



A MITEL  
PRODUCT  
GUIDE

# Unify OpenScape 4000

OpenScape 4000 V11, CSTA Connectivity Adapter - Application  
Developer's Guide

Programming Guide

03/2025

## Notices

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Europe Limited. The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes. No part of this document can be reproduced or transmitted in any form or by any means - electronic or mechanical - for any purpose without written permission from Mitel Networks Corporation.

## Trademarks

The trademarks, service marks, logos, and graphics (collectively "Trademarks") appearing on Mitel's Internet sites or in its publications are registered and unregistered trademarks of Mitel Networks Corporation (MNC) or its subsidiaries (collectively "Mitel"), Unify Software and Solutions GmbH & Co. KG or its affiliates (collectively "Unify") or others. Use of the Trademarks is prohibited without the express consent from Mitel and/or Unify. Please contact our legal department at [iplegal@mitel.com](mailto:iplegal@mitel.com) for additional information. For a list of the worldwide Mitel and Unify registered trademarks, please refer to the website: <http://www.mitel.com/trademarks>.

© Copyright 2025, Mitel Networks Corporation

All rights reserved

# Contents

<b>1 About This Book.....</b>	<b>11</b>
1.1 Who Should Read This Guide.....	11
1.2 How This Guide is Organized.....	11
1.3 Related Information.....	12
1.4 Documentation Feedback.....	12
<b>Part 1: Services and Events.....</b>	<b>13</b>
<b>2 OpenScape 4000 CSTA and Applications.....</b>	<b>13</b>
3.1 OpenScape 4000 CSTA System Overview.....	13
3.2 OpenScape 4000 CSTA System Characteristics.....	14
3.2.1 TCP/IP Stream Content.....	14
3.2.2 Interface.....	14
3.2.3 Application Link Start Up Sequence.....	16
3.2.4 Exceptional Behaviour and Recovery.....	17
3.2.4.1 Hang-up/Disconnect of Application.....	17
3.2.4.2 Restart of OpenScape 4000 / PBX Link Failure.....	17
3.2.4.3 Restart of the OpenScape 4000 CSTA.....	17
3.2.5 System Status.....	18
3.2.6 Correspondence between ACSE-licensing.....	19
3.3 Application Examples.....	20
3.3.1 Intelligent Answering.....	20
3.3.2 Coordinated Voice and Data Transfer.....	22
3.3.3 Routing Services.....	22
3.3.3.1 Get/Set Routeing Mode.....	23
3.3.3.2 Route Request and Route Select Dialog.....	23
3.3.3.3 Route Reject Dialog.....	24
3.3.3.4 Route End Dialog.....	25
3.3.3.5 Re-Route Request Dialog.....	26
3.3.4 Enhanced Business Statistics.....	27
3.3.5 Automated Outbound Dialing.....	27
3.4 ACSE messages.....	27
3.4.1 ACSE AARQ.....	27
3.4.2 ACSE AARE.....	29
3.5 Restrictions.....	30
<b>3 OpenScape 4000 CSTA Services .....</b>	<b>31</b>
4.1 General Notes.....	31
4.2 Services Descriptions.....	32
4.3 Services Summary.....	32
4.4 Capability Exchange Services.....	38
4.4.1 Get Logical Device Information.....	38
4.4.2 Get Physical Device Information.....	47
4.4.3 Get Switching Function Capabilities.....	50
4.4.4 Get Switching Function Devices.....	60
4.4.5 Switching Function Devices.....	62
4.5 System Services.....	65
4.5.1 Change System Status Filter.....	68
4.5.2 System Register.....	70
4.5.3 System Register Abort.....	72
4.5.4 System Register Cancel.....	72
4.5.5 Request System Status.....	73

4.5.6 System Status.....	74
4.5.7 Switching Function Devices Changed.....	79
4.6 Monitoring Services.....	80
4.6.1 Change Monitor Filter.....	80
4.6.2 Monitor Start.....	81
4.6.3 Monitor Stop.....	86
4.7 Snapshot Services.....	87
4.7.1 Snapshot Call.....	88
4.7.2 Snapshot CallData.....	91
4.7.3 Snapshot Device.....	94
4.7.4 Snapshot DeviceData.....	96
4.8 Call Control Services.....	99
4.8.1 Accept Call.....	100
4.8.2 Alternate Call.....	101
4.8.3 Answer Call.....	102
4.8.4 Call Back Call-Related.....	104
4.8.5 Clear Connection.....	106
4.8.6 Conference Call.....	108
4.8.7 Consultation Call.....	110
4.8.8 Deflect Call.....	112
4.8.9 Dial Digits.....	116
4.8.10 Hold Call.....	117
4.8.11 Make Call.....	119
4.8.12 Make Predictive Call.....	123
4.8.13 Reconnect Call.....	125
4.8.14 Retrieve Call.....	127
4.8.15 Single Step Transfer Call.....	129
4.8.16 Transfer Call.....	132
4.9 Call Associated Feature Services.....	134
4.9.1 Generate Digits.....	134
4.9.2 Generate Telephony Tones.....	136
4.9.3 Send User Information.....	139
4.10 Routing Registration Services.....	141
4.10.1 Route Register.....	141
4.10.2 Route Register Abort.....	143
4.10.3 Route Register Cancel.....	144
4.11 Routing Services.....	145
4.11.1 Re-Route.....	145
4.11.2 Route End.....	146
4.11.3 Route Reject.....	149
4.11.4 Route Request.....	150
4.11.5 Route Select.....	152
4.12 Physical Device Feature Services.....	154
4.12.1 Get Message Waiting Indicator.....	154
4.12.2 Set Display.....	155
4.12.3 Set Lamp Mode.....	157
4.12.4 Set Message Waiting Indicator.....	160
4.12.5 Button Press.....	161
4.12.6 Get Microphone Mute.....	162
4.12.7 Set Microphone Mute.....	164
4.13 Logical Device Feature Services.....	165
4.13.1 Cancel Call Back.....	165
4.13.2 Get Agent State.....	167
4.13.3 Get Do Not Disturb.....	171
4.13.4 Get Forwarding.....	172
4.13.5 Get Routeing Mode.....	176

4.13.6 Set Agent State.....	178
4.13.7 Set Do Not Disturb.....	182
4.13.8 Set Forwarding.....	184
4.13.9 Set Routeing Mode.....	187
4.14 I/O Registration Services.....	190
4.14.1 I/O Register.....	190
4.14.2 I/O Register Abort.....	192
4.14.3 I/O Register Cancel.....	193
4.15 Input / Output Services.....	194
4.15.1 Data Path Resumed.....	194
4.15.2 Data Path Suspended.....	195
4.15.3 Fast Data.....	196
4.15.4 Send Data.....	199
4.15.5 Start Data Path.....	202
4.15.6 Stop Data Path.....	205
4.16 Vendor Specific Extensions Services.....	206
4.16.1 Escape Services.....	207
4.16.1.1 Set lower Class of Service.....	207
4.16.1.2 Get lower Class of Service.....	208
4.16.2 Private Data Version Selection.....	210
<b>4 Events.....</b>	<b>212</b>
5.1 Call Control Events.....	215
5.1.1 Bridged.....	218
5.1.2 Connection Cleared.....	219
5.1.3 Delivered.....	221
5.1.4 Digits Dialed.....	225
5.1.5 Diverted.....	228
5.1.6 Established.....	232
5.1.7 Failed.....	235
5.1.8 Held.....	238
5.1.9 Network Reached.....	239
5.1.10 Offered.....	242
5.1.11 Originated.....	243
5.1.12 Queued.....	245
5.1.13 Retrieved.....	248
5.1.14 Service Initiated.....	250
5.1.15 Transferred.....	251
5.2 Call Associated Feature Events.....	254
5.2.1 Call Information.....	254
5.2.2 Digits Generated.....	255
5.2.3 Telephony Tones Generated.....	256
5.3 Logical Device Feature Events.....	258
5.3.1 Agent Busy.....	258
5.3.2 Agent Logged Off.....	259
5.3.3 Agent Logged On.....	260
5.3.4 Agent Not Ready.....	260
5.3.5 Agent Ready.....	261
5.3.6 Agent Working After Call.....	262
5.3.7 Call Back.....	262
5.3.8 Do Not Disturb.....	263
5.3.9 Forwarding.....	264
5.3.10 Routeing Mode.....	266
5.4 Physical Device Feature Events.....	267
5.4.1 Message Waiting.....	267
5.5 Maintenance Events.....	268

5.5.1 Back In Service.....	268
5.5.2 Out Of Service.....	268
5.6 Vendor Specific Extensions Events.....	269
5.6.1 Mobile User Status ( Private Event ).....	269
<b>Part 2: Call Scenarios.....</b>	<b>270</b>
<b>5 Call Scenarios.....</b>	<b>270</b>
7.1 Scope.....	270
7.2 References.....	271
7.3 Definitions and Abbreviations.....	271
7.4 Call Origination Scenarios.....	271
7.4.1 Manually dialled call.....	272
7.4.2 Manually dialled call - Called party is busy.....	274
7.4.3 Manually dialled call - Called party is Out Of Service (OOS).....	275
7.4.4 Manually dialled call - Dialed number is invalid.....	277
7.4.5 Manually dialled call - Incomplete dialling sequence, calling party goes onhook.....	279
7.4.6 Manually dialled call - Incomplete dialling sequence, dialling has timed out.....	280
7.4.7 Make Call service.....	281
7.4.8 Multi Stage dialling.....	282
7.4.9 Call offered to an application.....	284
7.5 Answering Call Scenarios.....	287
7.5.1 Succesful answer call.....	287
7.6 Connection Termination Scenarios.....	288
7.6.1 Device disconnects from a call by on-hook.....	288
7.6.1.1 Non-SIP Device disconnects from a call by on-hook.....	288
7.6.1.2 SIP Device disconnects from a call by on-hook.....	289
7.6.2 Device disconnects from a call using the Clear Connection service (remaining device goes blocked).....	291
7.7 External outgoing calls.....	292
7.7.1 Manual call to a device outside the CSTA subdomain.....	292
7.7.2 Manual call to a busy device outside the CSTA subdomain.....	295
7.7.3 External outgoing camp-on.....	297
7.8 External incoming calls.....	298
7.8.1 External incoming call.....	299
7.8.2 Incoming external call to a busy device.....	301
7.8.3 External incoming camp-on.....	302
7.9 Forwarding Call Scenarios.....	305
7.9.1 Call Forward No Answer.....	305
7.9.2 Call Forward no Answer: Forwarding Device and Destination (Busy) have Offered Mode on.....	307
7.9.3 Call Forward Immediate.....	310
7.9.4 Call Forward Immediate: Called Party is OOS (Out Of Service).....	312
7.9.5 Call Forward Busy.....	314
7.10 Multiple Forwarding Scenarios.....	314
7.10.1 Call Forward Immediate followed by Call Forward No Answer.....	314
7.10.2 Call Forward No Answer followed by Call Forward Immediate.....	317
7.10.2.1 Destination is available.....	317
7.10.2.2 Destination is not available.....	319
7.10.3 Call Forward Immediate followed by Call Forward Busy.....	321
7.11 Call Movement Scenarios.....	324
7.11.1 Deflect Call service.....	324
7.11.2 Deflect call with ReRouting enabled.....	326
7.11.3 Manual group pickup.....	330
7.11.4 Manual directed park call.....	332
7.11.5 Manual system park.....	335
7.11.6 Manual system unpark.....	338

7.11.7 Call movement, Forwarding or Pick Up on Remote OpenScape 4000.....	339
7.12 Hold/Retrieving Scenarios.....	339
7.12.1 Hold/Retrieve Call.....	339
7.12.1.1 Manual hold/Retrieve feature for DIGITE (except for KEYSETS).....	340
7.12.1.2 CSTA Hold/Retrieve feature for Client/Desk.....	343
7.12.2 Hold / Retrieve on Remote OpenScape 4000.....	344
7.13 Consultation Call Scenarios.....	344
7.13.1 Successful Consultation Call.....	344
7.13.1.1 Consulting party is a Non-SIP device.....	344
7.13.1.2 Consulting party is a SIP device.....	347
7.13.2 Consulting out of a conference.....	350
7.13.3 Reconnect Call.....	352
7.13.4 Held Party Releases.....	354
7.13.4.1 Consulting device is a Non-SIP device.....	354
7.13.4.2 Consulting device is a SIP device.....	356
7.13.5 Alternate Call.....	357
7.13.5.1 Consulting party is a Non-SIP device.....	358
7.13.5.2 Consulting party is a SIP device.....	359
7.14 Transfer Call Scenarios.....	361
7.14.1 Screened Transfer (with local view in Transferred event).....	361
7.14.1.1 Transferring party is a Non-SIP device.....	361
7.14.1.2 Transferring party is a SIP device.....	363
7.14.2 Blind Transfer (with local view in Transferred event).....	365
7.14.2.1 Transferring device is a Non-SIP device.....	365
7.14.2.2 Transferring device is a SIP device.....	367
7.14.3 Transfer to a busy station (with local view in Transferred event).....	370
7.14.4 Single Step Transfer (with local view in Transferred event).....	372
7.14.5 Single Step Transfer between network interface devices (with local view in Transferred event).....	374
7.14.6 Single Step Call Transfer, Phone Mail transfers.....	377
7.14.7 Single Step Transfer to destination with call forward immediate handled in the switching subdomain (D3 is internal analogue or digital device).....	379
7.14.8 Single Step Transfer attempt to busy destination with offered mode activated.....	382
7.14.9 Single Step Transfer for Consulting Party.....	386
7.14.10 Transfer on Remote OpenScape 4000.....	386
7.15 Conference Call Scenarios.....	387
7.15.1 Conference (with local view in Conferenced event).....	387
7.15.1.1 Conference master is a Non-SIP device.....	387
7.15.1.2 Conference master is a SIP device.....	388
7.16 Call Completion Scenarios.....	390
7.16.1 Call Back Call Related.....	390
7.16.2 Manual Camp On Call.....	393
7.17 Distribution Call Scenarios.....	394
7.17.1 Automatic Call Distribution Scenarios.....	394
7.17.1.1 Automatic Call Distribution Overview.....	395
7.17.1.2 External ACD Call completed to agent.....	398
7.17.1.3 Internal ACD call completed to Phone Mail agent=.....	402
7.17.1.4 External call overflow to another RCG.....	404
7.17.2 Make Predictive Call.....	407
7.17.2.1 Make Predictive Call - to external free device.....	407
7.17.2.2 Make Predictive Call - to external busy device.....	411
7.17.3 Route Services.....	413
7.17.3.1 Route Request Scenario.....	414
7.17.3.2 Re-Route Request Scenario.....	418
7.17.3.3 Route End Request Scenario.....	420
7.17.3.4 Reject Call Scenario.....	421
7.17.4 Hunting Groups (HG).....	422

7.17.4.1 General description Hunt Group.....	423
7.17.4.2 Successful Group Call (Multiple Alerting with Parallel Ringing).....	426
7.17.4.3 Internal call to Hunt Group, Hunt Advance.....	430
7.17.4.4 Call is queued at Hunt Group.....	434
7.17.4.5 Call is routed to Overflow Destination.....	435
7.17.4.6 Transfer Ringing into Hunt Group.....	437
7.17.4.7 Pick from Hunt Group Member.....	438
7.17.5 General Attendant (GA).....	441
7.17.5.1 General description GA.....	441
7.17.5.2 Internal call to GA2Q.....	445
7.17.5.3 Internal call to GAMQ.....	445
7.17.5.4 Overflow from one GA to another GA.....	448
7.17.5.5 Intercept without parallel call to GA2Q.....	450
7.17.5.6 Intercept with parallel call to GA2Q.....	453
7.17.5.7 Intercept with parallel call to GAMQ.....	455
7.17.5.8 Intercept with parallel call, call is routed to another GA after timeout.....	458
7.17.5.9 Trunk-to-trunk supervision.....	461
7.17.5.10 Night-Service: General Night Station answers.....	463
7.17.5.11 Night-Service: External Centralized Attendant Service group (CASEXT).....	465
7.17.5.12 Intercept on a transit node.....	467
7.17.5.13 Call forwarded (CFNA) to Attendant, Multiple alerting, providing Diverted event to calling side is enabled.....	470
7.18 Recall Scenarios.....	474
7.18.1 Softhold Recall.....	474
7.18.2 Transfer Recall.....	475
7.18.3 Transfer with restricted Connection.....	476
7.18.4 Conference Recall.....	478
7.18.5 Park Timer expires.....	480
7.18.6 Park Recall Timer Expires.....	481
7.18.7 Hard Hold Recall.....	482
7.19 OpenScope 4000 Specific Features.....	483
7.19.1 Route Optimization.....	484
7.19.2 Override.....	486
7.19.2.1 Network-wide override.....	487
7.19.2.2 D2 goes onhook after the override.....	489
7.19.2.3 D1 hits clear after the override.....	491
7.19.3 Silent Monitor.....	493
7.19.3.1 Agent goes onhook, afterwards original caller calls agent again.....	495
7.19.4 Transfer before ALERT.....	498
7.19.5 Keyset Call (Multiline device call).....	500
7.19.6 Making Calls in an Executive-Secretary team (CheSe feature).....	503
7.19.6.1 General Remarks.....	503
7.19.6.2 Successful basic call - call to Executive.....	504
7.19.6.3 Successful basic call - Representative is activated.....	505
7.19.6.4 Unsuccessful basic call - Secretary is busy.....	506
7.19.6.5 Unsuccessful basic call - Executive is busy.....	507
7.19.6.6 Camp on Executive - Secretary is busy.....	507
7.19.6.7 Camp on Executive - Executive is busy.....	509
7.19.7 Single Step Call Transfer with Await Connect.....	510
7.19.7.1 Successful basic SSCT call with Await Connect.....	510
7.19.7.2 Unsuccessful basic SSCT call with Await Connect.....	512
7.19.7.3 SSCT call with Await Connect, Camp On.....	514
7.19.7.4 SSCT call with Await Connect , Pick Up.....	516
7.19.8 Concept of "presentation indicator for devices" in CSTA events.....	519
7.19.8.1 The old concept of presentation indicator ("normal").....	520
7.19.8.2 Basic Internal Call with presentation restricted devices (CSTA III).....	520

7.19.8.3 Blind Transfer.....	522
7.19.8.4 Conference Initiation by Conference Master with presentation restricted devices.....	523
7.19.8.5 Blind Transfer with presentation restricted devices (CSTA III).....	524
7.19.8.6 Conference with presentation restricted devices (CSTA III).....	525
7.19.8.7 Presentation restricted is ignored.....	527
7.19.8.8 Presentation indicator represented by Private Data.....	527
7.19.8.9 Illustration of the new concept:.....	528
7.19.8.10 Basic call with presentation restricted devices.....	528
7.19.8.11 Blind Transfer with presentation restricted devices.....	532
7.19.8.12 Conference with presentation restricted devices.....	533
7.19.8.13 Affected events.....	535
7.19.8.14 Remarks.....	536
7.19.8.15 Multiple calls.....	536
7.19.8.16 Configuration of presentation indicator.....	536
7.19.8.17 Configure the presentation indicator by AMO.....	536
7.19.8.18 Configure the presentation indicator via OptiSet menu.....	536
7.19.9 Connect and Reconnect Timeslot Escape Services.....	536
7.19.9.1 Connect Timeslot Escape Service.....	536
7.19.9.2 Reconnect Timeslot Escape Service.....	537
<b>6 Device Identifier Formats.....</b>	<b>538</b>
<b>7 Appendix A - Generic Display Protocol.....</b>	<b>541</b>
<b>8 Appendix B - Representation of keys in the IOData field.....</b>	<b>545</b>
<b>9 Appendix C - Private Data.....</b>	<b>550</b>
11.1 CapExchangeServPrivParams.....	551
11.2 SystemStatusServPrivParams.....	551
11.3 MonitoringServPrivParams.....	552
11.4 SnapshotServPrivParams.....	552
11.5 CallControlEvtsPrivParams.....	553
11.6 IOServicesServPrivParams.....	555
11.7 VendorSpecificServPrivParams.....	555
11.7.1 CallFacilities.....	558
11.8 VendorSpecificEvtsPrivParams.....	558
11.8.1 MobileUserStatusEvent.....	559
11.9 CallControlServPrivParams.....	559
11.10 BasicEvtsPrivParams.....	559
11.11 LogicalServPrivParams.....	561
<b>10 Appendix D - Support of uaCSTA.....</b>	<b>562</b>
<b>Part 3: Shared-Bridged Appearance CSTA Interface Specification.....</b>	<b>564</b>
<b>11 Shared-bridged appearances modeling.....</b>	<b>564</b>
14.1 Device Identifier and DeviceID Presentation.....	569
14.2 Supported Phones.....	570
14.3 Supported CSTA Services.....	570
14.4 Feature Activation.....	575
14.5 CALL SCENARIOS.....	575
14.5.1 Logical Device Monitoring.....	575
14.5.2 Discovery Services.....	575
14.5.2.1 Example Configuration.....	576
14.5.2.2 Get Logical Device Information.....	576
14.5.2.3 Get Physical Device Information.....	578
14.5.3 Snapshot Services.....	578
14.5.3.1 Snapshot Device.....	578

## Contents

14.5.3.2 Snapshot Call.....	582
14.5.4 Call Origination.....	583
14.5.4.1 Basic Call answered on secondary line appearance.....	583
14.5.4.2 Basic Call to Busy Logical/Physical Device (no call waiting support).....	587
14.5.4.3 Basic Call - To Phantom Line.....	589
14.5.5 Call Completion Services.....	591
14.5.5.1 Call Back - Call Related.....	591
14.5.6 Bridged Call Flows.....	594
14.5.6.1 Bridging onto a Bridged/Queued connection using the Answer Call Service.....	594
14.5.6.2 Available Line Selection via CSTA.....	596
14.5.6.3 Releasing from Bridge Call using Clear Connection Service.....	599
14.5.7 Terminating a Bridged Call.....	601
14.5.7.1 Device releases from a bridged connection.....	601
14.5.8 Hold and Retrieve Call Flows.....	603
14.5.8.1 Placing a Shared-Bridged appearance on Hold using CSTA Hold Call service.....	603
14.5.8.2 Retrieving a Shared-Bridged appearance on Hold using CSTA Retrieve Call service.....	605
14.5.8.3 Placing a Shared-Bridged appearance on Hold using CSTA Hold Call service while on-hold.....	607
14.5.9 Call Forwarding and Call Diversion.....	609
14.5.9.1 Call Forward No Answer - Logical Device.....	609
14.5.9.2 Deflect Call - From Hunt Group to logical device.....	612
14.5.10 Transfer.....	615
14.5.10.1 Semi-Attended/Blind Transfer.....	615
14.5.10.2 Attended Transfer.....	619
14.5.10.3 Single Step Transfer.....	623
14.5.10.4 Semi-Attended/Blind Transfer (with multiple consulted appearances alerting).....	625
14.5.11 Conference.....	630
14.5.11.1 Conference Call Service.....	630
<b>12 Glossary.....</b>	<b>635</b>
<b>Index.....</b>	<b>642</b>

# 1 About This Book

OpenScape 4000 CSTA is a middleware that allows you to link the OpenScape 4000 system with computer environments that support the CSTA Phase III (Computer-Supported Telecommunications Applications) standard protocol. The Programming Guide describes the Computer Telephony Integration (CTI) supported by OpenScape 4000.

