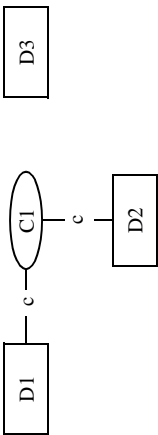
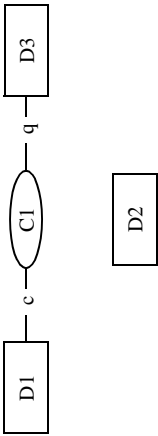


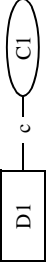

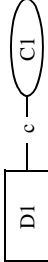

State Change	Action	Initial State	Final State	Comments
(D1) Null -> Initiated	A device goes off-hook or the computing function issues a Make Call service and the calling device is prompted.			
(D3) Null -> Queued	Device D2 parks the call to another device (D3).			
(D1) Null -> Connected	The computing function issues a Make Call service request. The switching function sends the Originated event for the calling device.			The calling device is an Auto Answer device therefore no Service Initiated event is seen prior to the Originated event.
(D2) Null -> Fail	Device D1 dials Device D2. Device D2 is currently active in another call and can not accept D1's call. D1 hears a busy tone.			
(D2) Null -> Alerting	Switching function alerts a device. This could be the result of a Make Call service and the called device is being alerted.			The switching function could alert a device that originates outside the device's switching domain.

Table A-1 Connection State Transition Call Flow Examples (continued)





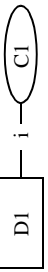

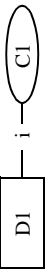
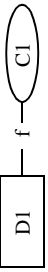
State Change	Action	Initial State	Final State	Comments
(D2) Null -> Null	Device D1 dials Device D2. Device D2 is busy on another call and cannot accept D1's call. The call is immediately forwarded and moves to the new device without ever creating a connection at D2. To report this, a Diverted event shall be based on a Null->Null transition.	 	 	
(D1) Initiated -> Null	A device that was taken off-hook is placed back on-hook before any actions are taken.			
(D1) Initiated -> Fail	A device that was taken off-hook is left off-hook without entering any digits. After a time-out period, the device receives the reorder tone and the computing function receives the Failed event.			

Table A-1 Connection State Transition Call Flow Examples (continued)

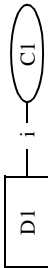
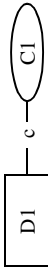

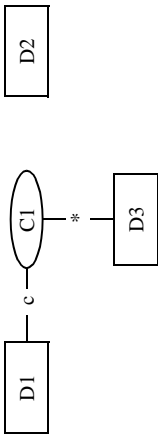




State Change	Action	Initial State	Final State	Comments
(D1) Initiated -> Connected	A device goes off-hook and finishes dialling. A call is originated to another device and the computing function receives an Originated event.			
(D2) Queued -> Null	A calling device can be queued to an ACD device or queued due to a Campon situation. The call moves from the queued device to a device that accepts the call. The computing function receives a Diverted event.			
(D2) Queued -> Alerting	A device that has a camped on call clears from its active call and is alerted by the camped on call.			
(D2) Queued -> Fail	A call is queued to an ACD device and then ACD becomes unstaffed with no alternate routing for queued calls on an unstaffed condition.			

Table A-1 Connection State Transition Call Flow Examples (continued)

State Change	Action	Initial State	Final State	Comments
(D2) Queued -> Hold	An appearance of a shared bridged device configuration that is in the inactive mode has been placed on hold as a result of actions by another appearance in the device configuration.			
(D2) Queued -> Connected	A call has been parked at device D2. The computing function successfully issues the Answer Call service for device D2.			
(D1) Connected -> Null	A device in a two-party call manually goes on-hook or a Clear Connection service is successfully invoked on behalf of this device.			
(D1) Connected -> Fail	A device that was in a two-party call stays off-hook after the other device goes on-hook. After a time-out period, the device receives the blocked tone and the computing function receives the Failed event.			

Table A-1 Connection State Transition Call Flow Examples (continued)

State Change	Action	Initial State	Final State	Comments
(D1) Connected -> Hold	A device in a two-party call manually places the other device on hold or the Hold Call service is successfully executed.			Device D1 is the device placing the call on hold.
(D2) Connected -> Queued	In a shared bridged configuration, Device D2 goes on-hook while another appearance in the bridged configuration remains connected to the call.			
(D2) Hold -> Alerting	A held or consulted call returns to a device because of a time-out.			
(D1) Hold -> Connected	A device that has previously placed another device on hold manually retrieves the held device or the Reconnect or Retrieve Call service is successfully executed.			Device D1 is the device that retrieves the call.

Table A-1 Connection State Transition Call Flow Examples (continued)

State Change	Action	Initial State	Final State	Comments
(D1) Hold -> Null	A device that has previously placed another device on hold manually goes on-hook or the Clear Connection service is successfully executed for the holding device.			
(D2) Hold -> Queued	A held appearance of a shared bridged device configuration is retrieved from hold and is placed in the inactive mode as a result of actions by another appearance in the device configuration.			
(D2) Hold -> Fail	A device that has previously placed another device on hold has exceeded the holding timer and has transitioned to the Fail state.			
(D2) Fail -> Connected	A device intrudes into a call that first failed because the called device was busy.			
(D2) Fail -> Alerting	A called device that was busy begins to alert.			