OpenScape 4000 CSTA software supports the CTI applications through its services and events. Services and events make it possible for a computer application to connect and interact with inbound or outbound telephone calls and perform other types of telephone activities under application control.

## 1.1 Who Should Read This Guide

This guide provides the information needed to understand the OpenScape 4000 system's built in CSTA III interface and to plan and develop software for CTI application programs. Use this document together with the documentation provided by OpenScape 4000.

## 1.2 How This Guide is Organized

### Part 1 (Services and Events)

- [Chapter 2, "OpenScape 4000 CSTA and Applications"](#), describes the product and explains how a computer application can use OpenScape 4000 CSTA III Interface.
- [Chapter 3, "OpenScape 4000 CSTA Services"](#), provides usage notes and descriptions of the CSTA III services. Services enable the computer application to control a variety of telephone functions such as dialing a call, answering a call, clearing a call, making consultation and conference calls, and routing a call.
- [Chapter 4, "Events"](#), provides usage notes and descriptions of the CSTA III events. Events provide call information to applications for tracking telephony activities. Events are always generated if an action takes place at a device, whether through manual invocation of the feature or through a service request. If the activity is not permitted on a device, no event can be generated.
- This part contains also a [Glossary](#).

### Part 2 (Call Scenarios)

- [Chapter 5, "Call Scenarios"](#): Call scenarios provide step-by-step descriptions of scenarios and lists for each activity of each scenario, the services, events and their parameters.

### Appendix

- [Chapter 7, "Appendix A - Generic Display Protocol"](#) describes the device-independent display protocol as it is used in the OpenScape 4000 CSTA.
- [Chapter 8, "Appendix B - Representation of keys in the IOData field"](#) describes the detailed octal representation of pressed keys in the I/O services' IOData field.

- [Chapter 9, "Appendix C - Private Data"](#) describes the private data supported and provided by the OpenScape 4000' switching subdomain.

#### **Glossary**

- [Glossary](#)

## **1.3 Related Information**

For more information, refer to the related publications:

#### **ECMA Documentation**

CSTA refers to the standards of the European Computer Manufacturing Association (ECMA) for Computer-Telephony Integration (CTI). These standards are published as ECMA-269 and ECMA-285. Access to the below listed European Computer Manufacturers Association documents may be obtained at the web location of <http://www.ecma.ch/>:

- Standard ECMA-269, Services for Computer-Supported Telecommunications Applications (CSTA), Dec 2011.
- Standard ECMA-285, Protocol for Computer-Supported Telecommunications Applications (CSTA), Dec 2011.
- Standard ECMA-323: XML Protocol for Computer-Supported Telecommunications Applications
- Standard ECMA-TR 82, Scenarios for Computer-Supported Telecommunications Applications (CSTA), June 2009.

## **1.4 Documentation Feedback**

To report a problem with this document, call your next level of support:

- Unify employees should call the National Support Center.
- Customers should call the Unify Customer Support Center.

When you call, be sure to include the following information. This will help identify which document you are having problems with.

- **Title:** OpenScape 4000 CSTA, Connectivity Adapter - CSTA III, Developers Guide
- **Order Number:** A31003-H31A0-R100-06-7620

## Part 1: Services and Events

## 2 OpenScape 4000 CSTA and Applications

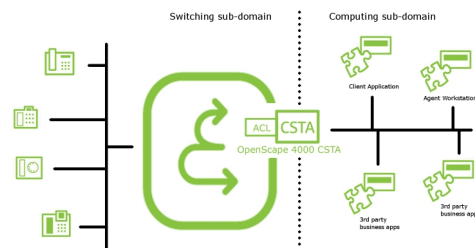
### 2.1 OpenScape 4000 CSTA System Overview

OpenScape is the name of the UNIFY's voice platforms supporting Unified Communication. OpenScape 4000 is a hybrid system supporting analog, TDM and IP based voice communication. OpenScape 4000 CSTA offers a Computer-Supported Telecommunications Application (CSTA) Phase III interface which connects with a CSTA compliant virtual server built in the OpenScape 4000 server.

The provided interface is based on the Standard ECMA-269 and ECMA-323 services for Computer Supported Telecommunications Applications (CSTA) Phase III. CSTA is the European Computer Manufacturing Association's (ECMA) worldwide accepted protocol standard for switch-to-server communications.

Figure 1 shows the general architecture of the CTI solution provided by OpenScape 4000.

The embedded CTI server provides the link between the switching sub-domain and the computing subdomain. To provide call control, the application software within the computing sub-domain must send service requests to the switching sub-domain. To notify the application software of telephony activity taking place within the switching unit, the switching unit sends event messages to the computing sub-domain. Events are sent only for devices monitored by the CTI application.



**Figure 1: Overview of CSTA operational model**

The following table describes the components in more detail:

<b>OpenScape 4000</b>	The OpenScape 4000 is a communication server that can perform telephony functions under application control such as dialing and transferring calls. In addition, the OpenScape 4000 sends call information to computer applications. The OpenScape 4000 has a built in communication interface called OpenScape 4000 CSTA
<b>OpenScape 4000 CSTA</b>	The OpenScape 4000 CSTA is the built in virtual CTI server. The CTI server performs the protocol conversion and allows the OpenScape 4000 to communicate with telephony applications residing in the computing sub-domain.

<b>Computing sub-domain</b>	The computing sub-domain consists of one or more computers. The CTI functions offered by the computers include an application programming interface (API) provider that interacts with the OpenScape 4000 CSTA, exchanging messages. The CTI application accesses the information contained in these messages by an API. The API allows the application to initiate telephony functions such as dialing a number, answering a call, clearing a call, making consultation and conference calls, and to dictate the routing of a call. In addition, the API is used to control extension monitoring so that call activity information is available to applications. This call information enables the application to track the progress of a call and appropriately provide business data information (screen pops) to the agents terminals.
<b>Agent workstation</b>	The agent workstation consists of a digital or analog telephone connected to the OpenScape 4000 and a terminal connected to the host computer. Depending on the application requirements, the agent workstation can be configured as a part of an ACD group.

## 2.2 OpenScape 4000 CSTA System Characteristics

### 2.2.1 TCP/IP Stream Content

All ACSE and ROSE messages sent to and from the OpenScape 4000 CSTA are prefixed by two bytes indicating the length of the ACSE or ROSE message. This length information does not include its own two bytes.

Therefore an ACSE abort message (ABRT)

64 03 80 01 00

for example would be sent as:

00 05 64 03 80 01 00

XML messages contain two additional zero bytes before the length's two bytes.

To keep this documentation simple this length field is not shown in the following chapters.

### 2.2.2 Interface

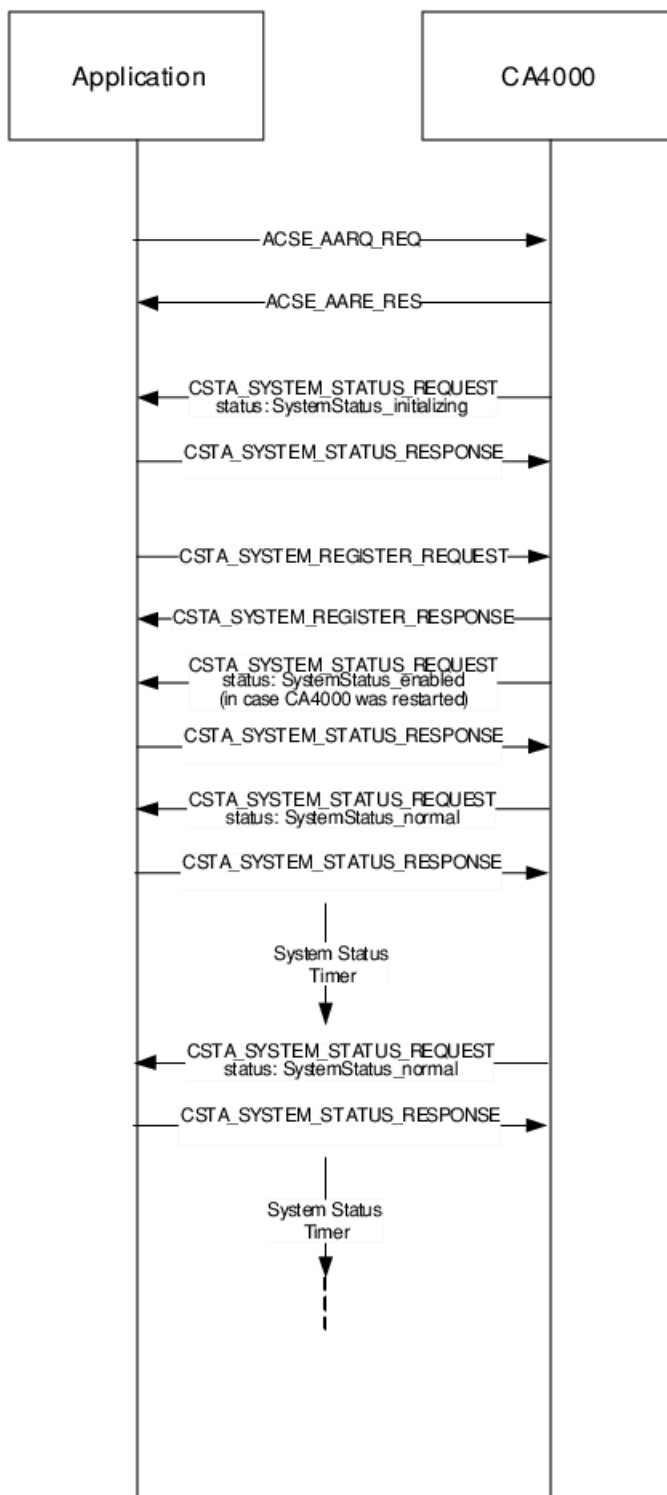
The applications connect to the CA with standard TCP/IP connection.

The supported protocols are: CSTA Phase III ASN.1 and CSTA Phase III XML.

The CSTA Phase III ASN.1 or XML protocol operates in the context of an application association, OpenScape 4000 CSTA uses an explicit association realized through the use of ACSE.

The application starts the initialization sequence by sending an ACSE association request (AARQ) with CSTA association information.

After the ACSE exchange, OpenScope 4000 CSTA sends a System Status message and the application shall respond with a positive acknowledgement.



**Figure 2: Successful connection**

If any problem occurs in the association the AARE has a negative result. The following types of errors can occur in OpenScope 4000 CSTA V6 (please bear in mind that there is no license checking in OpenScope 4000 CSTA any more):

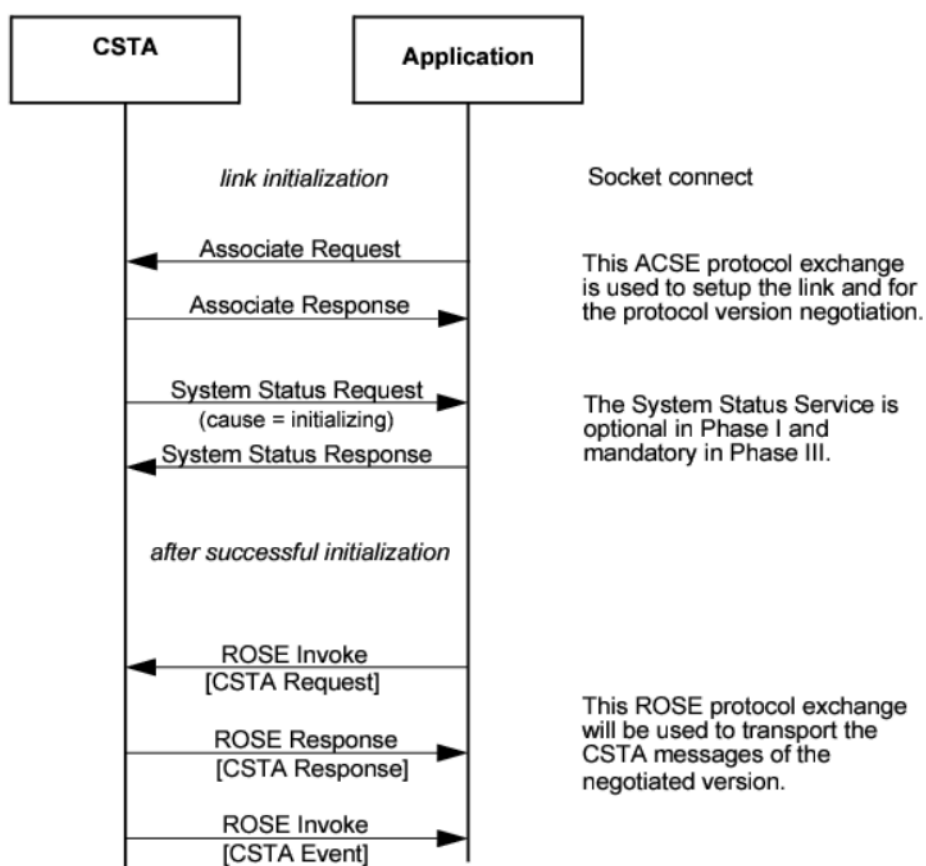
**Table 1: Possible licensing errors in OpenScape 4000 V6**

result-source-diagnostic	implementation-information	reason
authentication-required	-	the authentication value is missing or in wrong format
authentication-failure	"No license available!"	the feature exists, but no more license available
authentication-failure	"CAP-Inside internal licensing error!"	any other case if an error occurs in licensing

## 2.2.3 Application Link Start Up Sequence

For administration of application links please refer to OpenScape 4000 CSTA service documentation.

This chapter describes how an application link, that is configured for the use of ACSE (Association Control Service Element), is set up by the application. The following figure illustrates the necessary message exchange.

**Figure 3: Initialization message exchange**

OpenScape 4000 CSTA expects the following parameters in an ACSE Associate Request (AARQ), defined in the 4th edition of ECMA-285:

## 2.2.4 Exceptional Behaviour and Recovery

The following chapters describe the behaviour of the OpenScape 4000 CSTA in exceptional conditions.

### 2.2.4.1 Hang-up/Disconnect of Application

If the application does not respond to messages any more (e.g. the application is registered for System Status messages but does not respond to the System Status requests by the OpenScape 4000 CSTA) or has disconnected the TCP/IP socket, the OpenScape 4000 CSTA will:

- delete all monitor points set by the application
- delete the System Status Register ID if there had been one assigned to the application link
- delete all Route Register Request IDs for this application link
- end routing dialogues in OpenScape 4000 if a routing dialogue still exists
- delete all I/O Register Request IDs for this application link
- stop I/O data paths in OpenScape 4000 if an I/O data path still exists
- delete Service Cross Reference IDs (e.g. for Snapshot Device Data) if necessary
- delete still outstanding ROSE Invoke IDs for this application link

### 2.2.4.2 Restart of OpenScape 4000 / PBX Link Failure

If OpenScape 4000 disconnects the TCP/IP link (e.g. due to a restart procedure) or does not respond to heartbeat messages (Loopback Test) any more, the OpenScape 4000 CSTA will:

- send a Monitor Stop request to all applications that have set monitor points
- send a System Status message (indicating that message got lost) to all applications that have registered for System Status messages
- will not delete any Route Register Request IDs for this application link
- send a Route End Request to all applications that are still within a routing dialogue
- will not delete any I/O Register Request IDs for this application link
- send I/O Stop Data Path requests to all applications to which there exists an I/O data path
- delete still outstanding ROSE Invoke IDs

### 2.2.4.3 Restart of the OpenScape 4000 CSTA

If the OpenScape 4000 CSTA is resetted (e.g. through administration), prior to shutting down it will:

- send a Monitor Stop request to all applications that have set monitor points
- send a System Status message (indicating that the OpenScape 4000 CSTA is now disabled) to all applications that have registered for System Status messages
- delete any Route Register Request IDs for this application link

- send a Route End Request to all applications that are still within a routing dialogue
- delete any I/O Register Request IDs for this application link
- send I/O Stop Data Path requests to all applications to which there exists an I/O data path
- delete still outstanding ROSE Invoke IDs

### 2.2.5 System Status

The application can register (System Register) to get informed (frequently) about the system state of OpenScape 4000 and the OpenScape 4000 CSTA.

The following System Status messages are sent by OpenScape 4000 CSTA in certain conditions:

#### **'initializing'**

The system is re-initializing or restarting.

The OpenScape 4000 CSTA sends this status

- on start-up, i.e. when the TCP/IP connection with OpenScape 4000 is established,
- after a reset of the OpenScape 4000 CSTA, i.e. the TCP/IP connection with OpenScape 4000 is re-established,
- after re-establishing the TCP/IP connection with OpenScape 4000 after a link failure.

#### **'enabled'**

Requests and responses are enabled, usually after a disruption or restart.

The OpenScape 4000 CSTA sends this status

- when the layer 7 to OpenScape 4000 is up.

#### **'messageLost'**

This status indicates that one or more service requests, responses, or event reports may have been lost.

The OpenScape 4000 CSTA sends this status

- if messages got lost on the PBX link or within OpenScape 4000,
- if the OpenScape 4000 CSTA was unable to map the event stream of OpenScape 4000,
- if the application blocked the receiving of TCP/IP message (e.g. due to overload).

#### **'disabled'**

This status indicates that all active cross reference and registration IDs, for which the OpenScape 4000 CSTA did not send a cancelation message, have been disabled.

The OpenScape 4000 CSTA sends this status

- if the layer 7 to OpenScape 4000 is down,
- if the event stream of OpenScape 4000 got disabled by configuration.

**'overloadReached'**

The system has reached an overload condition and may take action to shed load.

The OpenScape 4000 CSTA sends this status

- if an overload condition in OpenScape 4000 occurs.

**'overloadRelieved'**

The system has determined that the overload condition has passed and normal application operation may resume.

The OpenScape 4000 CSTA sends this status

- if the overload condition in OpenScape 4000 ended.

**'normal'**

This value can be sent at any time to indicate that the status is normal.

The OpenScape 4000 CSTA sends this status

- frequently as a heartbeat mechanism.

The OpenScape 4000 CSTA does not support the system status values 'partiallyDisabled' and 'overloadImminent'.

The application does not need to register for system status services if it wants to send System Status requests. The OpenScape 4000 CSTA will only accept system status value 'normal' in System Status service request from the application. All other values will be answered with a return error.

System Status messages are also part of the ACSE link association. See [Figure 3](#).

## 2.2.6 Correspondence between ACSE-licensing

In OpenScape 4000 V6 CSTA there are three license modes available and they differ in the number of possible monitor points. The 3 modes are: "no license", "100 clients" and "unlimited". For the 2 later modes a license is required which is checked at the application's connection (Product ID: L30001-B39-A1, version: "V3.0", feature IDs: "100\_Clients" and "Unlimited"). The maximum number of allowed monitor points in the different modes: No license /10, 100 clients/100, Unlimited/100000. Monitor points are counted per application connection and if the same device has multiple monitor points each of them are counted.

Licensing can be set through the web based configuration interface on the application configuration dialog in the "License type" option. When the license mode is changed the application connection is automatically closed by the cbdriver and the application has to reconnect.

Please use the "no license" option only for test purpose.

In OpenScape 4000 V7 CSTA the above mentioned licensing has been removed.

## 2.3 Application Examples

OpenScape 4000 V10 can support a variety of business applications. The following paragraphs describe typical applications.

### 2.3.1 Intelligent Answering

Intelligent answering allows a computer system application to display customer or business data on the screen of the agent receiving an incoming call. To do this, the OpenScape 4000 sends the public network-provided information, such as automatic number identification (ANI), where the caller is calling from, or dialed number identification service (DNIS), the number the caller dialed, information, to the computer system environment and hence the application in the event stream.

- **ANI**

The computer system application can use ANI information as an index to look up customer data in a database and display relevant information, such as customer records, on the agent's screen.

For example, an emergency medical service provider can use ANI to identify callers. If a caller dials 911, the OpenScape 4000 identifies the call originator's telephone number and sends this information to the computer system application. The application uses this number to look up other information about the caller (such as name and address, or whether there are hazardous materials on-site) and displays it on the agent's screen.

- **DNIS**

The computer system application uses DNIS information to display the applicable screen on the agent's workstation.

For example, a telesales company could have several inbound 800 numbers and wish to provide a distinct greeting based on which number the caller dialed. The agent would be prompted to answer "Good afternoon... thank you for calling Company Z," or "Thank you for calling XYZ."

The computer system application also can use ANI or DNIS information to match the calling or called number to a database of skilled individuals or groups so that calls can be routed to the correct person or group. The call could be routed to the agent who last interacted with the customer, for example, promoting continuity of service and increasing customer satisfaction. This function is often referred to as host-based or skills-based routing. This is a significant enhancement to the ACD capabilities within the switching sub-domain.

The following example shows how events can be used to support intelligent answering applications.

Assume that Acme, Inc. is a manufacturer of a product line of power supplies. If any of the power supplies require service, the customer is instructed to call 1-800-WEFIXIT. Acme, Inc. has a call center to handle these service calls. Acme, Inc. uses the ACD feature of its PBX to manage distributing the calls and has a database program called DATA\_BANK to access product and customer account information from its main computer. Today, without CTI, when customers call Acme, Inc. to obtain service for power supplies and are connected to agents, the agents must first ask the customers for their names or account numbers, then enter the information into computer terminals, and

finally, wait for DATA\_BANK to bring up the customer records. This is not intelligent answering.

In contrast, with OpenScape 4000 CSTA, a CTI application can access DATA\_BANK and retrieve the customer record before the call is answered by the agent. Using OpenScape 4000 CSTA, agents do not need to ask customers for their names or account numbers, nor do they need to enter this information on computer terminals. With OpenScape 4000 CSTA, if a customer is identified by ANI (automatic number identification) or if this information has already been requested by an interactive voice response (IVR) unit, the OpenScape 4000 CSTA-enabled CTI application automatically accesses DATA\_BANK and presents the associated customer data to an agent along with the call. The agent receives the customer information without having to request it or enter it on the computer terminal. The agent can verify the customer data that appears on the agent's computer screen by asking the customer to confirm the data. This is intelligent answering.

To provide intelligent answering, the CTI application must be able to check the caller's identity to determine to which agent to assign the call. Assuming that the route control groups (RCGs) and the agents are monitored, call information (including ANI) is made available to the application in the form of events.

The key events enabling this application are:

- Delivered event. The call is received by ACD in an RCG.
- Queued event. The call is queued to an ACD group. There will be one queued event for every group for which the call is queued.
- Diverted event. The call has left ACD.
- Delivered event. The agent's telephone rings.
- Established event. The agent answers the call.
- Connection Cleared event. The call is ended by either party.

In our example, the following is the sequence in which these events occur:

- 1) The first *delivered event* establishes the caller's identity to the CTI application program and provides the ACD number being called. Assuming that Acme, Inc. subscribes to the ANI service, the caller is identified by an ANI number.

This ANI number is provided as a parameter in the delivered event. The number that is called (DNIS) also is identified as a parameter in the delivered event.

- 2) If the call is queued in ACD, a *queued event* is provided to the CTI application.

This queued event has the same call ID as the delivered event so that an application will know that it is associated with the same call.

- 3) When the call leaves ACD for an available agent:
  - a) A *diverted event* is sent to indicate that the call has left ACD.
  - b) A *delivered event* is sent to indicate that the call has been delivered to an agent and that the agent's telephone is ringing.
- 4) When the call is delivered to an available agent, the call has the same call ID and contains the ANI of the caller in the *delivered event*. When the target agent has been identified, the CTI application program can access DATA\_BANK, then retrieve and present the customer data to the agent.
- 5) When the agent answers the call, an *established event* is sent to the application. At this time, the screen of information is already displayed. (Data lookup is done while the agent's telephone is ringing.)

- 6) Finally, the call ends when the caller or the agent hangs up the telephone. At this point, *connection cleared events* are sent to the application program.

## 2.3.2 Coordinated Voice and Data Transfer

Coordinated voice and data transfer allow customer data to be transferred to a second agent's screen when a call is transferred from one agent to another. If the customer gives information to an agent (live or automated), the information is transferred automatically with the call, eliminating the need for each agent handling a call to request the same information. This promotes continuity of service and makes your call center agents seem like a coordinated team to your external caller.

## 2.3.3 Routing Services

The OpenScape 4000 V10 Routing services allow a computer system application to influence the routing of an a call *before* it is routed by the OpenScape 4000. Based on different criteria, the routing services enable the application to have exclusive control of the call to:

- Route a call to an alternate destination (internal or external).
- Reject a call (return busy to the caller) if the call center is too busy, thereby reducing toll charges because answer supervision is not returned prior to the reject.
- Redirect the call to a predefined alternate switching sub-domain using network services, such as the Alternate Destination Call Redirection (ADCR) Service.
- Pass control to the communication server's ACD.

A computer system application can base this routing determination on many different criteria, such as:

- The activity of a single call center (number of available agents versus average queue length).
- The activity of an entire network of call centers (individually monitored by the application) to support network load balancing applications.
- ANI and DNIS information supplied with the incoming call.

As an example, a routing determination may be based upon the information supplied with the call (ANI or DNIS or both). The computer application can base its routing determination on information in its database that is associated with the caller. If the caller had called previously and is identified by the application as a very important customer, the application can route the call to a specialist or to the agent who handled the call previously.

- Any other decision criteria that is accessible to the computer application, for example, sales statistics. A call can be routed to a qualified agent, rather than the first available one, thereby maximizing revenue.

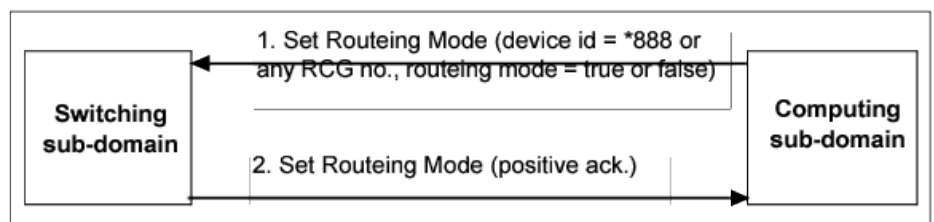
The routing features are used before the communications server sends alert indication (ringback) to the caller or answer supervision to the network. After the call has entered ACD, the Divert Call service and normal ACD switching sub-domain routing may be used to direct the ACD call.

### 2.3.3.1 Get/Set Routeing Mode

Figure 4 on page 31 illustrates how an application can set the routeing mode of all RCGs or individual RCG. The numbers shown in Figure 4 on page 31 correspond to the following steps:

- 1) The application issues a Set Routeing Mode request for an individual RCG by specifying RCG number or for all RCG by specifying \*888. For definition of Set Routeing Mode service request.
- 2) The switching sub-domain responds with a positive acknowledgment message.

The Set Routeing Mode request is typically sent by the application after it comes on-line. When a call arrives at an RCG, the ACD routing table (ART) delays ringback to the caller until a routing decision is made.



**Figure 4: Set Routeing Mode**

The Get Routeing Mode request can be sent by the application to know whether it will receive routeing request from the switching sub-domain or not.

### 2.3.3.2 Route Request and Route Select Dialog

This example continues the previous one. The switching sub-domain shown in Figure 5 on page 32 issues a Route Request message and the application responds to the request by sending a Route Select message specifying an alternate destination. Notice that the switching sub-domain allocates a routing cross reference identifier (XREFID) that associates all of the routing messages with the original call.

The numbers shown in Figure 5 correspond to the following steps:

- 1) A call has arrived at an RCG on which Routeing Mode has been set to TRUE or has been triggered (refer to set routeing mode example). The call was assigned to an ART that contains a delay ringback step. Because this ART is controlled by the RCG that was triggered, the ART is called a *triggered ART*.
- 2) The switching sub-domain sends a Route Request message to the application. Because, in this example, the network passed ANI information, the switching sub-domain sends this information along in the Route Request message.

The application receives the messages and checks the ANI information in its database and determines that the ANI is associated with a customer that had called earlier. The previous agent had noted in the transaction record that the customer may be ready to place a large order. The application determines the call needs to be handled by a special group of agents associated with ACD number 1234.

- 3) The application sends a Route Select message specifying destination 1234 to the switching sub-domain.

The switching sub-domain receives the message and determines that the number is an ACD number. This ACD number will be resolved to an ART that will queue the call to an ACD group if there are no agents available. The switching sub-domain then:

- Terminates the routing dialog
- Diverts the call to the specified number

- 4) Finally, the switching sub-domain sends a Route End message to the application indicating call routed. An application should check the cause in the Route End message to ensure that the call was successfully routed. Route End could also indicate a time-out waiting for the application to respond.

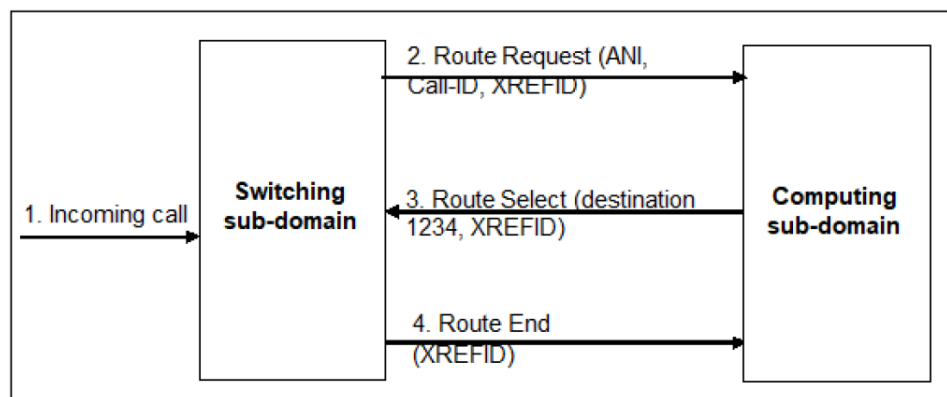


Figure 5: Route Request and Route Select Dialog

### 2.3.3.3 Route Reject Dialog

In this example (Figure 6), the switching sub-domain has determined that the local call center (CC1) is too busy and that another call center (CC2) has available agents. In this instance, the system was designed to handle this type of overflow using the network Alternate Destination Call Redirection (ADCR) service.

If the application rejects the call, it will be sent to the pre configured alternate switching sub-domain. This optimizes trunks, because there is no trunk connection needed between CC1 and CC2.

The numbers shown in Figure 6 correspond to the following steps:

- 1) The switching sub-domain receives an incoming call associated with an RCG on which Routeing Mode has been set to TRUE.
- 2) The switching sub-domain sends a Route Request message to the application, which receives the message and determines that the call should be redirected to CC2.
- 3) The application sends a Route Reject message to the switching sub-domain.
- 4) The switching sub-domain then sends a Route End message to the computer application terminating the routing dialog.
- 5) The switching sub-domain then sends a special ISDN cause code over the network trunk to enable the ADCR service for that call. The call is redirected by the network to the other switching sub-domain.

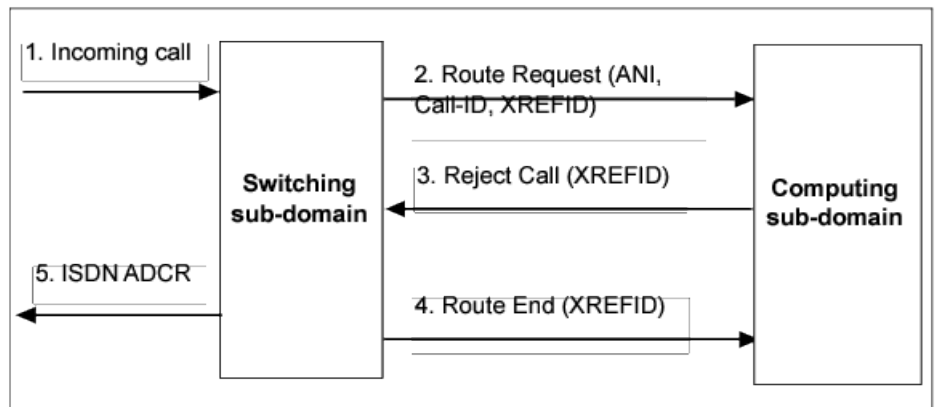


Figure 6: Reject Call

### 2.3.3.4 Route End Dialog

Figure 7 shows an example of a routing dialog terminated by the application. In this example, the switching sub-domain issues a Route Request message and, because the application will not route the call, it responds to the Route Request with a Route End message.

The numbers shown in Figure 7 correspond to the following steps:

- 1) The switching sub-domain receives an incoming call associated with an RCG on which Routeing Mode has been set to TRUE.
- 2) The switching sub-domain sends a Route Request message to the application. Because, in this example, the network passed ANI information, the switching sub-domain includes the ANI information in the Route Request message.

The application checks the ANI information against its database and detects that the ANI is associated with an unknown customer. For that reason the application does not process the call, and the system is designed so these types of calls fall through to normal ACD. (Routing services are exercised before ACD.)

- 3) The application sends a Route End message to the switching sub-domain to indicate that it will not influence the routing.

The switching sub-domain receives the message, terminates the routing dialog, continues the call through ART processing, and queues the call to an agent group.

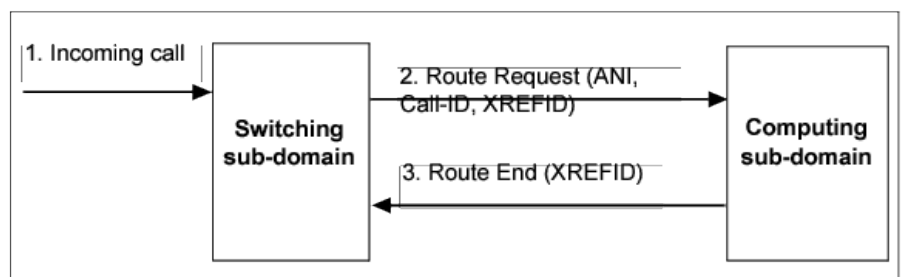


Figure 7: Route End

### 2.3.3.5 Re-Route Request Dialog

In this example, the switching sub-domain issues a Route Request message and the application responds to the request by sending a Route Select message specifying a particular agent extension. However, the switching sub-domain detects that the agent is busy and issues a Re-Route Request message to the application. The application determines that the call could be sent to another agent and specifies the destination to that agent in another Route Select message. This time the specified agent is free and the call goes to that agent. [Figure 8](#) illustrates the Re-Route Request dialog.

The numbers shown in [Figure 8](#) correspond to the following steps:

- 1) The switching sub-domain receives an incoming call associated with an RCG on which Routeing Mode has been set to TRUE.
- 2) The switching sub-domain sends a Route Request message to the application. Because, in this example, the network has passed ANI information, the switching sub-domain includes the ANI information in the Route Request message.

The application checks the ANI information in its database and determines that the ANI is associated with a very important customer. The application determined that the agent at extension 1234 handled this customer previously.

- 3) The application sends a Route Select message specifying destination 1234 to the switching sub-domain. The switching sub-domain determines that this number is busy. The switching sub-domain resets the routing timer to give the application additional time to respond to a second request for destination message.
- 4) The switching sub-domain sends a Re-Route Request message to the application and waits for another response from the application.
- 5) The application sends a Route Select message specifying another destination, 5678, to the switching sub-domain. The switching sub-domain determines that 5678 is available.
- 6) The switching sub-domain routes the call to 5678. The switching sub-domain terminates the routing dialog and sends a Route End message to the application.

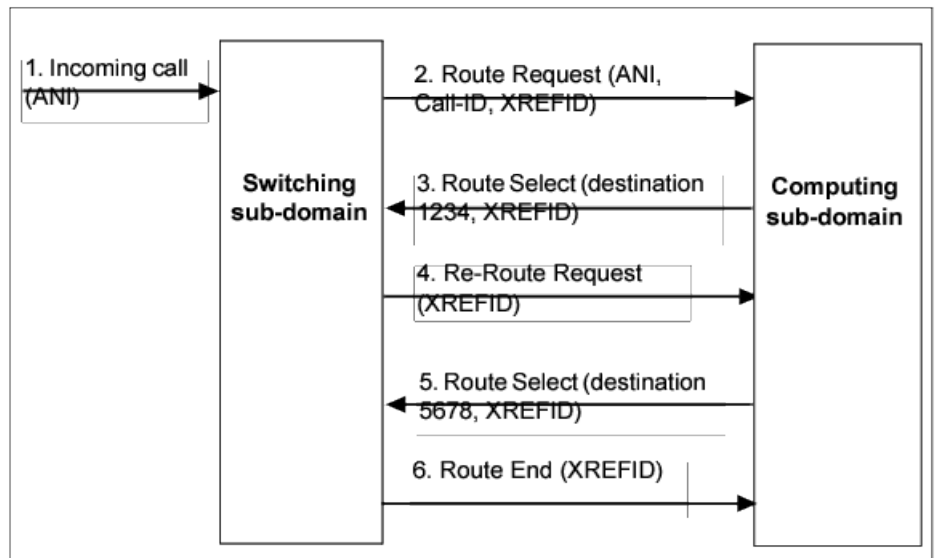


Figure 8: Re-Route Request

## 2.3.4 Enhanced Business Statistics

To improve the efficiency and effectiveness of your call center, measurements are required. Typical switching sub-domain has information on the maximum number of callers in queue, the average speed of answer, and so on. The business application has information on the number of units sold. Using CTI, you can correlate these statistics to determine average revenue per agent, and so on, to optimize your operation based on your performance goals.

## 2.3.5 Automated Outbound Dialing

Likewise, you can monitor incoming traffic and initiate outbound dialing campaigns during periods of low activity. The automated outbound dialing feature allows a business application to automatically place a call to a customer on behalf of an agent. The agent can scroll through a list of customers selected by the application, then press a function key to call a particular customer. This is referred to as *preview dialing*. This automation eliminates the time required to dial the number, and prevents dialing errors.

## 2.4 ACSE messages

### 2.4.1 ACSE AARQ

Example SNACC C++ code:

```

AARQ_apdu *aarq=new AARQ_apdu();

// application-context-name
aarq->application_context_name.Set(1, 3, 12, 0, 218);
    
```

```
// cSTAVersion in user-information
aarq->user_information = new Association_information();
Association_information* user_information = aarq->user_information;
// Alloc new extInformationForCSTA element, put it at end of
// SEQUENCE, and return the element
ExtInformationForCSTA* extInformationForCSTA = user_information->Append();
extInformationForCSTA->direct_reference = new AsnOid(1, 3, 12, 0, 285, 200);

// cSTAVersion in encoding (EncodingCsta)
extInformationForCSTA->encoding = new EncodingCsta();
EncodingCsta* encoding = extInformationForCSTA->encoding;
encoding->choiceId = EncodingCsta::single_ASN1_typeCid;
encoding->single_ASN1_type = new ACSEUserInformationForCSTA();
ACSEUserInformationForCSTA* single_ASN1_type =
encoding->single_ASN1_type;
single_ASN1_type->choiceId =
ACSEUserInformationForCSTA::newDefinitionCid;
single_ASN1_type->newDefinition =
new NewACSEUserInformationForCSTA();
NewACSEUserInformationForCSTA* newDefinition =
single_ASN1_type->newDefinition;
// Set CSTAVersion:
// versionFive=ECMA 285 2nd, versionSix=ECMA 323
newDefinition->cSTAVersion.Set(8); // set bitLen
newDefinition->cSTAVersion.SetBit(CSTAVersion::versionFive);
```

If the application uses CSTA III XML encoding (instead of CSTA III ASN.1 encoding) in the last line `CSTAVersion::versionFive` has to be replaced with `CSTAVersion::versionSix`. These are the mandatory fields to be filled in, but some optional field can be used as well. For example the calling-authentication-value field can hold any data to be backward compatible with 'CAP Inside', but the information given there doesn't affect the connection establishment.

```
AARQ
{ -- SEQUENCE --
protocol-version -- void --,
application-context-name {1 3 12 0 218},
called-AP-title -- void --,
called-AE-qualifier -- void --,
called-AP-invocation-identifier -- void --,
called-AE-invocation-identifier -- void --,
calling-AP-title -- void --,
calling-AE-qualifier -- void --,
calling-AP-invocation-identifier -- void --,
calling-AE-invocation-identifier -- void --,
sender-acse-requirements -- void --,
mechanism-name -- void --,
calling-authentication-value -- void --,
implementation-information -- void --,
,
user-information { -- SEQUENCE/SET OF --
{ -- SEQUENCE --
direct-reference {1 3 12 0 285 200} indirect-reference --
void --,
data-value-descriptor -- void --,
```

```

encoding single-ASN1-type newDefinition { -- SEQUENCE --
cSTAVersion '08'H -- BIT STRING bitlen = 8 --,
cSTAFunctionsRequiredByApplication -- void --,
cSTAFunctionsThatCanBeSupplied -- void --,
cSTAPrivateDataVersionList -- void --

}
}
}

}

```

The ASN.1 BER encoded AARQ (the first 2 bytes are the length prefix) for CSTA III ASN.1:

```

00 20 60 1e a1 07 06 05 2b 0c 00 81 5a be 13 28 11 06 07 2b 0c 00 82 1d 81 48 a0
06 a0 04 03 02 00 08

```

The same for CSTA III XML, the only difference is in the last byte:

```

00 20 60 1e a1 07 06 05 2b 0c 00 81 5a be 13 28 11 06 07 2b 0c 00 82 1d 81 48 a0
06 a0 04 03 02 00 04

```

## 2.4.2 ACSE AARE

Example ASN.1 ACSE AARE (BER encoded and decoded), request accepted:

```

61 2a a1 07 06 05 2b 0c 00 81 5a a2 03 02 01 00 a3 05 a1 03 02 01 00 be 13 28 11
06 07 2b 0c 00 82 1d 81 48 a0 06 a0 04 03 02 00 08

```

```

AARE
{ -- SEQUENCE --
protocol-version -- void --,
application-context-name {1 3 12 0 218},
result 0,
result-source-diagnostic acse-service-user 0
responding-AP-title -- void
--,
responding-AE-qualifier -- void --,
responding-AP-invocation-identifier -- void --,
responding-AE-invocation-identifier -- void --,
responder-acse-requirements -- void --,
mechanism-name -- void --,
responding-authentication-value -- void --,
implementation-information -- void --,
,
user-information { -- SEQUENCE/SET OF --
{ -- SEQUENCE --
direct-reference {1 3 12 0 285 200} indirect-reference --
void --,
data-value-descriptor -- void --,
encoding single-ASN1-type newDefinition { -- SEQUENCE --
cSTAVersion '08'H -- BIT STRING bitlen = 8 --,
cSTAFunctionsRequiredByApplication -- void --,
cSTAFunctionsThatCanBeSupplied -- void --,
cSTAPrivateDataVersionList -- void --

}

}
}

```

```
}  
}  
  
}
```

Example ASN.1 ACSE AARE (BER encoded and decoded), feature not available:

```
61 2d a1 07 06 05 2b 0c 00 81 5a a2 03 02 01 01 a3 05 a1 03 02 01 0d 9d 16 46 65  
61 74 75 72 65 20 6e 6f 74 20 61 76 61 69 6c 61 62 6c 65 21  
AARE  
{ -- SEQUENCE --  
protocol-version -- void --,  
application-context-name {1 3 12 0 218},  
result 1,  
result-source-diagnostic acse-service-user 13  
responding-AP-title -- void  
--,  
responding-AE-qualifier -- void --,  
responding-AP-invocation-identifier -- void --,  
responding-AE-invocation-identifier -- void --,  
responder-acse-requirements -- void --,  
mechanism-name -- void --,  
responding-authentication-value -- void --,  
,  
implementation-information  
'46656174757265206e6f7420617661696c61626c6521'H  
-- "Feature not available!" -- user-information -- void --  
}
```

## 2.5 Restrictions

Restrictions when not all devices are monitored

If not every device is monitored, who is participates in the scenario, then sometimes we cant get all necessary information. E.g.:

Call Back Call Related

D1 calling

D2 called - only this device is monitored

The restriction for monitoring can be seen in the EventFlow table, step 8 (The switching function reserves the CallBack destination (D2).):

Failed Event (from D2 device)

calling party: notKnown

called party: notKnown

These values are notKnown because we cant decide them from the provided state events.

## 3 OpenScape 4000 CSTA Services

This chapter describes the CSTA III services provided by OpenScape 4000 CSTA in cooperation with OpenScape 4000 and includes:

- General Notes ([Section 3.1, "General Notes"](#))
- A description of each CSTA service ([Section 3.2, "Services Descriptions"](#))
- A list of supported devices ([Table 2 "Services supported by OpenScape 4000 devices" on page 45](#))

### 3.1 General Notes

The following general items apply to services:

- 1) Stations may or may not have auto-answer capability or other feature capabilities. Station types are grouped according to their capabilities as follows:
  - a) Auto-answer devices. These are digital telephones with speakers controlled by the switching unit<sup>1</sup> and/or an activated headset<sup>2</sup>. (The National ISDN telephone Optiset NI 1200 is assumed to have an activated headset as stipulated in the Installation and Problem Determination Guide.)
  - b) Non-auto-answer devices. These are analog telephones [including wireless telephones, off premises stations<sup>3</sup> (OPS) and interactive voice response<sup>4</sup> (IVR) units] and digital telephones without switching unit controlled speakers<sup>1</sup> and without an activated headsets<sup>2</sup>.
  - c) Unsupported devices. These are non-voice units such as fax machines and modems, the national ISDN telephone Lodestar, and analog devices connected to digital telephones by terminal adapters (for example, TA-a/b).
- 2) For Call Control services, if the Device ID specified in the request refers to an extension number that appears on more than one device (multiple appearance), the Call Control service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active (that is, the Connection State is *connected*) on that device. If these conditions are not met, a negative response is returned.
- 3) For Logical Device services and Status Reporting services, if the Device ID specified in the request refers to an extension number that appears on more than one device (multiple appearance), the service applies to all appearances of that device as well as the primary device.

---

<sup>1</sup> To be controlled by the switching unit, hands-free operation must be configured for the device.

<sup>2</sup> To be activated, AICS (automatic incoming call signaling) must be turned on at the device.

<sup>3</sup> An OPS can be configured on a regular analog port (SLMA-OPS) or as a T1-OPS.

<sup>4</sup> An IVR can also be configured as an SLMA-OPS or as a T1-OPS.