Table A-1 Connection State Transition Call Flow Examples (continued)

State Change	Action	Initial State	Final State	Comments
(D2) Fail -> Hold	An exclusive-bridged appearance has placed a call on hold and the other appearances which were blocked from use, transition to the hold state as well.			
(D2) Fail -> Queued	A called device is busy. The calling device camps on to the called device.			
(D2) Fail -> Null	A device is busy. The calling device goes back on-hook.			
(D2) Alerting -> Null	A call is delivered to a device. Before the called device answers, the calling device goes on-hook.			
(D2) Alerting -> Queued	A call is delivered to an ACD device. No agents are available so the call queues at the ACD device.			
(D2) Alerting -> Connected	A call delivered to a device. The called device either manually answers the call or the Answer Call service is used.			

Table A-1 Connection State Transition Call Flow Examples (continued)

State Change	Action	Initial State	Final State	Comments
(D2) Alerting -> Alerted	A call has been offered to a device. After a timeout the call is delivered to the same device.			
(D2) Alerting -> Fail	A call is alerting a device. That device is removed from service and there is no coverage or redirection criteria for this situation.			

Annex B (normative)

Device Appearances

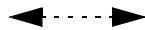
This annex describes the types of appearances supported by CSTA devices and their associated behaviour. For an introduction to logical elements and appearances, refer to 6.1.3.2, “Logical Element”, on page 14.

The type of appearance is based on its relationship with other devices. The type of appearance plays a great role in determining the functionality and behaviour associated with the logical element of the device. There are two types of appearances:

- *Standard* Appearance (see B.1)
- *Bridged* Appearance (see B.2)

The following notation is used in the figures throughout this Annex.

D	represents another device
L	represents the identifiers for the logical elements
A	represents an appearance of a logical element



indicates that there is an interaction and/or association between the elements or components of an element.

B.1 Standard Appearance

A standard appearance of a device’s logical element is not associated with other devices (i.e., cannot handle calls at another device). When a call arrives at the logical element, standard appearances are selected according to their behaviour:

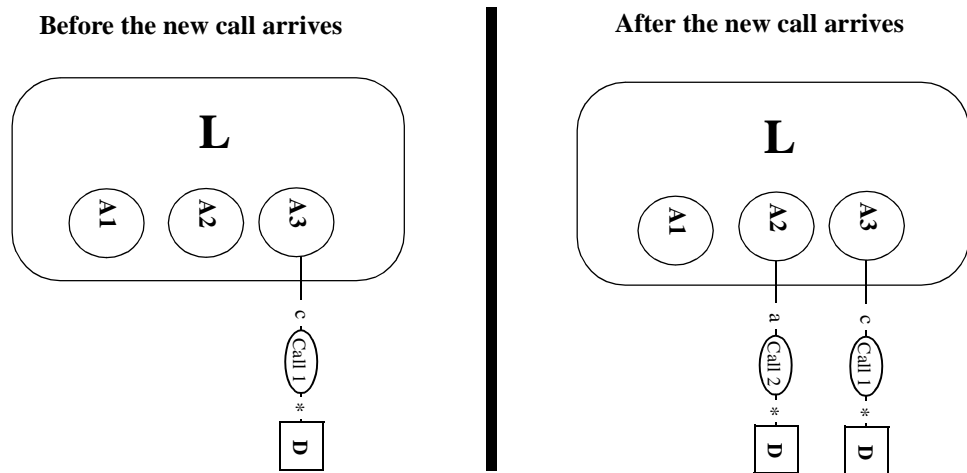
- Selected-Standard Appearance (see B.1.1)
- Basic-Standard Appearance (see B.2.1)

In addition, all media streams associated with calls corresponding to the appearances are only available to the physical element of the device (if it has one). This type of appearance can be either addressable or non-addressable. Refer to Clause 10, “CSTA Device Identifier Formats”, on page 50 for information on how to reference this appearance. When a call activity at one appearance results in the need for another appearance (i.e., the appearance puts the call on hold and wants to allow for the initiation of another call from the logical element), the logical element will use one of the available appearance (i.e., idle). If an available appearance is not present, the call will not be accepted by the logical element.

B.1.1 Selected-Standard Appearance

When calls are presented to the logical element, an available (i.e., idle) appearance is selected and only that appearance is presented with the call. From that point on, the handling of the call does not have any special characteristics or behaviours. Figure B-1 illustrates selected-standard appearances. In this example, the logical element is **L** and it has three addressable selected-standard appearances (**A1**, **A2**, **A3**). A call is currently being handled by appearance **A3**. When a new call (call 2) is presented to the logical element, it selects one of the available appearances (**A2**) to handle the call.