## 3.2 Services Descriptions

The following descriptions provide the following types of information about each service:

- An explanation of the service based on ECMA 269
- A table listing the supported parameters of the request message.
  - The original parameter name and content
  - An indication of whether the parameter is required or optional
  - A description of each parameter
    - based on ECMA 269 where improper or not relevant statements are crossed out in the text (like this)
    - additions are marked with OpenScape 4000:
- Successful response messages are listed in a service response parameter table (only if service specific parameters are provided in the message).
  - The original parameter name and content
  - An indication of whether the parameter is required or optional
  - A description of each parameter
    - based on ECMA 269 where improper or not relevant statements are crossed out in the text (like this)
    - additions are marked with OpenScape 4000:
- Unsuccessful responses are listed in an error code table.
  - Error Category and Value
  - Possible Causes
- Usage notes providing important additional information about the services.
  - Supported devices
  - OpenScape 4000 specific behaviour
  - Changes and additions to functional requirements in ECMA 269
  - Miscellaneous Characteristics

## 3.3 Services Summary

**Table 2: Services supported by OpenScape 4000 devices**

CSTA ASN.1														CSTA XML
Services	Description	Station Analog	Station Digital	SIP only	SIP- Classi from OpenScape 4000 V7R2 V5	CMI	Phant	Functi (ISDN)	Attend Conso	Gener Attend	RCG	Hunt Group	Trunk	same device types valid in case supported (marked with x)
Capability Exchange Services														
Get Logical Device Information	Obtains the current set of logical device information for a given device identifier.		x <sup>5</sup>	X		x <sup>6</sup>	X			X		X		-

<sup>5</sup> Keysystem (multi-line device) only

CSTA ASN.1														CSTA XML
Services	Description	Station Analog	Station Digital	SIP only	SIP-Classic from OpenScape 4000 V7R2 V5	CMI	Phantom	Function (ISDN)	Attend Consc	Gener Attend	RCG	Hunt Group	Trunk	same device types valid in case supported (marked with x)
Get Physical Device Information	Obtains the current set of physical device information for a given device identifier.	X	X	X	X	X	X	X	X				X	-
Get Switching Function Capabilities	Obtains the current set of capabilities for the entire switching function.													-
Get Switching Function Devices	Obtains the devices in the application working domain (i.e. devices that can be controlled and/or observed).	X	X	X	X	X	X		X	X	X	X	X	X
Switching Function Devices	Provides the actual list of devices in the application working domain (i.e. devices that can be controlled and/or observed).	X	X	X	X	X	X		X	X	X	X	X	X
System Status Services														
Change System Status Filter	Changes the system status filter options for a current system registration.													-
System Register	Registers the computing function for system services with the switching function.													X
System Register Cancel	Unregisters the computing function for system services with the switching function.													-
Request System Status	Request to query the system status of the function receiving the request (bi-directional).													-
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.													X
Switching Function Devices Changed	Request that reports that information associated with the current set of devices that can be controlled and observed in the switching sub-domain (available via the Get Switching Domain Devices service) has changed.													-
Monitor Services														
Change Monitor Filter	Modifies the event filter for an existing monitor.	X	X	X	X	X	X		X	X	X	X	X	-
Monitor Start	Initiates an event monitor on a specified device or call.	X	X	X	X	X	X		X	X	X	X	X	X
Monitor Stop	Terminates an existing monitor.	X	X	X	X	X	X		X	X	X	X	X	X
Snapshot Services														

## OpenScape 4000 CSTA Services

CSTA ASN.1															CSTA XML
Services	Description	Station Analog	Station Digital	SIP only	SIP- Classi from OpenS 4000 V7R2 V5	CMI	Phanti	Functi (ISDN	Attend Conso	Gener Attend	RCG	Hunt Group	Trunk	same device types valid in case supported (marked with x)	
Snapshot Call	Provides information about the devices participating in a specified call.	X	X	X	X	X	X		X	X	X	X	X	X	
Snapshot CallData	Provides Snapshot Call information in segmented messages	X	X	X	X	X	X		X	X	X	X	X	X	
Snapshot Device	Provides information on the status of calls at a specific device.	X	X	X	X	X	X		X	X	X	X	X	X	
Snapshot DeviceData	Provides Snapshot Device Information in segmented messages.	X	X	X	X	X	X		X	X	X	X	X	X	
Call Control Services															
Accept Call	Accept callis supported from V6 R1. Causes an offered call to transition to the Ringing or Entering Distribution mode of the alerting state.		X	X		X	X							X	
Alternate Call	Places an existing call on hold and then retrieves a previously held or alerting call at the same device.		X	X <sup>7</sup>		X	X							X	
Answer Call	Answers a call that is ringing, queued, or being offered to a device.		X	X <sup>8</sup>			X							X	
Call Back Call-Related	Allows a computing function to request that an originally called device return a call to the original calling device.		X	X		X	X							X	
Clear Connection	Releases a specific device from a call.	X	X	X	X <sup>9</sup>	X	X				X		X	X	
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.	X	X	10X		X	X							X	
Consultation Call	Places an existing active call at a device on hold and initiates a new call from the same device.	X	X	X		X	X							X	
Deflect Call	Deflects a call to another device.	X	X	X	X <sup>11</sup>	X	X				X	X	X	X	
Dial Digits	Dials a digit sequence for a call that has already been initiated.	X	X			X	X							X	
Hold Call	Places a specific connection on hold.	X	X				X							X	

<sup>7</sup> Supported only if consultaion was initiated previously va CSA Consultation Call and not from the SIP client

<sup>8</sup> SIP client must support procedure 9.18.1.1 defined in OSCAR SIP client spec. <https://www.g-dms.com/livelink/livelink.exe/Open/44767633>

<sup>9</sup> From OpenScape 4000 V6 only

<sup>10</sup> Supported only if consultaion was initiated previously va CSA Consultation Call and not from the SIP client

<sup>11</sup> From OpenScape 4000 V6 only

CSTA ASN.1														CSTA XML
Services	Description	Station Analog	Station Digital	SIP only from OpenScape 4000 V7R2 V5	SIP-Classic from HiPath 4000 V5	CMI	Phant	Function (ISDN)	Attend Consc	Gener Attend	RCG	Hunt Group	Trunk	same device types valid in case supported (marked with x)
Make Call	Establishes a call between two devices.	X	X	X <sup>12</sup>		X	X					X		X
Make Predictive Call	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.										X			X
Reconnect Call	Clears an existing connection and then connects a previously held connection at the same device.	X	X	X <sup>13</sup>		X	X							X
Retrieve Call	Connects to a call that had previously been placed on hold.	X	X											X
Single Step Transfer Call	Replaces a device in an existing call with another device.	X	X	X	X <sup>14</sup>	X	X						X	X
Transfer Call	Transfers a held call to the consulted party.	X	X	X <sup>15</sup>		X	X							X
Call Associated Feature Services														
Generate Digits	Generates DTMF or rotary digits on behalf of a connection in a call.	X	X			X	X							X
Generate Telephony Tones	Generates a specified telephony tone on behalf of a connection in a call.	X	X								X			X
Send User Information	Sends user-to-user information from a specified connection in a call.	X	X	X		X	X		X	X	X	X	X	-
Routing Registration Services														
Route Register	Registers the computing function as a routing server for a specified routing device or for the entire switching function.										X			-
Route Register Cancel	Unregisters the computing function as a routing server.										X			-
Routing Services														
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.										X			-
Route End	This service ends a routing dialogue.										X			-

<sup>12</sup> SIP client must support procedure 9.18.1.1 defined in OSCAR SIP client spec. <https://www.g-dms.com/livelink/livelink.exe/Open/44767633>

<sup>13</sup> Supported only if consultaion was initiated previously va CSA Consultation Call and not from the SIP client

<sup>14</sup> From OpenScape 4000 V6 only

<sup>15</sup> Supported only if consultaion was initiated previously va CSA Consultation Call and not from the SIP client

## OpenScape 4000 CSTA Services

CSTA ASN.1														CSTA XML
Services	Description	Station Analog	Station Digital	SIP only	SIP- Classi from OpenScape 4000 V7R2	CMI	Phanti	Functi (ISDN	Attend Conso	Gener Attend	RCG	Hunt Group	Trunk	same device types valid in case supported (marked with x)
Route Reject	This service is sent to the switching function during a routing dialogue to indicate that a call should be returned to the network for alternate routing.										X			-
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.										X			-
Route Select	This service is used by the computing function to provide the destination requested by a previous Route Request or Re-Route request.										X			-
Physical Device Feature Services														
Get Message Waiting Indicator	Get the message waiting status at a specified device.		X	X										
Set Display	Set the display on a specified device.		X											X
Set Lamp Mode	Set the lamp mode status of a specified button on a device.	X	X			X								X
Set Message Waiting Indicator	Set the message waiting status of a specified device.		X	X										X
Get Microphone Mute	Gets mute status of a microphone associated with an auditory apparatus at a specified device.		X											X
Set Microphone Mute	Sets mute status of a microphone associated with an auditory apparatus at a specified device.		X											X
Button Press	Button press of a specified button on a device		X											-
Logical Device Feature Services														
Cancel Call Back	Cancels a previous (or all) Call Back feature at a device.		X	X		X	X							X
Get Agent State	Get the agent state of a specified device.	X	X			X	X						X	X
Get Do Not Disturb	Get the do not disturb status of a specified device.	X	X	X <sup>16</sup>		X			X					X
Get Forwarding	Get the forwarding status of a specified device.	X	X	X <sup>17</sup>		X	X			X		X		X
Get Routing Mode	Get the routing mode at a specified device.										X			X
Set Agent State	Set the agent state of a specified device.	X	X			X	X							X
Set Do Not Disturb	Set the do not disturb status of a specified device.	X	X	X		X								X

<sup>16</sup> The requests get/set system features, not SIP client features

<sup>17</sup> The requests get/set system features, not SIP client features

CSTA ASN.1														CSTA XML
Services	Description	Station Analog	Station Digital	SIP only from OpenScape 4000 V7R2 V5	SIP-Classic from HiPath 4000 V5	CMI	Phantom	Function (ISDN)	Attend Consc	General Attend	RCG	Hunt Group	Trunk	same device types valid in case supported (marked with x)
Set Forwarding	Set the forwarding status of a specified device.	X	X	x <sup>18</sup>		X	X							X
Set Routing Mode	Set the routing mode of a specified device.										X			X
I/O Registration Services														
I/O Register	Registers the computing function as an I/O server for a specified device or for the entire switching function.	X	X			X								X
I/O Register Cancel	Unregisters the computing function as an I/O server.	X	X			X								X
Input / Output Services														
Data Path Resumed (S->C)	The Data Path Resumed service provides information that a previously suspended data path has been resumed.	X	X			X								X
Data Path Suspended (S->C)	The Data Path Suspended service provides information that a data path has been suspended.	X	X			X								X
Fast Data (C->S)	The Fast Data service starts a data path for only the duration of sending one data message.		X											X
Fast Data (S->C)		X	X			X								X
Send Data (C->S)	The Send Data service writes data to a specified data path.		X			X								X
Send Data (S->C)		X	X			X								X
Start Data Path (C->S)	The Start Data Path service starts a data path on the specified object.	X	X			X								X
Stop Data Path (C->S)	The Stop Data Path service terminates an existing data path.	X	X			X								X
Stop Data Path (S->C)		X	X			X								X
Vendor Specific Extensions Services														
Escape Services (Get Lower Class Of Service)	Provides a mechanism to send a non-standardized feature.	X	X	X		X	X							-
Escape Services (Set Lower Class Of Service)	Provides a mechanism to send a non-standardized feature.	X	X	X		X	X							-
Escape Services (Connect Timeslot)	Provides a mechanism to send a non-standardized feature.	X	X	X		X	X	x <sup>19</sup>	X				X	X
Escape Services (Reconnect Timeslot)	Provides a mechanism to send a non-standardized feature.	X	X	X		X	X	x <sup>20</sup>	X				X	X

<sup>18</sup> The requests get/set system features, not SIP client features

<sup>19</sup> As destination only

<sup>20</sup> As destination only

## OpenScape 4000 CSTA Services

### Capability Exchange Services

CSTA ASN.1														CSTA XML
Services	Description	Station Analog	Station Digital	SIP only	SIP- Classi from OpenS 4000 V7R2 V5	CMI	Phanti	Functi (ISDN	Attend Conso	Gener Attend	RCG	Hunt Group	Trunk	same device types valid in case supported (marked with x)
Escape Services (Get Reroute Prevention)	Provides a mechanism to send a non-standardized feature.	X	X	X		X	X						X	-
Escape Services (Set Reroute Prevention)	Provides a mechanism to send a non-standardized feature.	X	X	X		X	X						X	-
Private Data Version Selection	Provides the switching function with the selected version for private data.													-

## 3.4 Capability Exchange Services

### Capability Exchange Services Summary

**Table 3: Support of Capability Exchange Services**

Capability Exchange Services	Service Description Section
Get Logical Device Information	<a href="#">Section 3.4.1</a>
Get Physical Device Information	<a href="#">Section 3.4.2</a>
Get Switching Function Capabilities	<a href="#">Section 3.4.3</a>
Get Switching Function Devices	<a href="#">Section 3.4.4</a>
Switching Function Devices	<a href="#">Section 3.4.5</a>

### Capability Exchange Services Descriptions

The entire ECMA CSTA III standard covering Capability Exchange Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.4.1 Get Logical Device Information

The Get Logical Device Information service is used to obtain the current set of characteristics/capabilities associated with the logical element of a given device.

## Request Message

Table 4: Get Logical Device Information Service Request Parameters

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device being queried

## Positive Response

Table 5: Get Logical Device Information Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
deviceCategory	Enumerated	M	<p>Specifies the device category (station, ACD device, etc.) of the device in the service request. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• ACD</li> <li>• Group</li> <li>• Network Interface (e.g., trunk, CO line)</li> <li>• <del>Park</del></li> <li>• <del>Routing Device</del></li> <li>• Station (default)</li> <li>• <del>Voice Unit</del></li> <li>• Other</li> </ul>
groupDeviceAttributes	Bitmap	C	<p>Specifies the group device attributes of the device being queried. If a bit is TRUE then the specified attribute is present. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>• ACD</li> <li>• Hunt</li> <li>• <del>Pick</del></li> <li>• Other</li> </ul> <p>This parameter shall be provided if the deviceCategory is Group, otherwise it shall not be provided.</p>

Parameter Name	Content	M/C/O	Comments
namedDeviceTypes	Enumerated	O	<p>If assigned by the switching function, this parameter indicates the named device type associated with the device in the service request. The complete set of possible values are:</p> <ul style="list-style-type: none"> <li>• ACD</li> <li>• ACD-Group</li> <li>• Button</li> <li>• Button-Group</li> <li>• Conference-Bridge</li> <li>• Line</li> <li>• Line-Group</li> <li>• Operator</li> <li>• Operator Group</li> <li>• Parking-Device</li> <li>• Station</li> <li>• Station-Group</li> <li>• Trunk</li> <li>• Trunk-Group</li> <li>• Other</li> <li>• Other Group</li> </ul>
hasPhysicalElement	Boolean	M	<p>Specifies if the device has a physical element associated with this device identifier. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• FALSE - The device does not have a physical element.</li> <li>• TRUE - The device does have a physical element.</li> </ul> <p>The device identifier in the service request should be used with the Get Physical Device Information service to obtain the physical element's characteristics for this device.</p>
shortFormDeviceID	DeviceID	Ø	<p><del>Specifies an alternative (a shorter length, for example) device identifier that the switching function may use to reference the device in the service request.</del></p>
acdModels	Bitmap	M	<p>Specifies the type of ACD Model(s) that are present at this device. If a bit is TRUE, then the specified model is supported. The following is the list of bits (multiple bits may be set):</p> <p>Visible ACD-related Devices</p> <p>Non-Visible ACD-related Devices</p> <p>Note that these bits are valid when the device is an ACD device.</p>

Parameter Name	Content	M/C/O	Comments
agentLogOnModels	Bitmap	C	<p>Specifies the types of agent log on models that are supported by the device. If a bit is TRUE, then the specified agent log on model is supported. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>• <del>Log On to an ACD device</del></li> <li>• Log On to an ACD Group (explicit/one step)</li> <li>• <del>Log On to an ACD Group (explicit/two steps)</del></li> <li>• Log On to an ACD Group (implicit/one step)</li> </ul> <p>Note that Log On to an ACD Group (implicit/one step) model cannot be simultaneously supported with the Log On to an ACD device model.</p> <p>The switching function shall provide this parameter if the agent log on model is configured by the switching function at the logical device element level (agent, ACD device, or ACD group), otherwise the parameter may or may not be provided.</p>
appearanceAddressable	Boolean	M	<p>Specifies whether the appearances of the logical element are addressable (via the Call Appearance "CA" string or the physical element extension "EXT" string in the Switching Function Representation Device Identifier format). The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• FALSE - The appearances are not addressable.</li> <li>• <del>TRUE - The appearances are addressable</del></li> </ul>
appearanceType	Enumerated	M	<p>Specifies the type of appearances associated with the logical element. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• <del>Selected-Standard</del></li> <li>• <del>Basic-Standard</del></li> <li>• <del>Basic-Bridged</del></li> <li>• <del>Exclusive-Bridged</del></li> <li>• <del>Independent-Shared-Bridged</del></li> <li>• <del>Interdependent-Shared-Bridged</del></li> </ul>
appearanceList	List of Characters	C	<p><del>Specifies the list of device identifier suffices for each of the appearances that are available at the logical element. This parameter is mandatory if the appearances are addressable and if</del></p> <p><del>it is a Selected-Standard or a Basic-Standard type. This list will</del></p> <p><del>only contain appearance suffices that can be observed and/or</del></p> <p><del>controlled within the switching sub-domain (via the Call Appearance string in the Switching Function Representation Device Identifier format):</del></p>

Parameter Name	Content	M/C/O	Comments
otherPhysicalDeviceList	List of DeviceIDs	G	<p>Specifies the list of device identifiers for other devices with physical elements that are associated with the logical element</p> <p>appearance:</p> <p>This parameter is mandatory if the appearances are addressable</p> <p>and any type of bridged appearance. The Get Physical Device</p> <p>Information service should be used to obtain the physical element characteristics associated with these other devices. This</p> <p>list will only contain devices that can be either observed and/or</p> <p>controlled within the switching sub-domain.</p> <p>Note that, for a Hybrid configuration, the order of device identifiers in this list is the same as the order of devices in the</p> <p>appearanceList parameter. See Functional Requirement #3 in</p> <p>10.1.2 for additional information.</p>

Parameter Name	Content	M/C/O	Comments
miscMonitorCaps	Bitmap	O	<p>Specifies the special types of monitoring considerations for this device. If a bit is TRUE then the monitoring consideration is associated with the device. The following is the list of bits (Multiple bits may be set):</p> <ul style="list-style-type: none"> <li>Group Inclusive Model – the scope of the monitor on the group device includes the distribution mechanism and all member devices. This bit is only valid for group devices that include a distribution mechanism (e.g. Hunt and ACD groups). This bit shall not be set if the Group Exclusive Model bit is set.</li> <li>Group Exclusive Model – the scope of the monitor on the group device includes only the distribution mechanism. This bit is only valid for group devices that include a distribution mechanism (e.g. Hunt and ACD groups). This bit shall not be set if the Group Inclusive Model bit is set</li> <li>Monitor the physical element to report call control events for all appearances associated with a device. (Only a valid bit if the appearanceType is any form of bridge appearance.) (i.e. use the device identifiers from the otherPhysicalDeviceList).</li> <li>ACD Device Inclusive – the scope of the monitor on an ACD device includes both the ACD device and the distributed to devices (including ACD groups). (This capability is valid only for ACD devices). This bit shall not be set if the ACD Device Exclusive bit is set</li> <li>ACD Device Exclusive - the scope of the monitor on an ACD device only the ACD device. (This capability is valid only for ACD devices). This bit shall not be set if the ACD Device Inclusive bit is set</li> </ul> <p>Note that if this parameter is not present, then the monitoring considerations are not known.</p>
associatedGroupList	List of DeviceIDs	C	<p>Specifies the list of device identifiers for all the other devices which are members of this group device. Use the appropriate capabilities exchange services to obtain the characteristics of these devices. This list shall only contain devices that can be either observed and/or controlled within the switching sub-domain.</p> <p>This parameter shall be provided when the device is a Group device. It may or may not be provided if the device is a member of a group and shall not be provided if the device is neither a Group device nor is a member of a group.</p>
maxCallbacks	Value	Θ	<p>Specifies the maximum number of concurrent call back requests</p> <p>that can be outstanding for this device. If this parameter is not</p> <p>present, then the maximum number of concurrent callback requests is not known for the device.</p>

## OpenScape 4000 CSTA Services

Parameter Name	Content	M/C/O	Comments
maxAutoAnswerRings	Value	Θ	Specifies the maximum number of rings before a call is autoanswered at this device. If this parameter is not present, then the maximum number of Auto Answer rings is not known for the device.
maxActiveCalls	Value	O	Specifies the maximum number of concurrent calls that can be active at any one time for this device. If this parameter is not present, then the maximum number of active calls is not known for the device.
maxHeldCalls	Value	O	Specifies the maximum number of concurrent calls that can be held at any one time for this device. If this parameter is not present, then the maximum number of held calls is not known for the device.
maxFwdSettings	Value	O	Specifies the maximum number of user-specified settings (forwarding-type/forward-destination combinations) that can be activated at any one time for this device. If this parameter is not present, then the maximum number of activated user-specified settings is not known for the device.
maxDevicesInConf	Value	O	Specifies the maximum number of devices both within and outside the switching function that this device can conference into a call. If this parameter is not present, then the maximum number of devices in a conference is not known for the device. The minimum value that can be supplied for this value is 3.
transAndConfSetup	Bitmap	O	Specifies the different ways that this device can set up for a conference and/or transfer. (Note that if this parameter is not present, then the device can only set up transfers and conferences through the Consultation Call service.) If the bit is TRUE, then the specified way to setup a conference or transfer is supported by the switching function. The following is the list of bits (multiple bits may be set): <ul style="list-style-type: none"> <li>• Consultation Call</li> <li>• Hold Call – Make Call</li> <li>• Alternate Call</li> <li>• two calls in the initial state of Hold</li> <li>• two calls in the initial state of Connected</li> </ul>
deviceOnDeviceMonitorFilter	MonitorFilter	C	Specifies the complete monitorFilter parameter that this device supports with respect to device-type monitoring. This parameter shall be provided if this form of device-type monitoring is supported, otherwise the parameter shall not be provided. <p>The information in the monitor filter parameters used in the Get Logical Device Information and the Get Physical Device Information services should be the same when the same device identifier is used (assuming that the device identifier has a logical and a physical element).</p>

Parameter Name	Content	M/C/O	Comments
deviceOnConnectionMonitorFilter	MonitorFilter	⊖	Specifies the complete monitorFilter parameter that connections at this device supports with respect to device-type monitoring.  This parameter shall be provided if this form of device-type monitoring is supported, otherwise the parameter shall not be provided.
callOnDeviceMonitorFilter	MonitorFilter	C	Specifies the complete monitorFilter parameter that this device supports with respect to call-type monitoring on a device. This parameter shall be provided if this form of call-type monitoring is supported, otherwise the parameter shall not be provided.
callOnConnectionMonitorFilter	MonitorFilter	⊖	Specifies the complete monitorFilter parameter that this device supports with respect to call-type monitoring for a connection at this device. This parameter shall be provided if this form of calltype monitoring is supported, otherwise the parameter shall not be provided.
mediaClassSupport	Bitmap	⊖	Specifies the media class of calls that the device can support.
mediaServiceCapsList	List of Structures	⊖	Specifies a list of structures of the media service types; version; media service instances; connection modes supported.
connectionRateList	List of Values	⊖	Specifies the list of connection rates that are supported for this device.
delayToleranceList	List of Values	⊖	Specifies the list of delay Tolerances that are supported for this device.
numberOfChannels	Value	⊖	Specifies the number of available channels at this device. If the parameter is not present, the number of channels at the device is not known but it is one or greater.
maxChannelBind	Value	⊖	Specifies the maximum number of channels that can be associated with a given connection at a device. If the parameter is not present, the maximum number of channels per connection is one.
routeingServList	RouteingServList	C	Specifies a list of bitmaps. Each bitmap entry represents a Routeing service that is supported by the device (both service requests to and from the switching function, when the service is bi-directional). This includes the following categories of services:  <ul style="list-style-type: none"> <li>• Routing Services</li> </ul> <p>This parameter shall be provided if the switching function supports any of these categories of services for this device. If a Routeing service's bitmap entry is not included in the list, then the service is not supported by the switching function. Note that the Routeing Mode feature is grouped with Logical Device features.</p>