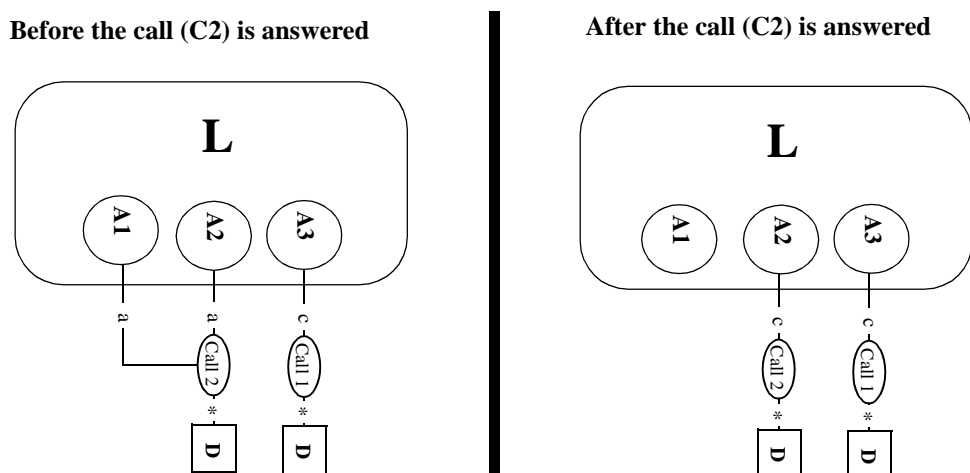
Figure B-1 Selected-Standard Appearances



B.1.2 Basic-Standard Appearance

When calls are presented to the logical element, each available (i.e., idle) appearance is presented with the call. When the call is answered by one of the appearances (i.e., the associated connection state has transitioned to the connected state), the other appearances are cleared from the call. From that point on, the handling of the call does not have any special characteristics or behaviours. The appearances that were cleared from the call can then be used to process other call and call-related activities associated with the logical element. Figure B-2 illustrates basic-standard appearances. In this example, the logical element is **L** and it has three addressable basic-standard appearances (**A1**, **A2**, **A3**). A call is currently being handled by appearance **A3**. When a new call (call 2) is presented to the logical element, it is presented to all available appearances (**A2**, **A1**). Appearance **A2** answers the call and appearance **A1** is cleared.

Figure B-2 Basic-Standard Appearances



B.2 Bridged Appearance

A bridged appearance of a device's logical element may be associated with other devices that have a physical element (i.e., can handle calls at another device).

A bridged appearance is a particular implementation of a shared logical element. Note that *associated* in the context of appearances means a relationship between a specific call appearance of a logical element and a device. A call appearance that is associated with a device implies that the call is physically handled at the device's physical element through this call appearance (e.g., makes use of the physical element's auditory apparatus)

These appearances can not be associated with devices that only have a logical element. When a call arrives at the logical element, bridged call appearances are simultaneously selected. Subsequent different behaviours are possible:

- Basic-Bridged (see B.2.1)
- Exclusive-Bridged (see B.2.2)
- Shared-Bridged (see B.2.3)

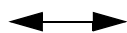
This type of appearance can be either addressable or non-addressable. Refer to Clause 10, “CSTA Device Identifier Formats”, on page 50 for information on how to reference this appearance.

When the appearance is addressable, the association between the appearance and the other device is one to one but the other device may have multiple bridged appearances associated with it. Changes in this association are reflected by the capabilities exchange services (13.1 beginning on page 80). In addition, all media streams associated with calls corresponding to this appearance are only available to the physical element (of the other device that is associated with this appearance).

When the appearance is non-addressable, the association between the appearance and the other device is one to one as long as a call is present at the appearance (i.e., a different device may be associated with this appearance every time a call is present). As a result, the media stream associated with the call is only available to the other device that was assigned to the particular appearance for the call. When a call activity at one appearance results in the need for another appearance (i.e., the appearance puts the call on hold and wants to allow for the initiation of another call from the logical element), the logical element will use one of the available appearance (i.e., idle) at the associated device. If an available appearance at the associated device is not present, the call will not be accepted by the logical element.