Parameter Name	Content	M/C/O	Comments
logDevServList	Structure	C	<p>Specifies a list of capability bitmap parameter types corresponding to categories of services. Each bitmap entry in the lists represents a service that applies to a logical device that is supported by the device. This includes the following categories of services:</p> <ul style="list-style-type: none"> <li>• <code>callControlServList</code> (O) <code>CallControlServList</code> - specifies the list of call control services supported.</li> <li>• <code>callAssociatedServList</code> (O) <code>CallAssociatedServList</code> - specifies the list of call associated services supported.</li> <li>• <code>logicalServList</code> (O) <code>LogicalServList</code> - specifies the list of logical device feature services supported.</li> <li>• <del><code>mediaServList</code> (O) <code>MediaServList</code> - specifies the list of media services supported.</del></li> <li>• <code>ioServicesServList</code> (O) <code>IOServicesServList</code> - specifies the list of I/O services supported.</li> <li>• <del><code>dataCollectionServList</code> (O) <code>DataCollectionServList</code> - specifies the list of data collection services supported.</del></li> <li>• <del><code>voiceUnitServList</code> (O) <code>VoiceUnitServList</code> - specifies the list of voice unit services supported.</del></li> </ul> <p>This parameter shall be provided if the switching function supports at least one of these categories of services for this device.</p> <p>If a logical device service's bitmap entry is not included in the list, then the service is not supported by the device.</p>
logDevEvtsList	Structure	C	<p>Specifies a list of capability bitmap parameter types corresponding to categories of events. Each bitmap entry in the lists represents an event that applies to a logical device that is supported by the device. This includes the following categories of events:</p> <ul style="list-style-type: none"> <li>• <code>callControlEvtsList</code> (O) <code>CallControlEvtsList</code> - specifies the list of call control events supported.</li> <li>• <code>callAssociatedEvtsList</code> (O) <code>CallAssociatedEvtsList</code> - specifies the list of call associated events supported.</li> <li>• <code>logicalEvtsList</code> (O) <code>LogicalEvtsList</code> - specifies the list of logical device feature events supported.</li> <li>• <del><code>mediaEvtsList</code> (O) <code>MediaEvtsList</code> - specifies the list of media events supported.</del></li> <li>• <del><code>voiceUnitEvtsList</code> (O) <code>VoiceUnitEvtsList</code> - specifies the list of voice unit events supported.</del></li> </ul> <p>This parameter shall be provided if the switching function supports any of these categories of events for this device.</p> <p>If a logical device event's bitmap entry is not included in the list, then the event is not supported by the device.</p>

Parameter Name	Content	M/C/O	Comments
deviceMaintEvtsList	DeviceMaintEvtsList	C	<p>Specifies a list of bitmaps. Each bitmap entry represents a device maintenance event that is supported by the device. This includes the following categories of services:</p> <ul style="list-style-type: none"> <li>Device Maintenance events</li> </ul> <p>This parameter shall be provided if the switching function supports any of these categories of events for this device.</p> <p>If a device maintenance's bitmap entry is not included in the list, then the event is not supported by the switching function.</p>
security	CSSTASecurityData	Θ	Specifies timestamp information, message sequence number, and security information.

### Negative Response

**Table 6: Get Logical Device Information Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <p>Device ID contains an unsupported device ID type,</p> <p>Device does not contain a logical element.</p>
SystemResourceAvailability	generic	internal resource error,.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000.</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Dialing number "\*"888" as requested device delivers all configured RCG's within the AssociatedGroupList.

## 3.4.2 Get Physical Device Information

The Get Physical Device Information service is used to obtain the current set of characteristics/capabilities associated with the physical element of a given device.

**Request Message****Table 7: Get Physical Device Information Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device being queried

**Positive Response****Table 8: Get Physical Device Information Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
deviceCategory	Enumerated	M	Specifies the device category (station, ACD device, etc.) of the device being queried. The complete set of possible values is: <ul style="list-style-type: none"> <li>• ACD</li> <li>• Group</li> <li>• Network Interface (i.e., trunk)</li> <li>• Park</li> <li>• Routing</li> <li>• Station (default)</li> <li>• Voice Unit</li> <li>• Other</li> </ul>
namedDeviceTypes	Enumerated	O	If assigned by the switching function, this parameter indicates the named device type associated with the device being queried. The complete set of possible values are: <ul style="list-style-type: none"> <li>• ACD</li> <li>• ACD-Group</li> <li>• Button</li> <li>• Button-Group</li> <li>• Conference Bridge</li> <li>• Line</li> <li>• Line-Group</li> <li>• Operator</li> <li>• Operator Group</li> <li>• Parking Device</li> <li>• Station</li> <li>• Station-Group</li> <li>• Trunk</li> <li>• Trunk-Group</li> <li>• Other</li> <li>• Other-Group</li> </ul>

Parameter Name	Content	M/C/O	Comments
hasLogicalElement	Boolean	M	<p>Specifies if the device has a logical element associated with this device identifier. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• <del>FALSE</del>—The device does not have a logical element.</li> <li>• TRUE - The device does have a logical element.</li> </ul> <p>The device identifier in the service request should be used with the Get Logical Device Information service to obtain the logical element's characteristics for this device.</p>
otherLogicalDeviceList	List of Device IDs	O	<p>Specifies the list of device identifiers for other devices with logical elements that are associated with this device.</p> <p>The Get Logical Device Information service should be used to obtain the logical element characteristics associated with these other devices. This list will only contain devices that can be either observed and/or controlled within the switching function.</p>
deviceModelName	Characters (64)	O	<p>Specifies the switching function specific model name of the device. If this parameter is not present, then the model name is not known.</p>
deviceOnDeviceMonitorFilter	MonitorFilter	C	<p>Specifies the complete monitorFilter parameter that the device supports with respect to device-type monitoring. This parameter shall be provided if this form of device-type monitoring is supported, otherwise the parameter shall not be provided.</p> <p>The information in the monitor filter parameters used in the Get Logical Device Information and the Get Physical Device Information services should be the same when the same device identifier is used (assuming that the device identifier has a logical and a physical element).</p>
maxDisplays	Value	O	<p>Specifies the maximum number of displays associated with the device being queried. If this parameter is not present, then the device either does not have any displays or the maximum number of displays at this device is not known.</p>
physDevServList	PhysDevServList	C	<p>Specifies a list of bitmaps. Each bitmap entry represents a Physical Device service that is supported by the specified device. This includes the following categories of services:</p> <ul style="list-style-type: none"> <li>• Physical Device Feature services</li> </ul> <p>This parameter shall be provided if the switching function supports any of these categories of services for this device.</p> <p>If a physical device service's bitmap entry is not included in the list, then the service is not supported by the specified device.</p>

Parameter Name	Content	M/C/O	Comments
physDevEvtsList	PhysDevEvtsList	C	<p>Specifies a list of bitmaps. Each bitmap entry represents a Physical Device Event that is supported by the specified device. This includes the following categories of events:</p> <ul style="list-style-type: none"> <li>Physical Device Feature events</li> </ul> <p>This parameter shall be provided if the switching function supports any of these categories of events for this device.</p> <p>If a physical device event's bitmap entry is not included in the list, then the event is not supported by the specified device.</p>

### Negative Response

**Table 9: Get Physical Device Information Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <ul style="list-style-type: none"> <li>Device ID contains an unsupported device ID type,</li> <li>Device does not contain a physical element</li> </ul>
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).

## 3.4.3 Get Switching Function Capabilities

The Get Switching Function Capabilities service is used by the computing function to obtain the current set of capabilities for the entire switching function.

**Request Message**

Since the Get Switching Function Capabilities service request does not contain any service specific parameters, no table is provided.

**Positive Response****Table 10: Get Switching Function Capabilities Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
Parameter Name	Content	M/C/O	Comments
switchingSubDomainName	Characters (64)	M	Specifies the name of switching sub-domain which distinguishes it from other switching sub-domains.
manufacturerName	Characters (64)	M	Specifies the name of the manufacturer of the switching sub-domain.
profiles	Bitmap	M	<p>Specifies the CSTA Profiles supported by the switching function. The following is the list of the possible profiles (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>• Basic Telephony Profile</li> <li>• Routing Profile</li> </ul> <p>Note that at least one profile shall be supported by the switching function.</p>

Parameter Name	Content	M/C/O	Comments
deviceIDFormat	Bitmap	M	<p>Specifies the types of device ID formats supported by the switching function in service requests. If a bit is TRUE, then the specified format is used by the switching function. The following is the list of the possible formats (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>• Diable Digits format - "*"</li> <li>• Diable Digits format - "#"</li> <li>• Diable Digits format - "A-D"</li> <li>• Diable Digits format - "I"</li> <li>• Diable Digits format - "P"</li> <li>• Diable Digits format - "T"</li> <li>• Diable Digits format - "."</li> <li>• Diable Digits format - "W"</li> <li>• Diable Digits format - "@"</li> <li>• Diable Digits format - "\$"</li> <li>• Diable Digits format - ":"</li> <li>• SF Representation format - "I"</li> <li>• SF Representation format - "&amp;"</li> <li>• SF Representation format - "/"</li> <li>• SF Representation format - "%"</li> <li>• SF Representation format - "NM"</li> <li>• SF Representation format - Generic</li> <li>• SF Representation format - ImplicitTON</li> <li>• SF Representation format - PublicTON - unknown</li> <li>• SF Representation format - PublicTON - international number</li> <li>• SF Representation format - PublicTON - national</li> <li>• SF Representation format - PublicTON - subscriber</li> <li>• SF Representation format - PublicTON - abbreviated</li> <li>• SF Representation format - PrivateTON - unknown</li> <li>• SF Representation format - PrivateTON - level-3 regional</li> <li>• SF Representation format - PrivateTON - level-2 regional</li> <li>• SF Representation format - PrivateTON - level-1 regional</li> <li>• SF Representation format - PrivateTON - local</li> <li>• SF Representation format - PrivateTON - abbreviated</li> <li>• SF Representation format - Other</li> <li>• Device Number format</li> </ul> <p>Note that the Diable Digits format with the 0-9 characters shall be supported by the switching function.</p>

Parameter Name	Content	M/C/O	Comments
swdomainFeature	Bitmap	M	<p>Specifies which features are supported by the switching function. If a bit is TRUE, then the specified feature is supported. The following is the list of possible features (multiple bits can be set):</p> <p>Forwarding Call Associated models:</p> <ul style="list-style-type: none"> <li>Is (Immediate) Forwarding triggered before the call is logically delivered to the device?</li> <li><del>Is (Immediate) Forwarding triggered after the call is logically delivered to the device?</del></li> </ul> <p>Level of Forwarding Default Settings:</p> <ul style="list-style-type: none"> <li>Switching function default setting - allows activation/deactivation of a single switching function forwarding type/forwarding destination combination.</li> <li>User specified settings - allows the setting of individual forwarding types and forwarding destinations.</li> <li>User specified setting (default forwarding type) - If this is TRUE, when the forwarding type is omitted in the Set Forward service, the switching function applies a default value, otherwise there is no default value applied.</li> <li>User specified setting (default forward destination) - If this is TRUE, when the forward destination is omitted in the Set Forward service, the switching function applies a default value, otherwise there is no default value applied.</li> </ul> <p>Connection Failure:</p> <ul style="list-style-type: none"> <li>Negative Acknowledgement</li> <li>Support of Failed event with an associated failed connection</li> <li><del>Support of Failed event without an associated failed connection</del></li> <li><del>Support of Failed event with an associated failed connection, not reported via monitors on the failing device</del></li> </ul> <p>Other:</p> <ul style="list-style-type: none"> <li>Recall</li> <li>Call Back</li> <li>External Callsâ€œIncoming Calls</li> <li>External Callsâ€œOutgoing Calls</li> <li>Prompting</li> </ul>
swAppearanceAddressability	Bitmap	M	<p>Specifies what types of appearance addressability is available within the switching sub-domain. The following is the list of bits (multiple bits can be set):</p> <ul style="list-style-type: none"> <li>addressable</li> <li>non-addressable</li> </ul>

Parameter Name	Content	M/C/O	Comments
swAppearanceType	Bitmap	M	<p>Specifies what types of appearances are available within the switching sub-domain. The following is the list of bits (multiple bits can be set):</p> <ul style="list-style-type: none"> <li>Selected-Standard</li> <li><del>Basic-Standard</del></li> <li><del>Basic-Bridged</del></li> <li><del>Exclusive-Bridged</del></li> <li><del>Independent-Shared-Bridged</del></li> <li><del>Interdependent-Shared-Bridged</del></li> </ul>
ignoreUnsupportedParameters	Bitmap	M	<p>Specifies how the switching function handles unsupported optional parameters in service requests. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>Ignore parameters - This indicates that the switching function treats unsupported optional parameters as if they were not present.</li> <li><del>Reject Request - This indicates that the switching function returns a negative acknowledgement in response to any requests that contain unsupported optional parameters.</del></li> </ul>
privateDataFormat	Bitmap	C	<p>Specifies the format(s) of the privateData information supported by the switching function. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li><del>octetStringFromSF - the switching function provides privateData in the octetString format</del></li> <li>otherTypeFromSF - the switching function provides privateData in another format.</li> <li><del>octetStringToSF - the switching function supports receiving privateData in the octetString format</del></li> <li>otherTypeToSF - the switching function supports receiving privateData in another format.</li> </ul> <p>This parameter shall be provided if the switching function supports sending or receiving privateData.</p>
transAndConfSetup	Bitmap	O	<p>Specifies the different ways that the switching function can set up for a conference and/or transfer. (Note that if this parameter is not present, then the switching function can only set up transfers and conferences through the Consultation Call service.) If the bit is TRUE, then the specified way to setup a conference or transfer is supported by at least one device in the switching sub-domain. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>Consultation Call</li> <li><del>Hold Call - Make Call</del></li> <li><del>Alternate Call</del></li> <li><del>two calls in the initial state of Hold</del></li> <li><del>two calls in the initial state of Connected</del></li> </ul>

Parameter Name	Content	M/C/O	Comments
deviceOnDeviceMonitorFilter	Bitmap	C	<p>Specifies the complete monitorFilter parameter that is supported by the switching function when the monitorObject is a device and the monitorType is a device. Each bitmap entry represents an event that is supported by at least one of the devices in the switching function.</p> <p>This parameter shall be provided if this form of device-type monitoring is supported by at least one of the devices in the switching function, otherwise it shall not be provided.</p>
miscMonitorCaps	Bitmap	O	<p>Specifies the special types of monitoring capabilities that are present within the switching sub-domain. If a bit is TRUE then the monitoring capability is present within the switching sub-domain. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>• <del>Group Inclusive Model - the scope of the monitor on a group device includes the distribution mechanism and all member devices. This bit applies to group devices that include a distribution mechanism (e.g. Hunt and AGD groups):</del></li> <li>• <del>Group Exclusive Model - the scope of the monitor on the group device includes only the distribution mechanism. This bit applies to group devices that include a distribution mechanism (e.g. Hunt and AGD groups):</del></li> <li>• <del>Monitor the physical element to report call control events for all appearances associated with a device. (This capability is valid only if an appearanceType of any form of a bridge appearance is supported.)</del></li> <li>• <del>ACD Device Inclusive - the scope of the monitor on an ACD device includes both the ACD device and the distributed-to devices (including AGD groups). (This capability is valid only for ACD devices):</del></li> <li>• <del>ACD Device Exclusive - the scope of the monitor on an ACD device only the ACD device. (This capability is valid only for ACD devices).</del></li> </ul> <p>If this parameter is not present, then the monitoring considerations are not known.</p>
correlatorDataSupported	Boolean	O	<p>Specifies if the switching function supports the correlatorData parameter on service requests and events. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• <del>TRUE - Option supported.</del></li> <li>• FALSE - Option is not supported.</li> </ul> <p>Refer to "Correlator Data" on page 28 for the required events that shall support correlator data if this option is supported.</p>

Parameter Name	Content	M/C/O	Comments
dynamicFeatureSupported	Enumerated	O	<p>Specifies how the switching function provides the servicesPermitted parameter on events. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li><del>none – servicesPermitted not provided on any events</del></li> <li>all - servicesPermitted provided on all events where it is specified</li> <li><del>some – servicesPermitted provided on some events. Refer to the logDevEvtsList parameter for the events that support the parameter.</del></li> </ul>
callLinkageOptions	Bitmap	O	<p>Specifies if the switching function supports the call linkage and thread linkage features. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>callLinkageFeatureSupported - the switching function supports the call linkage feature. This feature shall be supported if the thread linkage feature is supported.</li> <li>threadLinkageFeatureSupported - the switching function supports the thread linkage feature.</li> </ul>
acdModels	Bitmap	O	<p>Specifies the types of ACD models that are supported by the switching function. If a bit is TRUE, then the specified ACD model is supported by the switching function. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>Visible ACD-Related Devices</li> <li>Non-Visible ACD-Related Devices</li> </ul> <p>Note that if more than one type of ACD model is present in the switching function, then the Get Logical Device Information service shall be used to determine the particular ACD models supported by for each ACD device or ACD group by the switching function.</p>
agentLogOnModels	Bitmap	O	<p>Specifies the types of agent log on models that are supported by the switching function. If a bit is TRUE, then the specified agent log on model is supported by the switching function. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li><del>Log On to an ACD device</del></li> <li>Log On to an ACD Group (explicit/one step)</li> <li><del>Log On to an ACD Group (explicit/two steps)</del></li> <li>Log On to an ACD Group (implicit/one step)</li> </ul> <p>Note that if more than one type of model is present in the switching function, then the Get Logical Device Information service shall be used to determine the particular Log On models supported for each device which is or can be associated with an agent.</p>

Parameter Name	Content	M/C/O	Comments
agentStateModels	Bitmap	O	<p>Specifies the types of agent models that are supported by the switching function. If a bit is TRUE, then the specified agent model is supported by the switching function. The following is the list of bits (multiple bits may be set):</p> <ul style="list-style-type: none"> <li>• <del>Agent Multi-State Model</del></li> <li>• <del>Agent Multi-State Model (Semi-Independent Linked)</del></li> <li>• Agent Oriented Model</li> </ul> <p>Note that if more than one type of model is present in the switching function, then the Get Logical Device Information service shall be used to determine the particular Agent models supported for each ACD device or ACD group by the switching function.</p>
connectionView	Enumerated	M	<p>Specifies the meaning of the primary and secondary old call parameters in the Conferenced and Transferred events. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• <del>fixed view - the contents of the primary and secondary old call parameters are independent of the monitoring type and the role of the device in the conference or transfer.</del></li> <li>• local view - the contents of the primary and secondary old call parameters are dependent upon which device is being monitored.</li> </ul> <p>Refer to the descriptions of the Conferenced and the Transferred events for more information.</p>

## OpenScape 4000 CSTA Services

Parameter Name	Content	M/C/O	Comments
maxLengthParameters	List of Values	M	<p>Each value is the switching function's maximum length (in octets/characters) for the corresponding parameters and parameter types. The computing function should not send larger data or the service request will be rejected. The following list provides the different parameters and parameter types for which a maximum value is provided. The number in parenthesis specifies the maximum possible length.</p> <ul style="list-style-type: none"> <li>AccountInfo parameter type: (32) - OpenScape 4000: (0)</li> <li>AuthCode parameter type: (32) - OpenScape 4000: (0)</li> <li>AgentID parameter type: (32) - OpenScape 4000: (6)</li> <li>AgentPassword parameter type: (32) - OpenScape 4000: (0)</li> <li>callID in the ConnectionID parameter type: (8) - OpenScape 4000: (4)</li> <li>CorrelatorData parameter type: (32) - OpenScape 4000: (0)</li> <li>CSTAPrivateData parameter type: (any value) - OpenScape 4000: (256)</li> <li>Device Identifiers parameter type: (128)</li> <li>UserData parameter type: (256) - OpenScape 4000: (32)</li> <li>buttonLabel parameters: (64) - OpenScape 4000: (0)</li> <li>lampLabel parameters: (64) - OpenScape 4000: (0)</li> <li>charactersToSend parameter: (64) - OpenScape 4000: (44)</li> </ul> <p>If any of the above values is zero, then the parameter or parameter type is not supported.</p>

Parameter Name	Content	M/C/O	Comments
servEvsList	List	C	<p>Specifies a list of capability bitmap parameter types corresponding to categories of services. Each bitmap entry in the lists represents a service or event that is supported by the switching function. This includes the following categories of services/events:</p> <ul style="list-style-type: none"> <li>capExchangeServList (O) CapExchangeServList</li> <li>systemStatusServList (O) SystemStatusServList</li> <li>monitoringServList (O) MonitoringServList</li> <li>snapshotServList (O) SnapshotServList</li> <li>callControlServList (O) CallControlServList</li> <li>callControlEvsList (O) CallControlEvsList</li> <li>callAssociatedServList (O) CallAssociatedServList</li> <li>callAssociatedEvsList (O) CallAssociatedEvsList</li> <li><del>mediaServList (O) MediaServList</del></li> <li><del>mediaEvsList (O) MediaEvsList</del></li> <li>routeingServList (O) RouteingServList</li> <li>physServList (O) PhysServList</li> <li>physEvsList (O) PhysEvsList</li> <li>logicalServList (O) LogicalServList</li> <li>logicalEvsList (O) LogicalEvsList</li> <li>deviceMaintEvsList (O) DeviceMaintEvsList</li> <li>ioServicesServList (O) IOServicesServList</li> <li><del>dataCollectionServList (O) DataCollectionServList</del></li> <li><del>voiceUnitServList (O) VoiceUnitServList</del></li> <li><del>voiceUnitEvsList (O) VoiceUnitServList</del></li> <li><del>cdrServList (O) CDRServList</del></li> <li>vendorSpecificServList (O) VendorSpecificServList</li> <li>vendorSpecificEvsList (O) VendorSpecificEvsList</li> </ul> <p>This parameter shall be provided if the switching function supports any of these categories of services/ events.</p> <p>If a list entry is not included in the list, then the corresponding category of services/events is not supported by the switching function.</p>
privateDataVersionList	List of Values	O	<p>If the switching function supports the private data mechanism, this parameter provides the list of supported private data versions associated with the switching function manufacturer</p> <p>- OpenScape 4000: 1</p>
systemStatusTimer	Value	C	<p>Specifies a timer value indicating how often the switching function sends periodic system status requests to the computing function (i.e. heartbeats). This parameter has a value between 0 and 180 seconds. 0 means that the switching function does not send periodic System Status service requests.</p> <p>This parameter shall be provided if the switching function supports the heartbeat timer via the System Status service.</p> <p>- OpenScape 4000: (30)</p>

Parameter Name	Content	M/C/O	Comments
simpleThreshold	Value	O	<p>Specifies the number of unacknowledged service requests that are allowed at any time for the switching function. 0 means there is no limit.</p> <p>If this parameter is not provided, the simpleThreshold is unknown. - OpenScape 4000: (100)</p>

### Negative Response

**Table 11: Get Switching Function Capabilities Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
SystemResourceAvailability	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

## 3.4.4 Get Switching Function Devices

The Get Switching Function Devices service is used by the computing function to obtain the current set of devices in the application working domain along with their associated device categories and associated device names.