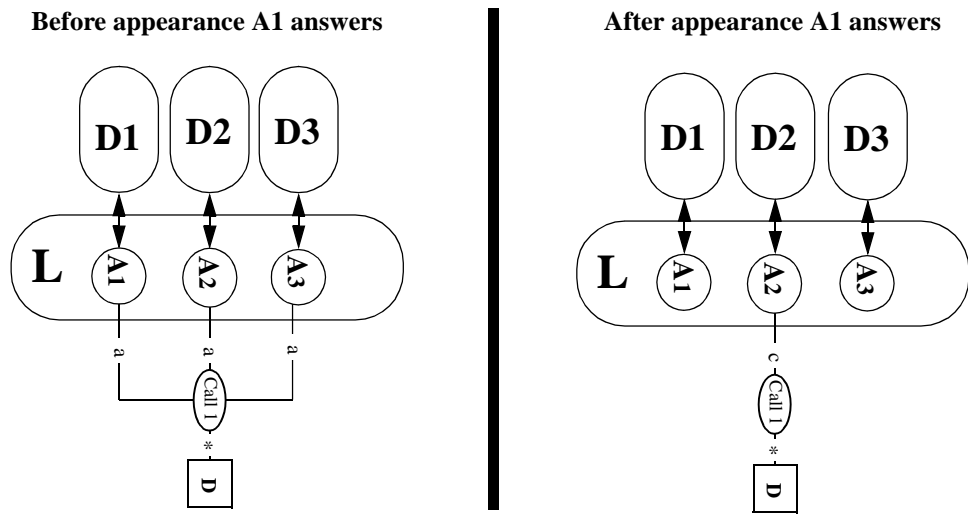
B.2.1 Basic-Bridged

When calls are presented to the logical element, each available (i.e., idle) appearance is presented with the call. When the call is answered by one of the appearances (i.e., the associated connection state has transitioned to the connected state), the other appearances are cleared from the call. From that point on, the handling of the call does not have any special characteristics or behaviours. The appearances that were cleared from the call can then be used to process other call and call-related activities associated with the logical element. Figure B-3 illustrates basic-bridged appearances. In this example, the logical element is **L** and it has three addressable basic-bridged appearances (**A1**, **A2**, **A3**). When a new call (call 1) is presented to the logical element, it is presented to all available appearances (**A1**, **A2**, **A3**). Appearance **A2** answers the call and appearances **A3** and **A1** are cleared.



The arrows indicates that the particular bridged appearance of the logical element is associated with the corresponding device.

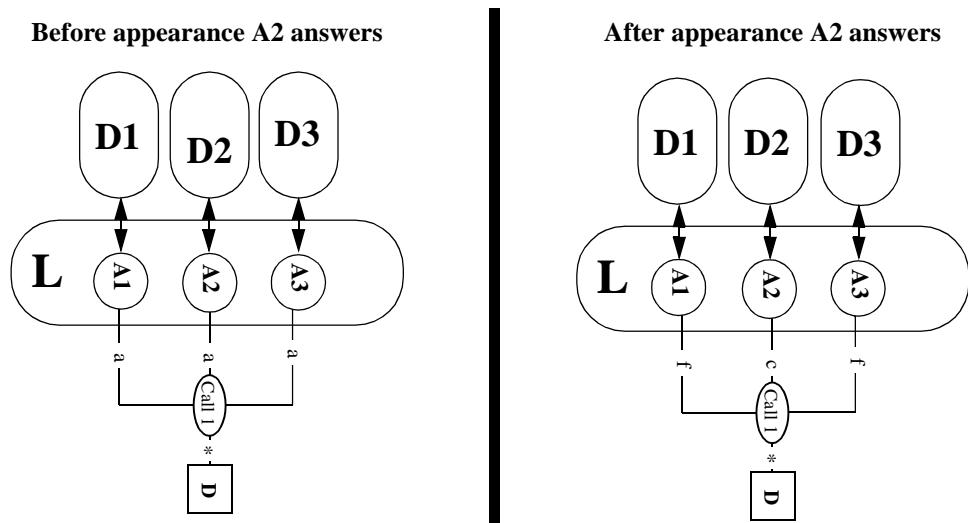
Figure B-3 Basic-Bridged Appearances



B.2.2 Exclusive-Bridged

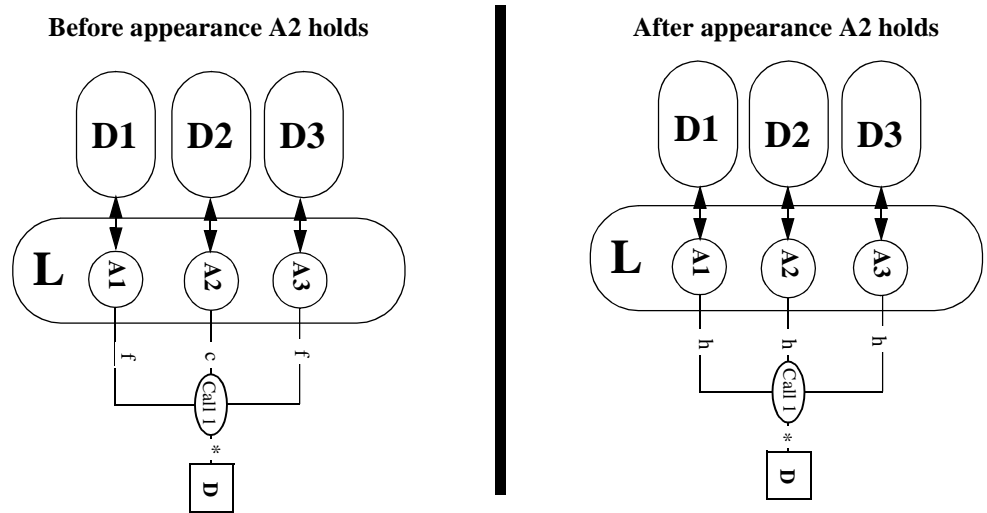
When calls are presented to the logical element, each available (i.e., idle) appearance is presented with the call. When the call is answered by one of the appearances (i.e., the associated connection state has transitioned to the connected state), the other appearances are blocked from being used (i.e., the associated connection state is transitioned to the Failed state) until the connection that answered the call is cleared or the call is moved away. At which time, the connections in the Failed state are cleared. Otherwise, the handling of the call does not have any special characteristics or behaviours. Figure B-4 illustrates exclusive-bridged appearances. In this example, the logical element **L** has three addressable exclusive-bridged appearances (**A1**, **A2**, **A3**). When a new call (call 1) is presented to the logical element, it is presented to all available appearances (**A1**, **A2**, **A3**). Appearance **A2** answers the call and appearances **A1** and **A3** are blocked.

Figure B-4 Exclusive-Bridged Appearances



When the connected appearance places the call on hold, all other appearances will transition to the hold state. If all appearances are in the hold state and one of appearances retrieves the call, then the appearance retrieving the call will transition to the connected state and the other appearances will transition to the blocked from use mode (i.e., the associated connection state is transitioned to the Failed state). Figure B-5 illustrates this behaviour of exclusive-bridged appearances (Note that this is a continuation of the illustration above). Appearance **A2** places the call on hold while appearances **A2** and **A3** are in the blocked from use mode.

Figure B-5 Exclusive-Bridged Appearances (Continuation of Figure B-4)



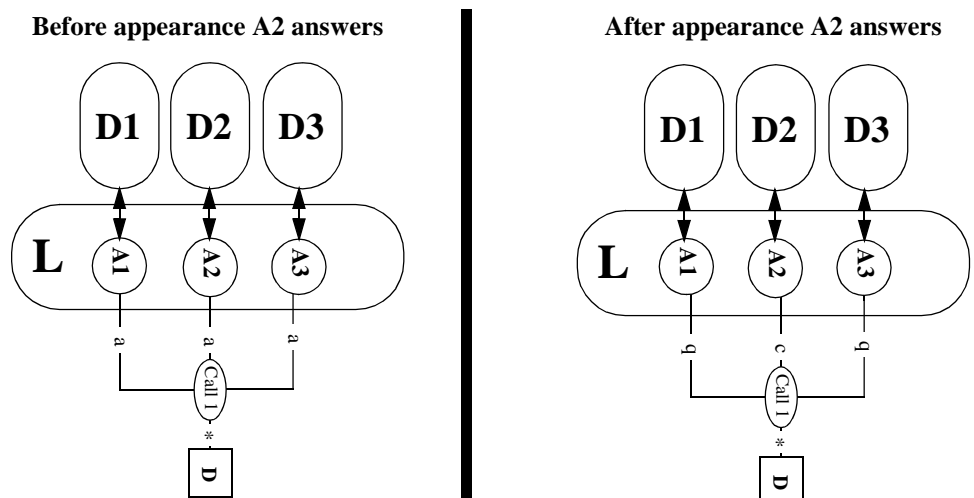
B.2.3 Shared-Bridged

This is the most common form of bridged appearances. When calls are presented to the logical element, each available (i.e., idle) appearance is presented with the call. When the call is answered by one of the appearances (i.e., the associated connection state has transitioned to the connected state), the other appearances enter an inactive mode (i.e., the associated connection state transitions to the queued state) until one of the following actions happen:

- They connect to the call.
- The call is ended.
- They are permanently cleared from the call through some feature/service.
- The call moves away from the device and none of the appearances are connected into the call.

Figure B-6 illustrates this behaviour of shared-bridged appearances. In this example, the logical element is **L** and it has three addressable shared-bridged appearances (**A1**, **A2**, **A3**). When a new call (call 1) is presented to the logical element, it is presented to all available appearances (**A1**, **A2**, **A3**). Appearance **A2** answers the call and appearances **A1** and **A3** are made inactive.