### Request Message

**Table 12: Get Switching Function Devices Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
requestedDeviceID	DeviceID	O	<p>Specifies the device identifier of the device being queried. If this parameter is not present, the switching function returns a list of all devices (of the requestedDeviceCategory, if that parameter is provided) in the switching sub-domain.</p>

Parameter Name	Content	M/C/O	Comments
requestedDeviceCategory	Enumerated	O	<p>Specifies that only devices of the requested category be provided. If this parameter is not present, the switching function will return a list of all devices (or just the requested device) in the switching function. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• ACD</li> <li>• <del>Group (ACD)</del></li> <li>• Group (Hunt)</li> <li>• <del>Group (Pick)</del></li> <li>• Group (Other)</li> <li>• Network Interface</li> <li>• <del>Park</del></li> <li>• <del>Routeing Device</del></li> <li>• Station</li> <li>• <del>Voice Unit</del></li> <li>• Other</li> </ul>

### Positive Response

**Table 13: Get Switching Function Devices Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
serviceCrossRefID	ServiceCrossRefID	M	Specifies the correlator used to associate subsequent Switching Function Devices services to this service request.

### Negative Response

**Table 14: Get Switching Function Devices Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error.
	invalidFeature	<p>The service request specified a feature that is invalid. Often, this is because the switching or computing function does not support the requested feature.</p> <ul style="list-style-type: none"> <li>• The requested device category is not identified by the switching domain.</li> </ul>
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>• No link to OpenScape 4000</li> </ul>

**Usage Notes**

- 1) This service is supported for devices listed in [Table 2 "Services-supported-by OpenScape 4000 devices"](#).
- 2) The Switching Function Devices service returns all device identifiers that refer to all devices in the application working domain (i.e devices that can be controlled and/or observed).
- 3) Changes of devices in OpenScape 4000 only appear to the computing domain after the use of CBAdmin (select Application - Update Device List ...).
- 4) The CA4000 retrieves the information of the queried devices based on its internal device list from the OpenScape 4000, which is filled up default at 1:00 AM To activate the feature please set the GETSWFUNCDEV\_TIME option in the configuration file. The format is hhmm, e.g. to set the time to 9:30: GETSWFUNCDEV\_TIME=930 Possible values are from 1 to 2359, although it is not advised to use the time from 23:55 to 0:05.

**3.4.5 Switching Function Devices**

The Switching Function Devices service is used by the switching function to provide a list of devices in the application working domain. This service is generated as a result of the Get Switching Function Devices service.

The switching function may generate a sequence of Switching Function Devices services, individually referred to as segments, in response to a single Get Switching Function Devices service request.

**Request Message****Table 15: Switching Function Devices Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
serviceCrossRefID	ServiceCrossRefID	M	Specifies the cross reference used to associate the Switching Function Devices service request to the Get Switching Function Devices 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"> <li>• TRUE - Indicates that this is the last segment</li> <li>• FALSE - Indicates that this is not the last segment in the sequence.</li> </ul>

Parameter Name	Content	M/C/O	Comments
deviceList	List of structures	M	<p>Specifies the list of device Identifiers representing the devices that can be controlled and/or observed. It includes the following components for each device in the list:</p> <ul style="list-style-type: none"> <li>deviceID (M) DeviceID - Specifies a Device Identifier associated with the entry in the list.</li> <li>deviceCategory (O) Enumerated - Specifies the device category associated with the entry in the list.</li> </ul> <p>The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>ACD</li> <li>Group</li> <li>Network Interface (e.g. trunk, CO Line)</li> <li>Park</li> <li>Routing</li> <li>Station (default)</li> <li>Voice Unit</li> <li>Other</li> </ul>

Parameter Name	Content	M/C/O	Comments
(continued)	(continued)		<ul style="list-style-type: none"> <li>namedDeviceTypes (O) Enumerated - indicates the named device type associated with the device in the service request. The complete set of possible values is:   ACD  ACD-Group  <del>Button</del>  <del>Button-Group</del>  <del>Conference-Bridge</del>  <del>Line</del>  <del>Line-Group</del>  Operator  Operator Group  <del>Parking-Device</del>  Station  <del>Station-Group</del>  Trunk  <del>Trunk-Group</del>  Other  <del>Other-Group</del> </li> <li>deviceAttributes (O) Bitmap - Specifies additional attributes of the device associated with the entry in the list. This is a bit list of the following set of possible values (multiple bits may be set):   <del>mediaAccessDevice - indicates that the device is also a media access device</del>   routeingDevice - indicates that the device is also a routeing device   <del>Group (ACD) - indicates that the Group device has an ACD attribute (e.g. is an ACD-Group)</del>   Group (Hunt) - indicates that the Group device has a Hunt attribute   <del>Group (Pick) - indicates that the Group device has a Pick attribute</del> </li> <li>deviceModelName (O) Characters (64) - Specifies the switching function specific model name associated with the entry in the list.</li> </ul>

Parameter Name	Content	M/C/O	Comments
privateData	CSTAPrivateData	O	<p>Specifies non-standardized information.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>trunkGroup</li> </ul> <p>(see <a href="#">Chapter 9, "Appendix C—Private Data"</a>)</p>

#### Positive Response

There is no positive acknowledgment associated with this service request.

#### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific and will be ignored by OpenScape 4000. An entry in the error log will be generated.

#### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) PrivateData is only supported for the deviceCategory Network Interface.

## 3.5 System Services

### System Services Summary

**Table 16: Support of System Services**

System Services	Service Description Section
Change System Status Filter	<a href="#">Section 3.5.1</a>
System Register	<a href="#">Section 3.5.2</a>
System Register Abort	<a href="#">Section 3.5.3</a>
System Register Cancel	<a href="#">Section 3.5.4</a>
Request System Status	<a href="#">Section 3.5.5</a>
System Status	<a href="#">Section 3.4.3</a>
Switching Function Devices Changed	<a href="#">Section 3.5.7</a>

#### System Services Descriptions

The entire ECMA CSTA III standard covering system status Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

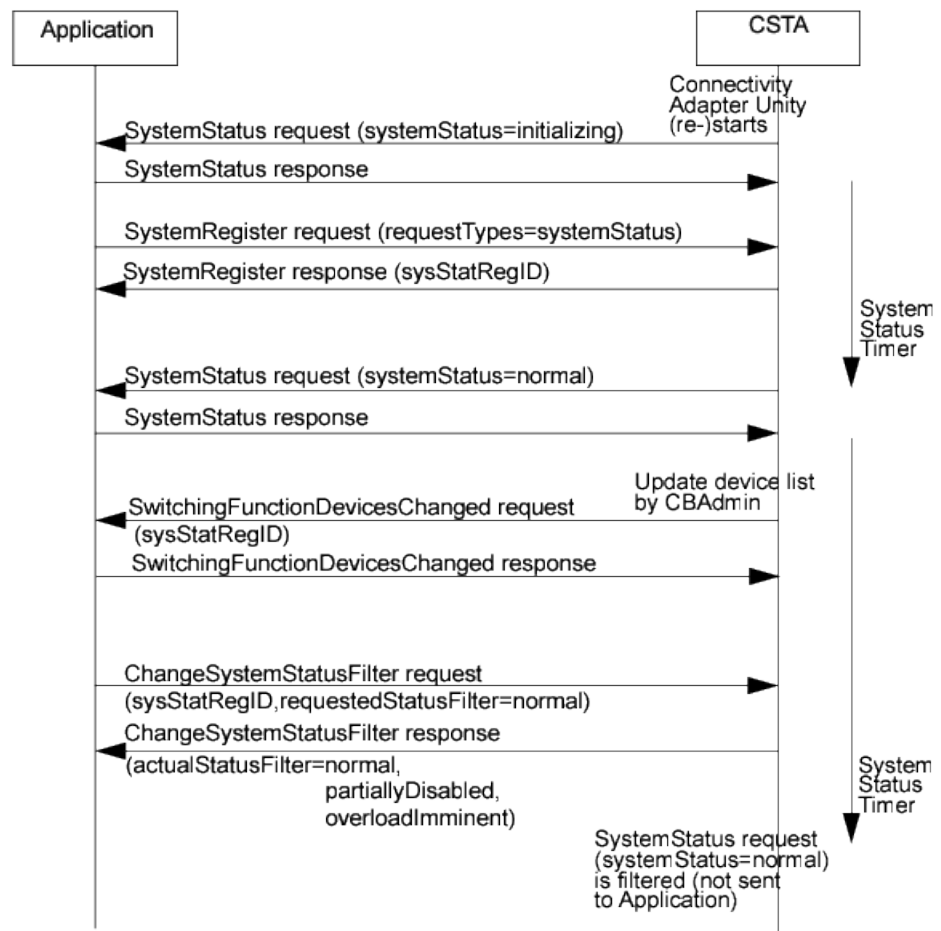
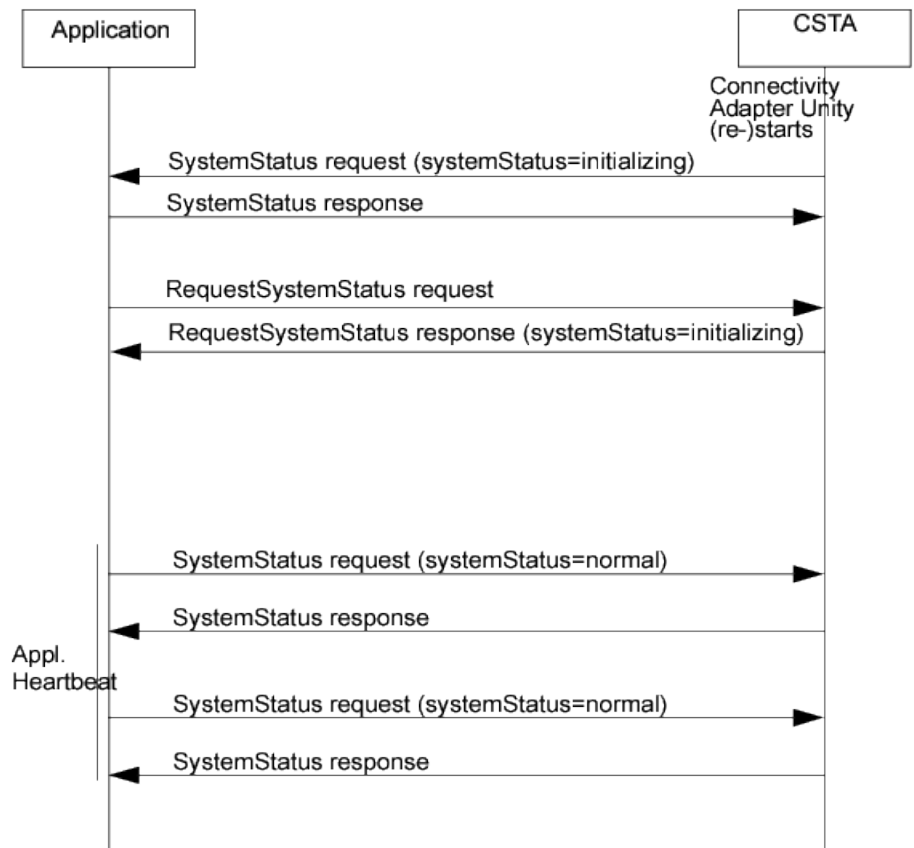
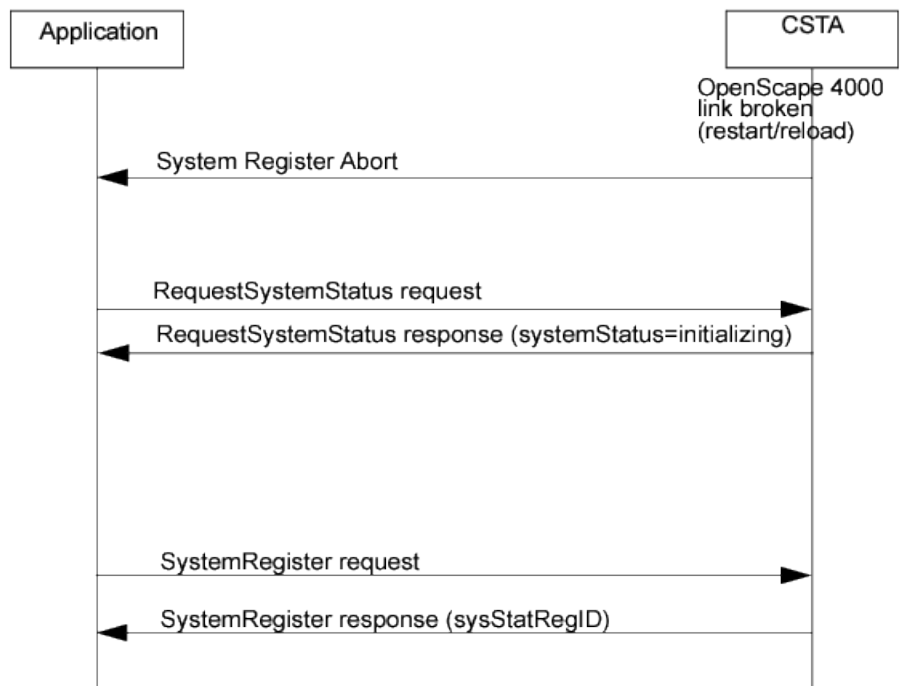


Figure 9: Sequence for System Services with registration



**Figure 10: Sequence for System Services without registration**



In case the connection between CA4000 and OpenScape4000 is broken, a System Register Abort service will be sent to inform the application that the

switch is not available. The System Status registration will be cleared as well. This means that the application has to start the initialization by requesting System Status until the switch becomes available. Afterwards the System Status Register service can be invoked to register System Status again.

When the switch becomes available again the SystemRegister Request will be successful and the application will get the restart type in the Response as private data.

### 3.5.1 Change System Status Filter

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

#### Request Message

**Table 17: Change System Status Filter Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
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"> <li>• Initializing</li> <li>• Enabled</li> <li>• Normal</li> <li>• Messages Lost</li> <li>• Disabled</li> <li>• Partially Disabled</li> <li>• Overload Imminent</li> <li>• Overload Reached</li> <li>• Overload Relieved</li> <li>• Multiple bits may be set.</li> </ul>

**Positive Response****Table 18: Change System Status Filter Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
actualStatusFilter	Bitmap	M	<p>Specifies the actual set of System Status Types that will be filtered (not sent) by the switching function.</p> <p>This parameter is a bitmap with the same set of possible values as in the service request.</p> <p>The actualStatusFilter may differ from the requestedStatusFilter parameter in the service request.</p>

**Negative Response****Table 19: Change System Status Filter Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidCrossRefID	<p>The service request specified a Cross Reference Identifier that is not in use.</p> <ul style="list-style-type: none"> <li>The computing function is not registered for system status</li> </ul>
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

**Usage Notes**

- 1) The service request will be answered with a positive response, when the computing function has registered to System Status service.
- 2) The actualStatusFilter in the response includes the received ones (included in requestedStatusFilter) and all of the following list:
  - partiallyDisabled
  - overloadImminent
- 3) When the computing function requests all possible status to be filtered, nothing will be reported, but the service request will be positively acknowledged.

## 3.5.2 System Register

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.

### Request Message

**Table 20: System Register Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
requestTypes	Bitmap	M	<p>Specifies the system services that are being registered. This parameter is a bitmap of the following values:</p> <ul style="list-style-type: none"> <li>System Status</li> <li><del>Request System Status</del></li> <li><del>Switching Function Capabilities Changed</del></li> <li>Switching Function Devices Changed</li> </ul> <p>Multiple bits may be set.</p>
requestedStatusFilter	Bitmap	C	<p>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:</p> <ul style="list-style-type: none"> <li>Initializing</li> <li>Enabled</li> <li>Normal</li> <li>Messages Lost</li> <li>Disabled</li> <li><del>Partially Disabled</del></li> <li><del>Overload Imminent</del></li> <li>Overload Reached</li> <li>Overload Relieved</li> </ul> <p>Multiple bits may be set.</p> <p>This parameter is mandatory if the requestTypes parameter includes System Status, otherwise it shall not be provided.</p>

### Positive Response

**Table 21: System Register Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
sysStatRegisterID	SysStatRegisterID	M	Specifies the system registration identifier for this registration.

Parameter Name	Content	M/C/O	Comments
actualStatusFilter	Bitmap	C	<p>Specifies the actual set of System Status Types that will be filtered (not sent) by the switching function.</p> <p>This parameter is a bitmap with the same set of possible values as in the service request.</p> <p>The actual types filtered may differ from what was requested in the service request.</p> <p>If the requestType parameter in the service request includes System Status, then this parameter is mandatory, otherwise it shall not be provided.</p>
switchRestartType (private)	Enumerated	O	<p>This parameter will be sent after a link brokenage or a switch restart.</p> <p>Possible values:</p> <p>switchRestartTypeSoftSymplex (0)</p> <p>switchRestartTypeSoftDuplex (1)</p> <p>switchRestartTypeHard (2)</p> <p>switchRestartTypeServerOnly (3)</p> <p>switchRestartTypeNone (4) (only link brokenage)</p> <p>switchRestartTypeReload (5)</p>

### Negative Response

**Table 22: System Register Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidParameterValue	A value for a parameter is invalid. A value is in the specified range but invalid in the circumstance where it is used.
SubscribedResourceAvailabilityErrors	objectRegistrationLimitExceeded	This service request would exceed the switching function's limit on the number of registrations.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) The registration is deleted, if the link to the computing function (host) fails or if OpenScape 4000 CSTA restarts. The registration is aborted if the link between OpenScape 4000 CSTA and OpenScape 4000 fails. This will cause the deletion of registration as well.

- 2) The number of simultaneous system registrations per link allowed is one. When this limit is reached, subsequent System Register service requests result in negative acknowledgements from OpenScape 4000 CSTA.
- 3) OpenScape 4000 CSTA does not support all System Status Types. It 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 OpenScape 4000 CSTA) than what was requested in the service request.

### 3.5.3 System Register Abort

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.

#### Request Message

**Table 23: System Register Abort Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
sysStatRegisterID	SysStatRegisterID	M	Specifies the system registration identifier for the system registration that was aborted.

#### Positive Response

There is no positive acknowledgement defined for this service.

#### Negative Response

Negative acknowledgement is not supported by Connectivity Adapter.

#### Usage Notes

- 1) The registration is aborted by the switching function if the link between OpenScape 4000 CSTA and OpenScape 4000 fails.
- 2) This service will delete the registration of System Status, so the application has to try to register with the System Status Register service again until the switch becomes available.

### 3.5.4 System Register Cancel

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.

**Request Message****Table 24: System Register Cancel Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
sysStatRegisterID	SysStatRegisterID	M	Specifies the system registration identifier for which the system registration is to be cancelled.

**Positive Response**

Since the SystemRegister Cancel service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 25: System Register Cancel Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidCrossRefID	The service request specified a Cross Reference Identifier that is not in use.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

**Usage Notes**

- 1) The registration is deleted, if the link to the computing function (host) falls out or if OpenScape 4000 CSTA restarts.

**3.5.5 Request System Status**

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.

**Request Message**

Since the Request System Status service request does not contain any service specific parameters, no table is provided.

**Positive Response****Table 26: Request System Status Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
systemStatus	Enumerated	M	<p>Specifies the status of the function issuing the service request. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• Initializing</li> <li>• Enabled</li> <li>• Normal</li> <li>• Messages Lost</li> <li>• Disabled</li> <li>• Partially Disabled</li> <li>• Overload Imminent</li> <li>• Overload Reached</li> <li>• Overload Relieved</li> </ul>

**Negative Response****Table 27: Request System Status Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>• No link to OpenScape 4000</li> </ul>

**Usage Notes****1) Miscellaneous Characteristics:**

- Switching Functions supports sending the service request
- Switching Functions supports receiving the service request

**3.5.6 System Status**

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.

**Request Message**

This is a bidirectional service.

**Table 28: System Status Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
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"> <li>• Initializing</li> <li>• Enabled</li> <li>• Normal</li> <li>• Messages Lost</li> <li>• Disabled</li> <li>• <del>Partially Disabled</del></li> <li>• <del>Overload Imminent</del></li> <li>• Overload Reached</li> <li>• Overload Relieved</li> </ul> Refer to <a href="#">Table 29 "System Status Causes" on page 99</a>
privateData	CSTAPrivateData	O	Specifies non-standardized information.  OpenScape 4000: <ul style="list-style-type: none"> <li>• applicationName (C-&gt; S)</li> <li>• routeDestinationWhenLossOfHeartbeat (C -&gt; S)</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

[Table 29 "System Status Causes" on page 99](#) describes the supported system status for both OpenScape 4000 and computer-originated service request.

**Table 29: System Status Causes**

Originator	Cause	Description and Expected Response from the Application
OpenScape 4000	Messages Lost	<p>CSTA events may have been lost.</p> <p>The computer-resident software should take whatever recovery actions are necessary to keep databases updated while facing the prospect that at least one CSTA event may have been lost.</p> <p>This message also may indicate that the OpenScape 4000 is restarting. In this case, service request messages are rejected until the OpenScape 4000 link is re-established. This process may take up to eight minutes.</p>

Originator	Cause	Description and Expected Response from the Application
	Disabled	<p>Possible meanings:</p> <ul style="list-style-type: none"> <li>• OpenScape 4000 CSTA is shutting down, restarting, or reloading. In this scenario, this message may not reach the computer-resident software.</li> <li>• The OpenScape 4000 CSTA event stream has been disabled through the OpenScape 4000 configuration software.</li> </ul> <p>In either case, the computer-resident software needs to treat OpenScape 4000 CSTA as if it were in the not ready state. In this state, it should not send any more services to OpenScape 4000 CSTA, until OpenScape 4000 CSTA has signalled that it has entered the ready state, by sending a System Status message with a value enabled.</p> <p>If the OpenScape 4000 CSTA is performing a shutdown, the enabled message is not sent until the server is manually restarted.</p> <p>If the OpenScape 4000 CSTA is performing a software restart, it could take a minute or so before the enabled message is sent.</p> <p>If the OpenScape 4000 CSTA is being reloaded, it may take 5 - 7 minutes before the enabled message is sent.</p> <p>In all these states, services sent to OpenScape 4000 CSTA from the computer-resident software are not acknowledged.</p> <p>All active monitors have been cleared.</p>
OpenScape 4000	Enabled	<p>Possible meanings:</p> <ul style="list-style-type: none"> <li>• OpenScape 4000 CSTA is ready. If no initializing message has been sent previously, all monitor points are still valid.</li> <li>• The OpenScape 4000 CSTA event stream has been re-enabled through the OpenScape 4000 configuration software.</li> </ul> <p>In either case, OpenScape 4000 CSTA is in the ready state. The computer-resident software can now start to send services to OpenScape 4000 CSTA.</p> <p>Monitors are still valid unless this message was preceded by an initializing message.</p>

Originator	Cause	Description and Expected Response from the Application
	Initializing	<p>OpenScape 4000 CSTA is initializing. All monitors have been lost.</p> <p>The computer-resident software needs to treat OpenScape 4000 CSTA as if it has entered the not ready state. In this state it should not send any more services to OpenScape 4000 CSTA until OpenScape 4000 CSTA has signaled that it has entered the ready state, by sending a System Status message with a value enabled.</p> <p>At this point there are no active monitors.</p>
	Overload reached	<p>An overload condition has been reached.</p> <p>If possible, the computer-resident software must not send further requests until it receives a System Status message with the value overload relieved.</p>
	Overload relieved	<p>An overload condition has been relieved.</p> <p>The computer-resident software resumes normal processing.</p>
	Normal	<p>Used as a loopback or heartbeat message.</p> <p>The computer-resident software positively acknowledges this service.</p>
Computing Function	Enabled	<p>The computer-resident software requests to clear all active monitors.</p> <p>OpenScape 4000 CSTA clears all active monitors.</p>
	Normal	<p>Used as a loopback or heartbeat message.</p> <p>OpenScape 4000 CSTA sends a positive response to acknowledge this service.</p>

#### Positive Response

Since the System Status service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific and will be ignored by OpenScape 4000. An entry in the error log will be generated.