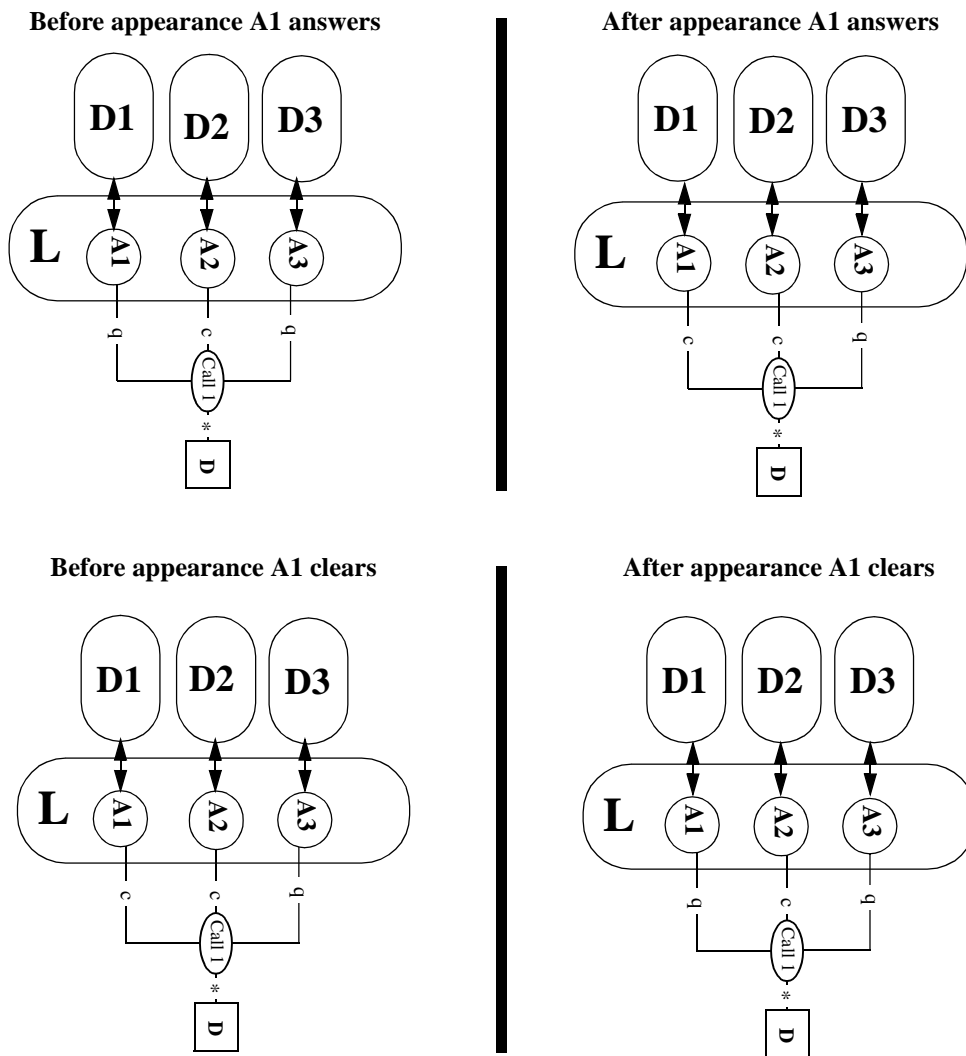
Figure B-6 Shared-Bridged Appearances



The significant behaviour of shared-bridged appearances is that any appearance can join and/or clear from the call at any time up to a switching function limit on the number of appearances connected on the call. As long as at least one of the appearances remains connected to the call, all of the other appearances retain the ability to join the call.

The Answer Call model is used to inform the computing function of appearances joining the call (Established Events). When an appearance clears from the call (with other appearances still connected), the appearance is returned to the inactive mode (i.e., the associated connection state transitions to the queued state). The call ends when all of the appearances clear from the call (i.e., all associated connection states transition to the queued state). Figure B-7 illustrates this behaviour of shared-bridged appearances (Note that this is a continuation of the illustration above). In this case, appearance **A1** adds to and clears from the call while appearance **A2** remains in the call.

Figure B-7 Shared-Bridged Appearances (Continuation of Figure B-6)

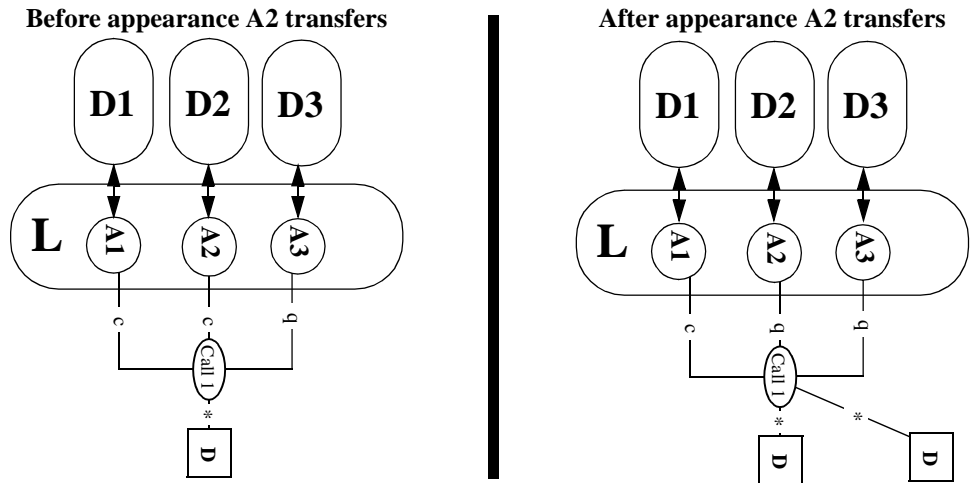


When the call is moved away from an appearance or an appearance places the call on hold, the shared-bridged appearance type becomes two types:

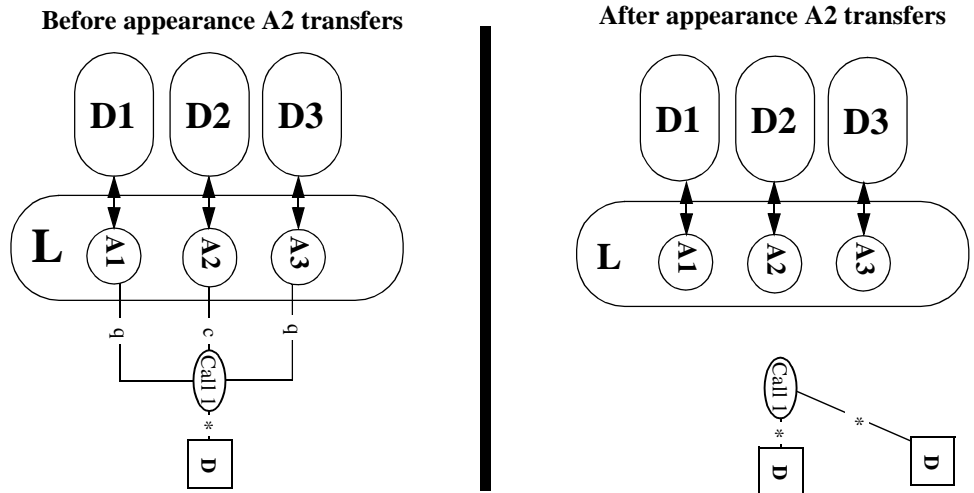
- *Independent-shared-bridged* - When the call is moved away from the logical element by one of the appearances, the other appearances are unaffected, as long as one appearance remains connected in the call, otherwise all appearances are cleared from the call. As for the appearance moving the call, it will return to the inactive mode (if another appearance remains connected in the call). Figure B-8 illustrates this behaviour of independent-shared-bridged appearances (Note that this is a continuation of the illustration above). In case one, appearance **A2** immediately transfers the call to another device while appearance **A1** remains in the call. In case two, appearance **A2** immediately transfers the call to another device while all appearances **A1** and **A3** are in the inactive mode.

Figure B-8 Independent-Shared-Bridged Appearances

Case One

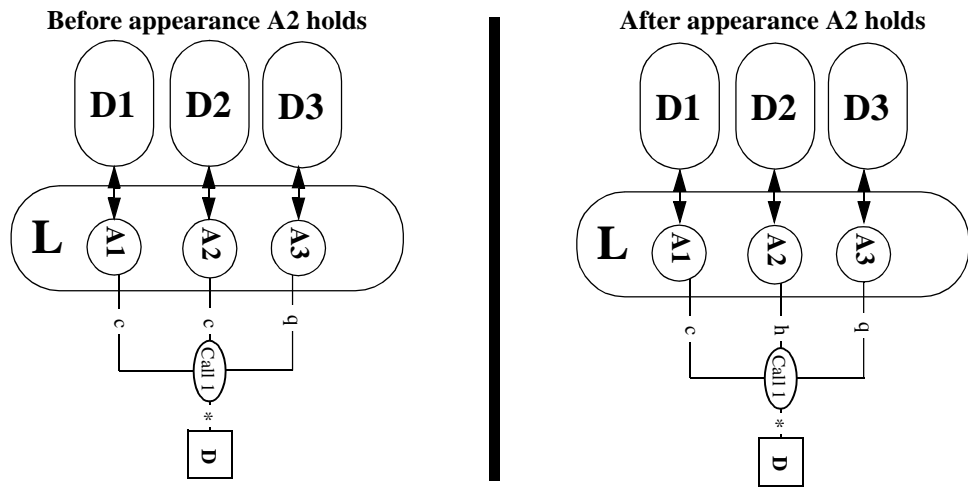


Case Two

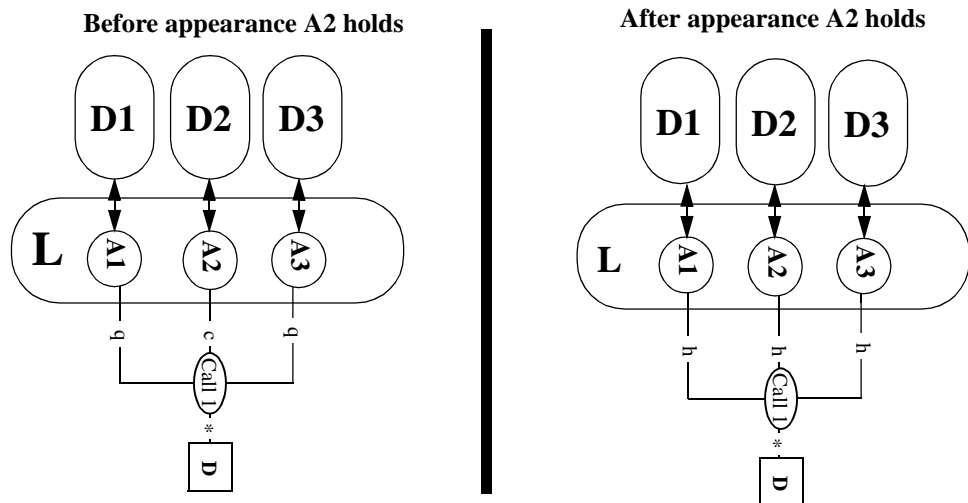


When one of the appearances places the call on hold and at least one other appearance is connected in the call, only the holding appearance will transition to the hold state (i.e., the other appearances are unaffected). When one of the appearances places the call on hold and the holding appearance was the only one connected in the call, all other appearances will transition to the hold state. If all appearances are in the hold state and one of appearances retrieves the call, then the appearance retrieving the call will transition to the connected state and the other appearances will transition to the inactive mode. Figure B-9 illustrates this behaviour of independent-shared-bridged appearances (Note that this is a continuation of the illustration above). In case one, appearance **A2** places the call on hold while appearance **A1** remains in the call. In case two, appearance **A2** places the call on hold while all appearances **A2** and **A3** are in the inactive mode.