If a System Status service is requested by the computing domain the following negative responses from OpenScape 4000 CSTA are possible:

**Table 30: System Status Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidParameterValue	<p>A value for a parameter is invalid. A value is in the specified range but invalid in the circumstance where it is used.</p> <ul style="list-style-type: none"> <li>The system status of the requesting application is not normal or enabled</li> </ul>
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <ul style="list-style-type: none"> <li>The type of the redirection party of the heartbeat mechanism is illegal</li> </ul>
SystemResourceAvailability	invalidDestination	<p>The service request contains a destination that is invalid.</p> <ul style="list-style-type: none"> <li>The redirection party of the heartbeat mechanism is invalid</li> </ul>
	generic	Internal resource error
	resourceLimitExceeded	The service is supported by the server, but is unavailable because it would exceed the internal usage limit of the resource.
PrivateDataInfoErrors	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	cSTAPrivateDataInfoError	<p>An error occurred in the privateData parameter. The reason for this error is implementation specific.</p> <ul style="list-style-type: none"> <li>System Status service request contains private data with manufacturer "Unify CAP" but with no application name.</li> </ul>

#### Usage Note

- 1) The System Status service request in direction switching function to computing function is send after registration of an application with the requestTypes System Status in a time periode of 30 sec (systemStatusTimer in Get Switching Function Capabilities positive acknowledgement).
- 2) The System Status service request in direction computing function to switching function is used to implement a heartbeat mechanism between a computing function and the OpenScape 4000. No registration is necessary.
- 3) If the received manufacturer doesn't match to the defined one, the optional privateData will be ignored.

- 4) The first (mandatory) System Status service request (initializing) in direction to the computing function is sent from the switching function during an initializing sequence before registration of the computing function.
- 5) Miscellaneous Characteristics:
  - Switching Functions supports sending the service request
  - Switching Functions supports receiving the service request

### 3.5.7 Switching Function Devices Changed

The Switching Function Devices Changed service is used to indicate that information associated with the current set of devices that can be controlled and observed in the switching sub-domain has changed.

The Get Switching Function Devices service may be used to obtain the revised information.

#### Request Message

**Table 31: Switching Function Devices Changed Parameters**

Parameter Name	Content	M/C/O	Comments
sysStatRegisterID	SysStatRegisterID	C	<p>Specifies the system registration identifier associated with the system registration for this request.</p> <p>This parameter is mandatory if the switching function supports system registration and shall not be provided otherwise.</p>

#### Positive Response

Since the Switching Function Devices Changed positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific and will be ignored by OpenScape 4000. An entry in the error log will be generated.

#### Usage Notes

- 1) The service request is activated via CBAdmin (select Application - Update Device List ...).
- 2) The Get Switching Function Device Service request can be used to get the current device list.

## 3.6 Monitoring Services

### Monitoring Services Summary

Table 32: Support of Monitoring Services

Monitoring Services	Service Description Section
Change Monitor Filter	<a href="#">Section 3.6.1</a>
Monitor Start	<a href="#">Section 3.6.2</a>
Monitor Stop	<a href="#">Section 3.6.3</a>

### Monitoring Services Descriptions

The entire ECMA CSTA III standard covering Monitoring Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.6.1 Change Monitor Filter

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.

### Request Message

Table 33: Change Monitor Filter Service Request Parameters

Parameter Name	Content	M/C/O	Comments
crossRefIdentifier	MonitorCrossRefID	M	This indicates the monitor for which to change the filter.
requestedFilterList	MonitorFilter	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.

### Positive Response

**Table 34: Change Monitor Filter Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
actualFilterList	MonitorFilter	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. This parameter is optional if the actualFilterList is the same as the requestedFilterList parameter on the service request, otherwise it is mandatory.

### Negative Response

**Table 35: Change Monitor Filter Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCrossReference Identifier	The service request specified a Cross Reference Identifier that is not in use.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  • No link to OpenScape 4000

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices" on page 45](#).

## 3.6.2 Monitor Start

The Monitor Start service initiates event reports (otherwise known as events) for a 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 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.

### Request Message

**Table 36: Monitor Start Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
monitorObject	DeviceID	M	Specifies the monitor object of a call or device to be monitored. This shall be one of the following choices: <ul style="list-style-type: none"> <li>• <del>call (ConnectionID)</del> - Specifies a <del>call (connection)</del> object.</li> <li>• device (DeviceID) - Specifies a device object.</li> </ul>
requestedMonitorFilter	MonitorFilter	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.  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).
monitorType	Enumerated	O	Specifies the type of monitor requested. The complete set of possible values is: <ul style="list-style-type: none"> <li>• call-type</li> <li>• device-type</li> </ul> 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).

## Positive Response

Table 37: Monitor Start Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
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	MonitorFilter	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.</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>

Parameter Name	Content	M/C/O	Comments
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"> <li>• TRUE - Indicates event reports will be provided for calls that are at the device at the time of the acknowledgement [Default].</li> <li>• FALSE - Indicates event reports will not be provided for calls that are at the device at the time of the acknowledgement.</li> </ul> <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>

### Negative Response

**Table 38: Monitor Start Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	<p>Device ID contains an unsupported device ID type or is not known to the switching function.</p> <p>The monitor party is not configured as a device or is not configured in the OpenScape 4000 dialing plan.</p>
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	invalidMonitorObject	The monitor object is invalid.
	invalidMonitorType	The monitor type is invalid.
SystemResourceAvailability	generic	Internal resource error.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  • No link to OpenScape 4000
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.
	overallMonitorLimitExceeded.	The service request would exceed a switching function limit on the number of monitors (either an overall limit on the aggregate number of monitors been exceeded. The limit should be configured.
SubscribedResourceAvailability	generic	Internal resource error.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) If filtering of the individual Private Events is desired, then the CSTA Private Data Information (privateData parameter) is not used.
- 3) Monitoring is only guaranteed for devices in the switching sub domain.
- 4) OpenScape 4000 supports only device monitoring. So if the call leaves the monitored device the monitoring can't be continued.
- 5) An ACD agent is not a device and can therefore not be monitored. An ACD agent ID in the monitorObject parameter is not allowed. An ACD agent device is a device where agents logon. ACD agent devices are monitorable stations.
- 6) An ACD group is a non monitorable logical device but the corresponding ACD Route Control Group (RCG) is monitorable.
- 7) Dialling number "\*888" as requested monitorObject activates monitoring of all RCGs with one crossRefIdentifier.
- 8) Multiple monitors may exist for the same device or group. Each monitor may have a unique event filter. This should be used with care, because it may result in duplicated events among the OpenScape 4000, the computer interface, and additional internal traffic. It should be verified that the particular system has been engineered to support this increased throughput.
- 9) Monitors exist only during an application association. If the application association is terminated, all active monitors established during that association are stopped. Additional information see [Section 3.6.3, "Monitor Stop"](#)
- 10) To enable monitoring the sub-application number must be configured in OpenScape 4000. Note: There are some additional OpenScape 4000 parameters which may influence the device monitoring (check the Service Manual for further details).
- 11) The limit on the number of monitors depends on licence.

**12) Miscellaneous Characteristics:**

- CallIDOnly - The switching function does not accept the CallID only format of the Connection ID for this service.
- Switching function default for monitor-type is device-type.

**3.6.3 Monitor Stop**

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.

**Request Message****Table 39: Monitor Stop Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
crossRefIdentifier	MonitorCrossRefID	M	This specifies which monitor to cancel. This parameter is the value that was returned in the Monitor Start (positive) response of the monitor to be canceled.

**Positive Response**

Since the Monitor Stop service positive response does not contain any service specific parameters, no table is provided.

**Negative Response**

The negative acknowledgement error codes sent by the computing domain are application specific and will be ignored by OpenScape 4000. An entry in the error log will be generated.

If a Monitor Stop service is requested by the computing domain the following negative responses from OpenScape 4000 CSTA are possible:

**Table 40: Monitor Stop Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request
SystemResourceAvailability	generic	Internal resource error
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	resourceBusy	The request has been rejected because another request for the same subscriber is already active. The request could not be executed now because of temporary system congestion.
	overallMonitorLimitExceeded	The service request would exceed a switching function limit on the number of monitors (either an overall limit on the aggregate number of monitors been exceeded. The limit should be configured.

#### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The switching function may issue a Monitor Stop service when it can no longer provide information. This may occur are:
  - The OpenScape 4000 is restarting.
  - If the monitor object (device) leaves the switching sub-domain via configuration.
- 3) Miscellaneous Characteristics:
  - The switching function generates this service request.
  - The switching function supports receiving this service request.

## 3.7 Snapshot Services

### Snapshot Services Summary

**Table 41: Support of Snapshot Services**

Snapshot Services	Service Description Section
Snapshot Call	<a href="#">Section 3.7.1</a>

Snapshot Services	Service Description Section
Snapshot CallData	<a href="#">Section 3.7.2</a>
Snapshot Device	<a href="#">Section 3.7.3</a>
Snapshot DeviceData	<a href="#">Section 3.7.4</a>

### Snapshot Services Descriptions

The entire ECMA CSTA III standard covering Snapshot Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

## 3.7.1 Snapshot Call

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.

### Request Message

**Table 42: Snapshot Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
snapshotObject	ConnectionID	M	This indicates the connectionID of the call to be snapshot.

## Positive Response

Table 43: Snapshot Call Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
snapshotData	List of Structures	C	<p>Specifies information for each endpoint in a call.</p> <p>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.</p> <p>The complete set of possible information is:</p> <ul style="list-style-type: none"> <li>• deviceOnCall (M) DeviceID - Of a device involved with the endpoint.</li> <li>• connectionIdentifier (C) ConnectionID - For the endpoint. This is mandatory if the endpoint is in the switching sub-domain, otherwise it is optional.</li> <li>• localConnectionState (O) LocalConnectionState - For the endpoint.</li> <li>• servicesPermitted (C) ServicesPermitted - This is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange service).</li> <li>• mediaServiceInformationList (O) List of Structure - Specifies information about the media services that are attached (bound to) the connection in the call. The complete set of possible information is: <ul style="list-style-type: none"> <li>mediaServiceType (M) MediaServiceType - A media service type that has been bound to the connection.</li> <li>mediaServiceVersion (O) Value. The version of the media services.</li> <li>mediaServiceInstance (O) MediaServiceInstanceID - A media service instance associated with the media service bound to the connection.</li> <li>mediaStreamID (C) MediaStreamID - The media stream identifier for the media service binding. This shall be provided if the switching function supports providing the mediaStreamID as indicated by the capability exchange services.</li> <li>connectionInformation (O) ConnectionInformation - The connection information associated with the callIdentifier connection.</li> </ul> </li> </ul>

Parameter Name	Content	M/C/O	Comments
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.
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.
callLinkageData	CallLinkageData	C	Specifies the global call data and thread data associated with the call. This parameter is mandatory if the switching function supports the call linkage feature otherwise it shall not be provided.

## Negative Response

**Table 44: Snapshot Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidConnectionID	Connection identifier or some component of the connection identifier is invalid.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000.</li> </ul>

**Usage Notes**

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScope 4000 devices"](#).
- 2) In general, OpenScope 4000 CSTA supports SnapshotCallInformation in the positive response of SnapshotCall. In case SnapshotCall request is for multiline devices, Snapshot CallData is invoked in segments due to the size of the message:
  - For XML Long tag interface, if more than 10 appearances are reported.
  - For XML Short tag interface, if more than 20 appearances are reported.
- 3) This service does not affect calls at the specified device.
- 4) Values for ServicesPermitted are only generated, if a monitor is set on the requested object.
- 5) The connection ID provided in the service must consist of a CallID and a DeviceID.
- 6) The device numbers of Trunks, General Attendants, Hunt Groups and RCGs must be encoded using the appropriate high-order byte encoding as specified in Table 5-1 on page 5-5, or a negative response is returned.
- 7) Miscellaneous Characteristics
  - CallIDOnly - The switching function does not accept or support the CallID only format of the Connection ID for this service.
  - The switching function does not use the Snapshot CallData service to report the requested information.

**3.7.2 Snapshot CallData**

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 acknowledgment). See Usage Notes below.

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.

**Request Message****Table 45: Snapshot CallData Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
serviceCrossRefID	ServiceCrossRefID	M	Specifies the reference used to associate the Snapshot CallData service messages to the Snapshot Device service request.

Parameter Name	Content	M/C/O	Comments
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	<p>Specifies if this segment is the last one associated with the serviceCrossRefID. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• TRUE - Indicates that this is the last segment;</li> <li>• FALSE - Indicates that this is not the last segment in the sequence.</li> </ul>

Parameter Name	Content	M/C/O	Comments
snapshotData	List of Structures		<p>Specifies information for each call at a device.</p> <p>The complete set of information is:</p> <ul style="list-style-type: none"> <li>• connectionIdentifier (M) ConnectionID</li> <li>• localCallState (M) Choice Structure This will be one of the following choices:</li> </ul> <p>compoundCallState (List of LocalConnectionStates) - consists of a sequence of local connection states.</p> <p><del>simpleCallState (SimpleCallState) - the simple call state.</del></p> <p>unknown</p> <ul style="list-style-type: none"> <li>• servicesPermitted (C) ServicesPermitted - mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services).</li> <li>• mediaServiceInformationList (O) List of Structures - specifies information about the media services that are attached (bound) to the connection in the call. The complete set of possible components include:</li> </ul> <p><del>mediaStreamID (C) MediaStreamID - the media stream identifier for the media service binding. This shall be provided if the switching function supports providing the mediaStreamID as indicated by the capability exchange servers.</del></p> <p>connectionInformation (O) ConnectionInformation - the connection information associated with the callIdentifier connection</p> <ul style="list-style-type: none"> <li>• MediaCallCharacteristics (O) MediaCallCharacteristics specifies the media class and data characteristics of the call.</li> </ul>

Parameter Name	Content	M/C/O	Comments
privatedata	CSTAPrivateData		<p>Specifies non-standardized information.</p> <p>OpenScape4000:</p> <ul style="list-style-type: none"> <li>• numberOfQueuedCalls, if the snapshot is for an ACD Group number</li> <li>• mobileUserDirectoryNumber, if the device has mobile User feature active</li> </ul> <p>See <a href="#">Chapter 9, "Appendix C - Private Data"</a>.</p>

**Positive Response**

There is no positive acknowledgment associated with this service request.

**Negative Response**

The negative acknowledgment error codes sent by the computing domain are application specific and will be ignored by OpenScape 4000. An entry in the error log will be generated.

**Usage Notes**

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) As mentioned for Snapshot Call request, Snapshot CallData might be generated only for multiline devices.

### 3.7.3 Snapshot Device

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.

**Request Message****Table 46: Snapshot Device Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
SnapshotObject	DeviceID	M	The Device ID is the directory number of the snapshot device or the device number of a trunk (regular trunk or a trunk configured as a Prompt Response IVR (with APRI) Server port.

**Positive Response****Table 47: Snapshot Device Service Positive Response**

Parameter Name	Content	M/C/O	Comments
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.

**Negative Response****Table 48: Snapshot Device Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
System resource availability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <p>No link to OpenScape 4000</p>

#### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) This service does not affect calls at the specified device.
- 3) OpenScape 4000 CSTA supports only SnapshotDataInformation in one or more messages using the Snapshot DeviceData Service.
- 4) The service supports ACD-Groups. In this case the number of queued calls are reported.
- 5) The device numbers of Trunks, General Attendants, Hunt Groups, RCGs and ACD Groups must be encoded using the appropriate high-order byte encoding as specified in Table 5-1 on page 5-5, or a negative response is returned.
- 6) This service reports activity on the prime line of a device. Device information can be requested for an appearance/phantom line by specifying the appearance or phantom line's directory number in the request.
- 7) Miscellaneous Characteristics
  - The switching function uses the Snapshot DeviceData service to report the requested information.

### 3.7.4 Snapshot DeviceData

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.

## Request Message

Table 49: Snapshot DeviceData Service Request Parameters

Parameter Name	Content	M/C/O	Comments
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"> <li>• TRUE - Indicates that this is the last segment</li> <li>• FALSE - Indicates that this is not the last segment in the sequence.</li> </ul>

Parameter Name	Content	M/C/O	Comments
snapshotData	List of Structures	M	<p>Specifies information for each call at a device.</p> <p>This complete set of information is:</p> <ul style="list-style-type: none"> <li>connectionIdentifier (M) ConnectionID</li> <li>localCallState (M) Choice Structure - This shall be one of the following choices:  compoundCallState (List of LocalConnectionStates) - This consists of a sequence of local connection states.  simpleCallState (SimpleCallState)- The simple call state.  unknown</li> <li>servicesPermitted (C) ServicesPermitted - This is mandatory if the switching function supports the Dynamic Feature Availability option (as indicated through the capabilities exchange services).</li> <li>mediaServiceInformationList (O) List of Structures - Specifies information about the media services that are attached (bound to) the connection in the call. The complete set of possible components include:  mediaStreamID (C) MediaStreamID -The media stream identifier for the media service binding. This shall be provided if the switching function supports providing the mediaStreamID as indicated by the capability exchange services.  connectionInformation (O) ConnectionInformation - The connection information associated with the callIdentifier connection</li> <li>MediaCallCharacteristics (O) MediaCallCharacteristics - specifies the media class and data characteristics of the call.</li> </ul>

Parameter Name	Content	M/C/O	Comments
privateData	CSTAPrivateData		<p>Specifies non-standardized information.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>• numberOfQueuedCalls if the snapshot is for an ACD Group number</li> <li>• mobileUserDirectoryNumber if device has mobile User feature active</li> </ul> <p>(see <a href="#">Chapter 9, "Appendix C - Private Data"</a>)</p>

#### Positive Response

There is no positive acknowledgment associated with this service request.

#### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific and will be ignored by OpenScape 4000. An entry in the error log will be generated.

#### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) When a device is idle, the local connection state is null and no call ID is included in the connection ID.
- 3) If the snapshot addresses an ACD group number, the private data parameter contains the number of calls queued at the ACD group.
- 4) If the snapshot device has its Mobile User Feature active, the private data parameter contains the Mobile User Directory Number.
- 5) Values for ServicesPermitted are not generated.

## 3.8 Call Control Services

### Call Control Services Summary

**Table 50: Support of Call Control Services**

Call Control Services	Service Description Section
Accept Call	<a href="#">Section 3.8.1</a>
Alternate Call	<a href="#">Section 3.4.3</a>
Answer Call	<a href="#">Section 3.8.3</a>
Call Back Call-Related	<a href="#">Section 3.8.4</a>
Clear Connection	<a href="#">Section 3.8.5</a>

Call Control Services	Service Description Section
Conference Call	<a href="#">Section 3.8.6</a>
Consultation Call	<a href="#">Section 3.8.7</a>
Deflect Call	<a href="#">Section 3.8.8</a>
Dial Digits	<a href="#">Section 3.8.9</a>
Hold Call	<a href="#">Section 3.8.10</a>
Make Call	<a href="#">Section 3.8.11</a>
Make Predictive Call	<a href="#">Section 3.8.12</a>
Reconnect Call	<a href="#">Section 3.8.13</a>
Retrieve Call	<a href="#">Section 3.8.14</a>
Single Step Transfer Call	<a href="#">Section 3.8.15</a>
Transfer Call	<a href="#">Section 3.8.16</a>

### Call Control Services Descriptions

The entire ECMA CSTA III standard covering Call Control Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

## 3.8.1 Accept Call

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.

### Request Message

**Table 51: Accept Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
callToBeAccepted	ConnectionID	M	Specifies the connection to be accepted.

### Positive Response

**Since the Accept Call service positive response does not contain any service specific parameters, no table is provided.**

## Negative Response

**Table 52: Accept Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidCallIdentifier	Call ID contains an invalid value.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 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 Accept Call service for the alerting or queued call.

## 3.8.2 Alternate Call

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).

### Request Message

**Table 53: Alternate Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
heldCall	ConnectionID	M	Specifies the held connection for the alternating device.
activeCall	ConnectionID	M	Specifies the active connection for the alternating device.

**Positive Response**

Since the Alternate Call service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 54: Alternate Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

**Usage Notes**

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Alerting as initial state for heldCall is not supported.
- 3) If the alternating party is connected to a call that is not using voice service (e.g. a data call), a negative response is returned.
- 4) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - Does not support Offered Mode of Alerting.
  - Service Request Acknowledgement Model (Atomic [FALSE]).

**3.8.3 Answer Call**

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.

### Request Message

**Table 55: Answer Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
callToBeAnswered	ConnectionID	M	Specifies the connection to be answered.

### Positive Response

Since the Answer Call service positive response does not contain any service specific parameters, no table is provided.

### Negative Response

**Table 56: Answer Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	InvalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The answering device must be a supported auto-answer device as stipulated in [Section 3.1, "General Notes"](#).

- 3) If the answering Device ID refers to an extension number that appears on more than one device (multiple appearance), the Answer Call service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active (that is, the Connection State is *connected*) on that device.
- 4) Queued and Initiated as initial state for callToBeAnswered are not supported.
- 5) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - Does not support Offered Mode of Alerting.
  - Service Request Acknowledgement Model (Atomic [FALSE])

### 3.8.4 Call Back Call-Related

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.

#### Request Message

**Table 57: Call Back Call-Related Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
callbackConnection	ConnectionID	M	Specifies the call back connection at the calling device.

#### Positive Response

**Table 58: Call Back Call-Related Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
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.

#### Negative Response

**Table 59: Call Back Call-Related Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	invalidConnectionID	Connection identifier or some component of the connection identifier is invalid.  - The connectionID includes CallID only, DeviceID only or an unsupported deviceID type.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the subject device.
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
	noActiveCall	The service request operates on an active call, but there was no active call.
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Only one Call Back service request (Call-Related) can be outstanding for any calling and called device pair. It is a OpenScape 4000 option (as indicated by the capability exchange services) if additional Call Back service requests (Call-Related) for that pair result in a positive acknowledgement from the switching function.
- 3) Null as initial state for the target device is not supported.
- 4) In the case where a call is forwarded from a called device (busy forwarding, for example), the call back request is placed on the device that the call was forwarded to.
- 5) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - Additional Call Back Call-Related services for the same calling and called devices does not result in a negative acknowledgement.
  - Service Request Acknowledgement Model (Multi-Step [TRUE]).

### 3.8.5 Clear Connection

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.~~

#### Request Message

**Table 60: Clear Connection Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
connectionToBeCleared	ConnectionID	M	Specifies the connection to be cleared.