Figure B-9 Independent-Shared-Bridged Appearances (Continuation of Figure B-8)
Case One



Case Two



- *Interdependent-shared-bridged* - When the call is moved away from the logical element by one of the appearances (no matter what the connection state of the other appearances in the call), all appearances are cleared from the call. Figure B-8 in Case 2 illustrates this behaviour of interdependent-shared-bridged appearances.

When one of the appearances places the call on hold, all appearances will transition to the hold state. When one of appearances retrieves the call, then the appearance retrieving the call will transition to the connected state and the other appearances will transition to the inactive mode. Figure B-9 in Case 2 illustrates this behaviour of interdependent-shared-bridged appearances.

Annex C
(informative)

Examples of Device Appearances (to be defined)

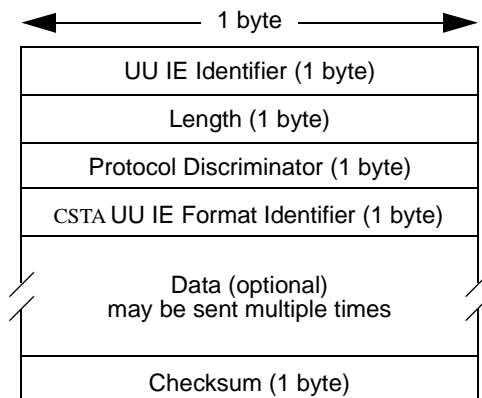
Annex D (Normative)

ISDN User-User Information Element Encoding for CSTA

Correlator data, user data, or both can be delivered to a switching sub-domain from an ISDN network connection. The data is transmitted through the network in the ISDN User-User (UU) Information Element (IE) in ISDN call control and User Info messages where supported. Transmission of User-User data is not supported by all parts of the public network.

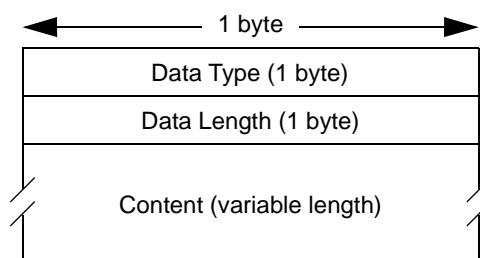
The following CSTA UU IE format shall be used for packing correlator data, user data, or both. The first three bytes of the UU IE are defined in ITU-T Recommendation Q.931.

Figure D-1 CSTA User-User Information Element (UU IE) Field Format



- The *UU IE Identifier* field identifies the UU IE packet. Its value is 0x7E.
- The *Length* field provides the total length of the UU IE packet.
- The *Protocol Discriminator* signals end-to-end transparency to the public switch vendor. Its value is 0x00.
- The CSTA UU IE Format Identifier identifies that the information element contains correlator data, user data, or both. Its value is 0x45.
- The *Data* field is optional, may be sent multiple times and has the following format:

Figure D-2 Data Field Format



- The *Data Type* sub-field specifies whether the data is correlator data (value is 0x80) or User Data (value is 0x81).
- The *Data Length* sub-field provides the length of the Content field.
- The *Content* sub-field contains the appropriate Data (correlator data or user data).
- The *Checksum* field verifies that the packet is the CSTA UU IE packet defined in this section.

Functional Requirements

1. Multiple Data Type entries may be sent in each UU IE. The entries may be repeated up to a maximum packet length of 128 bytes.¹
2. A switching function reports its supported maximum lengths for Correlator Data and User Data in the capabilities exchange services.
3. The possible maximum data lengths in ISDN messages are 121 bytes for User Data and 32 bytes for Correlator Data.¹
4. If a switching function does not support a data type, it is discarded. If a switching function does not support as many entries of a particular type as were received, the additional entries are discarded.
5. Length 0 (zero) of Correlator Data is used to signal null Correlator Data to the receiving switching sub-domain. Current Correlator Data associated with the call shall be deleted.
6. Format and interpretation of the Content sub-field is a function of the computing function.
7. The Checksum byte contains the XOR checksum of all the bytes starting with the CSTA UU IE Format Identifier.

1. The maximum size of an ISDN UU IE packet may be limited based upon certain implementations (i.e., some public network implementations).

Printed copies can be ordered from:

ECMA
114 Rue du Rhône
CH-1204 Geneva
Switzerland

Fax: +41 22 849.60.01
Internet: documents@ecma.ch

Files can be downloaded from our FTP site, **ftp.ecma.ch**, logging in as **anonymous** and giving your E-mail address as **password**. This Standard is available from library **ECMA-ST** as an Acrobat PDF file (file E269-PDF.PDF). File E269-EXP.TXT gives a short presentation of the Standard.

Our web site, <http://www.ecma.ch>, gives full information on ECMA, ECMA activities, ECMA Standards and Technical Reports.

ECMA

**114 Rue du Rhône
CH-1204 Geneva
Switzerland**

This Standard ECMA-269 is available free of charge in printed form and as a file.

See inside cover page for instructions