#### Positive Response

Since the Clear Connection service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 61: Clear Connection Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	Generic	Default or internal error
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	invalidConnectionID	Connection identifier or some component of the connection identifier is invalid. <ul style="list-style-type: none"> <li>The connectionID includes CallID only, DeviceID only or an unsupported deviceID type.</li> </ul>
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
State incompatibility	Generic	Internal resource error.
	noConnectionToClear	There is no connection for the connection identifier specified as the connectionToBeCleared. <ul style="list-style-type: none"> <li>The application did not find a connection associated with clearing party.</li> </ul>
	invalidObjectState	Object is in an incorrect state for the service.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
System resource availability	Generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The Clear Connection service will not automatically retrieve a held call. If an application wants to clear an active call and retrieve a currently held call, the Reconnect service must be used instead of Clear Connection.
- 3) If there is a held or queued call at the clearing device, the clearing device will be recalled after the transition to null.
- 4) The Clear Connection service is allowed when the clearing party device has a parked call. The parked call is not affected if the clearing party device is the party that parked the call.
- 5) If the clearing party is an analog device and has a call camped on to it, then the Clear Connection request is not allowed.
- 6) The Clear Connection service results in a negative acknowledgement if the clearing party is connected to a conference and has a call on soft hold.
- 7) The Clear Connection service results in a negative acknowledgement if the clearing party has consulted or picked up a call and now has a conference on hold.
- 8) If the clearing device's Device ID refers to an extension that appears on more than one device (multiple appearance), the Clear Connection service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active (that is, the Connection State is *connected*) on that device. If these conditions are not met, a negative response is returned. In addition, if an appearance of the primary device is connected into the call (bridged conference), a negative response is returned.
- 9) If the primary device is ringing, the other appearances of that line are also cleared.
- 10) The clearing party cannot be the target party of an executive override.
- 11) After the successful execution of the Clear Connection service, the clearing device and any non-conferenced devices in the call transition from connected, to failed, and to null. Depending on the device type(s), the following occurs:
  - Analog devices transition from connected, to failed (receive a blocked tone) and are inaccessible to other services. These devices remain inaccessible until the user goes on-hook.

- Digital devices using the handset transition from connected, to failed (receive a blocked tone), and transition to null after a time-out or going on-hook.
  - Digital devices using the headset or speakerphone will transition from connected to failed and then to null immediately.
- 12)** A held or queued connection can only be cleared if there is no other active connection at the clearing device.
- 13)** The Clear Connection service results in a negative acknowledgement if the clearing party has more than one connection and the service request was initiated with DeviceIDOnly.
- 14)** Miscellaneous Characteristics:
- DeviceIDOnly -The switching function accepts and supports the DeviceID only format of the Connection ID for this service.
  - Service Request Acknowledgement Model (Atomic [FALSE])

### 3.8.6 Conference Call

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.

#### Request Message

**Table 62: Conference Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
heldCall	ConnectionID	M	Specifies the held connection.
activeCall	ConnectionID	M	Specifies the active connection.

#### Positive Response

**Table 63: Conference Call Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
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.

## Negative Response

Table 64: Conference Call Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	InvalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	conferenceMemberLimitExceeded	Executing the service request would exceed the limit on the number of devices that may connect to a call.
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) There is a limit of eight parties in a conference call, of which only one can be an attendant.
- 3) Prior to the conference, the following connection states must exist or a negative response is returned:
  - Held connection and connected connection at the conferencing device.
  - The connection at the held device must be in the connected state.
  - The connection at the active device must be in the connected state.
- 4) If the conferencing device's Device ID refers to an extension number that appears on more than one device (multiple appearances), the Conference Call service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active on that device.

- 5) To add a party to an existing conference, the conferencing device must have a conference on consultation hold or on background hold due to a call pickup, and an active call when the Conference Call service is initiated.
- 6) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - Service Request Acknowledgement Model (Atomic [FALSE]).

### 3.8.7 Consultation Call

This 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.

#### Request Message

**Table 65: Consultation Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
existingCall	ConnectionID	M	Specifies the active connection.
consultedDevice	DeviceID	M	Specifies the device to be consulted.
userData	UserData	O	Specifies the user data to be sent to parties in the call.

#### Positive Response

**Table 66: Consultation Call Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
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.

#### Negative Response

**Table 67: Consultation Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error.
	invalidConnectionID	Connection identifier or some component of the connection identifier is invalid.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	invalidCallingDeviceIdentifier	The calling device parameter is invalid.
	invalidCalledDeviceIdentifier	The called device parameter is invalid.
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <ul style="list-style-type: none"> <li>Device ID contains an unsupported device ID type.</li> <li>The service is not allowed for the specified digital device because the Transfer key is not configured.</li> </ul>
StateIncompatibility	invalidObjectState	<p>Object is in an incorrect state for the service.</p> <ul style="list-style-type: none"> <li>The consulting party is busy on a secondary line.</li> <li>Another device is using an appearance of the consulting party's line.</li> <li>The consulted Device ID is the same Device ID as the consulting device or the party to be held.</li> </ul>
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.
	resourceLimitExceeded	<p>The service is supported by the server, but is unavailable because it would exceed the internal usage limit of the resource.</p> <ul style="list-style-type: none"> <li>There are no Generic Data buffers to handle userData</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Prior to the consultation, the connection at the originally called device must be connected or a negative response is returned. This existing connection may be a conference call.
- 3) After the successful execution of the consultation, the active connection at the consulting device will be in the connected state, and the consultation-held connection at the consulting device will be in the hold state.
- 4) After successful execution of the consultation, the connection at the consulted device will be alerting, queued, or null (if a feature is active such as call forwarding immediate).

- 5) If the active call has been placed on consultation hold, but the OpenScape 4000 is unable to extend the call to the called device, the application must use the Reconnect service to retrieve the party on consultation hold.
- 6) A consultedDevice (dialingNumber) of zero length, or the partial dialing character ';' only, or a ';' at the end of the dialing number, indicates partial dialing. This means a multi stage dialling. The completion of the dialling sequence can be accomplished either by entering rest of the sequence manually at the actual device or computing function can use the dial digit service.
- 7) OpenScape 4000 has a time-out period for multi-stage dialling. If the dialling sequence does not complete prior to this time-out, it aborts the call and issues the Connection Cleared event.
- 8) If the consulting device's Device ID refers to an extension number that appears on more than one device (multiple appearance), the Consultation Call service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active (that is, the Connection State is *connected*) on that device.
- 9) If user data (supported length see [Section 3.4.3, "Get Switching Function Capabilities"](#)) is sent in the service request and the called device is in another switching domain, the user data is sent to that switching domain if it is accessed by an ISDN (PRI) trunk or a CorNet-N trunk.
- 10) To handle the user data OpenScape 4000 needs a sufficient number of configured Generic Data buffers.
- 11) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - MultiStage - The switching function supports multistage dialling with this service.
  - The switching function does not support adjustment of the media characteristics.
  - Service Request Acknowledgement Model (Atomic [FALSE]).

### 3.8.8 Deflect Call

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.

#### Request Message

**Table 68: Deflect Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
callToBeDiverted	ConnectionID	M	Specifies the connection to be diverted.

Parameter Name	Content	M/C/O	Comments
newDestination	DeviceID	M	Specifies the device to which the call is to be diverted (newDestination device).  OpenScape 4000: Next to directory numbers the following device types are supported: <ul style="list-style-type: none"> <li>• RCG</li> <li>• General Attendant</li> <li>• Hunt Group</li> </ul>
userData	UserData	O	Specifies the user data to be sent to parties in the call.

**Positive Response**

Since the Deflect Call service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 69: Deflect Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error
	invalidConnectionID	Connection identifier or some component of the connection identifier is invalid.
	invalidDestination	The service request contains a destination that is invalid.
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <ul style="list-style-type: none"> <li>• For this service it might also mean that the connection between the diverted party and the new destination is not allowed.</li> </ul>

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	invalidAllocationState	<p>The service request (MakePredictiveCall) specified an allocation state that is invalid in the present circumstance.</p> <ul style="list-style-type: none"> <li>For this service it might also mean that connection to be diverted is a device in a call state that does not support the Deflect Call service.</li> </ul>
State incompatibility	generic	Internal resource error
	invalidObjectState	<p>Object is in an incorrect state for the service.</p> <ul style="list-style-type: none"> <li>For this service it might also mean that a Route Request service is already in progress for the connection to be diverted.</li> </ul>
	noActiveCall	The service request operates on an active call, but there was no active call.
System resource availability	generic	<p>Internal resource error</p> <ul style="list-style-type: none"> <li>The communications server configuration limit on the number of generic buffers for user data has been exceeded. The limit should be reconfigured.</li> </ul>
	resourceBusy	<p>The service is supported by the server, but is unavailable due to a resource that is busy.</p> <ul style="list-style-type: none"> <li>(e.g. all trunk resources busy)</li> <li>It could also mean that OpenScape 4000 is temporarily congested or that another service is still in progress.</li> </ul>
	resourceLimitExceeded	<p>The service is supported by the server, but is unavailable because it would exceed the internal usage limit of the resource.</p> <ul style="list-style-type: none"> <li>There are no Generic Data buffers to handle userData</li> </ul>
	deviceOutOfService	A device that is needed to carry out the service is out of service.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>• (e.g. the new destination device is out of service or not configured correctly)</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) For this service the initial states of the connection to be diverted can be
  - alerting (not for Hunt Group)
  - queued
- 3) Deflection of calls at an RCG is only possible if the calling or the RCG device itself is monitored.
- 4) To divert a call at an alerting ACD Route Control Group with an application, a delay of the internal ACD must be configured in OpenScape 4000.
- 5) The maximum number of re-deflecting the new destination device again can be configured in OpenScape 4000.
- 6) Re-deflecting to the originally called device or to the last redirection device is not possible. However, if the new destination is an ACD number or DNI number, this restriction is not applied.
- 7) If the new destination device is busy, a negative response is returned.
- 8) Some of the above restrictions are not considered in the service permitted indication of events.
- 9) If in OpenScape 4000 re-routing of trunks is configured the event flow after successful call deflection might differ from the normal event flow.
- 10) If the UserData (supported length see [Section 3.4.3, "Get Switching Function Capabilities"](#)) exceeds the maximum length it will get ignored.
- 11) If user data is sent in the service request and the new destination device is in another switching domain, the user data is sent to that switching domain if it is accessed by an ISDN (PRI) trunk or a CorNet-N trunk.
- 12) To handle the user data OpenScape 4000 needs a sufficient number of configured Generic Data buffers.
- 13) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - Service Request Acknowledgement Model (Multi-Step [TRUE])
- 14) In netwide cases where the destination is out of the switch, then the switch delivers positive response. However it is possible to configure the CSTA so that it delivers the real result based on the cause of next event from the switch. In case the divert is from an RCG then the proposed config of the corresponding Connectivity Adapter is: CSTA3\_DELAY\_DEFLECT\_CALL\_RESP=1 In other cases (digital, analog subscribers, HGs etc) the following settings can be used but it is working only from OpenScape 4000 V6: CSTA3\_DELAY\_DEVICE\_DEFLECT\_CALL\_RESP=1

### 3.8.9 Dial Digits

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.

#### Request Message

**Table 70: Dial Digits Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
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.  OpenScape 4000: This parameter is always interpreted as partial dialling sequence. The ";" character is not allowed.

#### Positive Response

Since the Dial Digits service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 71: Dial Digits Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Successive requests to dial digits will be rejected until the active request is complete.
- 3) The diallingSequence parameter must be in Diallable Digits format.
- 4) The diallingSequence parameter containing a ";" results in a negative acknowledgement.
- 5) The diallingSequence parameter is always interpreted as partial dialling sequence.
- 6) OpenScape 4000 has a time-out period for multi-stage dialling. If the dialling sequence does not complete prior to this time-out, it aborts the call and issues the Connection Cleared event.
- 7) If the connection state at the calling party is not in initiated state, a negative response is returned.
- 8) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function accepts or supports the DeviceID only format of the Connection ID for this service.
  - Service Request Acknowledgement Model (Multi-Step [TRUE]).

## 3.8.10 Hold Call

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.

### Request Message

**Table 72: Hold Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
callToBeHeld	ConnectionID	M	Specifies the active connection to be held.

### Positive Response

Since the Hold Call service positive response does not contain any service specific parameter, no table is provided.

## Negative Response

Table 73: Hold Call Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Hold Call service places an active call on hold. OpenScape 4000 does not allow new service initiation.
- 3) This service interrupts communication for an existing call at a device. The relationship between the holding device and the held call is maintained until the call is retrieved from the held state or until the call is cleared. There is no time-out period for the call once it is held.
- 4) Only one call can be put on hold using the Hold Call service. While this call remains on hold, subsequent Hold Call service requests are denied with a negative response.
- 5) This service only supports devices that are configured as keysystem (basic two party calls, consulted party and conference) or anate (basic two party calls only).

- 6) For digital devices, this feature invokes the Manual Hold station feature. The device must have Manual Hold feature configured or a negative response is returned. Manual Hold means:
  - The line cannot be used to make a second call after a call has been placed on hold (although a call can be made on another line appearance or on a phantom line).
  - The following call types cannot be placed on manual hold: paging access, PhoneMail, or attendant.
  - If the user has the Automatic Privacy COS, when manual hold is invoked, the automatic privacy feature is released (and activated when the held call is retrieved).
  - If the user has a camped-on call (queued state) or a call on background hold because of a pick-up feature, the active call cannot be put on hold.
- 7) For analog devices, this service invokes the Hard Hold (Call Hold) station feature. Hard Hold means that:
  - The line can be used to originate or receive other calls after a call has been placed on hold (that is, after the hold, the device is in the initiated state)
  - Only the device that held the call can retrieve it.
- 8) If the holding device's Device ID refers to an extension number that appears on more than one device (multiple appearance), the Hold Call service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active (that is the Connection State is connected) on that device.
- 9) This call must be in the connected state on the devices' primary line or a negative response is returned. In a consultation call scenario, the holding device may be the consulted party. In a conference call scenario, the holding device must be a conference member or the party who was consulted by a conference member.
- 10) For digital stations, the user receives the same indications as with the use of the Manual Hold key; namely, LEDs, displays, and tones. The successful execution of this service results in the same event sequence as if a call has been placed on hold through the use of the following station features (feature interactions and allowed "held party" types are consistent too):
  - Manual Hold for digital devices.
  - Hard Hold (Call Hold) for analog devices.
- 11) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - Service Request Acknowledgement Model (Atomic [FALSE]).

### 3.8.11 Make Call

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.

### Request Message

**Table 74: Make Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
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.
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"> <li>• Prompt (default)</li> <li>• Do Not Prompt (auto originate)</li> </ul>

### Positive Response

**Table 75: Make Call Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
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).

## Negative Response

Table 76: Make Call Error Service Response Error Codes

CSTA Error Codes		Possible Cause
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error
	requestIncompatibleWithObject	DeviceID contains an unsupported device ID type  OpenScape 4000: The calling device is configured for the hotline feature or off-hook alarm feature.
	invalidCallingDeviceIdentifier	The calling device parameter is invalid.
	invalidCalledDeviceIdentifier	The called device parameter is invalid.
State incompatibility	invalidObjectState	Object is in an incorrect state for the service.  OpenScape 4000: The calling device is a hunt group (with or without a master device) and none of the supported devices in the hunt group is idle, or all of the supported devices are out of service.  The calling device is currently active on a non-primary line.  The calling device ID is the same as the called device ID.
System resource availability	generic	Internal resource error
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  • No link to OpenScape 4000

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) If the calling device's Device ID refers to an extension number that appears on more than one device (multiple appearance), the Make Call service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active on that device.

- 3) The called device in a Make Call request via dialling number is restricted to the following types, if not, a negative response is returned:
  - Internal user (including attendant)
  - Private network user
  - External number
  - ACD number
  - Hunt group number
  - PhoneMail
  - IVRs
- 4) The called device in a Make Call request via device number is restricted to the following types, if not, a negative response is returned:
  - ACD Route Control Group (RCG)
  - General Attendant
  - Hunt Group
- 5) The calledDirectoryNumber parameter may contain either a valid device identifier or the character ";". This way computing function indicates that it wishes to stage the dialing sequence. The completion of the dialing sequence can be accomplished either by entering rest of the sequence manually at the actual device or the computing function can use the Dial Digits service to complete a sequence.
- 6) Calling device:
  - Digital calling device may only be in the null connection state or a negative response is returned.
  - Analog calling devices may also be in the hold connection state in another call, in addition to the null connection state.
  - Call pickup, call queueing, ringer cutoff, Do Not Disturb, and abbreviated dialing features are not honored for the calling device.
  - Call forwarding for the calling device is not honored. The call is presented at the calling device regardless of the activation of any type of call forwarding feature.
- 7) If the autoOriginate parameter has a value of "Do Not Prompt", and if the calling device is not capable of automatically answering the device, then this parameter is ignored and the call is prompted at the calling device.
- 8) Rules for pilot hunting (applies to the calling device only):
  - There are several types of call hunt groups in the OpenScape 4000. Only Hunt Groups with a group access code are supported for the calling device.
  - In the case of a positive response, the directory number of the alerted hunt group member is sent as the calling device in the response.
  - If no device is available, a negative response is returned.
  - The auto-answer attribute of a device is not part of the selection criteria for selecting a device from a hunt group.

- 9) The impact of time constraints.
  - The amount of time to wait for a party to answer is known as time monitoring. The calling device always is subjected to time monitoring. If auto-answer is invoked, the timer will not expire.
  - If the calling device does not answer within the specified time, prompting is discontinued.
  - Time monitoring for the calling device is controlled by a value provided in the OpenScape 4000 configuration.
  - OpenScape 4000 has a time-out period for multi-stage dialling. If the dialling sequence does not complete prior to this time-out, it aborts the call and issues the Connection Cleared event.
- 10) When the calling device, with display, is alerted but not configured for auto answer, it displays *Computer initiated call* until it is answered or until prompting is discontinued.
- 11) Miscellaneous Characteristics:
  - MultiStage - The switching function supports multistage dialling with this service.
  - Prompting - The switching function supports prompting for the callingDevice
  - Prompting Mode - Prompting is part of the execution of the service.
  - Offhook - The switching function does not support the performing of a Make Call while the callingDevice is off-hook.
  - The switching function can not adjust the media characteristics of the call.
  - Service Request Acknowledgement Model (Multi-Step [TRUE])

### 3.8.12 Make Predictive Call

The Make Predictive Call service shall originate a call between two devices by first creating a connection to the called device. The service returns a positive acknowledgement that provides the connection at the called device.

Subsequent actions are taken depending upon the call progress and the actions requested. Examples are:

- An attempt is made to connect at the calling device because a connected state was determined at the called device.
- The call was cleared because a connected state was determined at the called device and the switching function detected a FAX and was instructed by the computing function to clear the call in this case.

#### Request Message

**Table 77: Make Predictive Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
callingDevice	DeviceID	M	Specifies the device on behalf of which the call is originated.
calledDirectoryNumber	DeviceID	M	Specifies the called device.

## Positive Response

Table 78: Make Predictive Call Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
initiatedCall	ConnectionID	M	Specifies the initial connection between the called device and the call. The ConnectionID shall have the CallID of the resulting new call and the DeviceID representing the called device.

## Negative Response

Table 79: Make Predictive Call Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidCalledDeviceIdentifier	The called device parameter is invalid.
	invalidCallingDeviceIdentifier	The calling device parameter is invalid.
	requestIncompatibleWithObject	DeviceID contains an unsupported device ID type
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceLimitExceeded	The service is supported by the server, but is unavailable because it would exceed the internal usage limit of the resource.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).

- 2) The called device in a Make Predictive Call request via dialling number is restricted to the following types, if not, a negative response is returned:
  - Internal user
  - Private network user
  - External number
- 3) For the called party:
  - Do Not Disturb and Call Forwarding Immediate will be honored.
  - Call Forwarding No Answer will not be honored.
  - No multiple diversion will be allowed, this means that only the first call forward will be executed.
- 4) If the calling party (RCG) is forwarded, the service is rejected.
- 5) Miscellaneous Characteristics:
  - Prompting - The switching function does not support prompting for the callingDevice.
  - Prompting Mode - Prompting is not part of the execution of the service.
  - The switching function does not generate a Service Initiated event for the calling device prior to any call activity at the calling device (i.e. reserve the calling device).
  - Service Request Acknowledgement Model (Multi-Step [TRUE])

### 3.8.13 Reconnect Call

The Reconnect Call service will clear a specified connection at the reconnecting device and retrieve a specified held connection at the same device.

#### Request Message

**Table 80: Reconnect Call Service Request Parameters**

Parameter Name	Content	M/C/ O	Comments
activeCall	ConnectionID	M	Specifies the connection to be cleared.
heldCall	ConnectionID	M	Specifies the held connection.

#### Positive Response

Since the Reconnect Call service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 81: Reconnect Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	invalidCallIdentifier	Call ID contains an invalid value.
	requestIncompatibleWithObject	Device ID contains an unsupported device ID type.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) For an analog device with calls on consultation hold and hard hold, the Reconnect service is only applied to the call on consultation hold. In that case, if the hard hold call is specified as the held call in the service request, a negative response is returned.
- 3) For a digital device with calls on consultation hold and on background hold, the Reconnect service is only applied to the call on consultation hold. In that case, if the background hold call is specified as the held call in the service request, a negative response is returned.
- 4) If the reconnecting device's Device ID refers to an extension number that appears on more than one device (multiple appearances), the Reconnect Call service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active (that is, the Connection State is *connected*) on that device.
- 5) If a phantom line is specified, the phantom line should be configured on at least one device. If not, a negative response is returned.
- 6) If the reconnecting device is connected to a call that is not using voice service (e.g. a data call), a negative response is returned.
- 7) If the reconnecting device is an overridden party, a negative response is returned.
- 8) If the reconnecting device is currently involved in a toggle, a negative response is returned.
- 9) Queued and Alerting as initial state for activeCall are not supported.

**10) Miscellaneous Characteristics:**

- DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
- Service Request Acknowledgement Model (Atomic [FALSE]).

**3.8.14 Retrieve Call**

The Retrieve Call service connects a specified held connection.

**Request Message****Table 82: Retrieve Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
callToBeRetrieved	ConnectionID	M	Specifies the held connection to be retrieved.

**Positive Response**

Since the Retrieve Call service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 83: Retrieve Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) This service only supports devices that are configured as keysystem (holding from basic two party calls, consulted party and conference) or anate (holding from two party calls only).
- 3) For digital devices, the Retrieve Call service works like the station feature that retrieves a call on manual hold with the following exceptions:
  - For non-single button mode, the service retrieves the retrieved call immediately by cutting through on the speaker or the headset. However, when the station feature manually retrieves, the user presses the line key, Preselected appears on the phone display, and then the user must go off-hook or push the speakerphone key.
  - Retrieval of a station parked call is not supported. A negative response is returned.
- 4) For analog devices, the Retrieve Call service works like the station feature that retrieves a call on hard hold.
- 5) If the retrieving device is an auto-answer device, the call must be in the held state or a negative response is returned. If the retrieving device is a non-auto-answer device, a call in the initiated state (device must be off-hook with dial tone present but with no dialed digits) must exist in addition to the call in the held state, or a negative response is returned.
- 6) If the retrieving device's DeviceID refers to an extension number that appears on more than one device (multiple appearance), the Retrieve Call service is only attempted on the device where the specified extension number is configured as its primary device, and only if no other appearances are active on that device.
- 7) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - Service Request Acknowledgement Model (Atomic [FALSE]).

### 3.8.15 Single Step Transfer Call

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.

#### Request Message

**Table 84: Single Step Transfer Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
activeCall	Connection ID	M	Specifies the connected or held connection in the call to be replaced.
transferredTo	DeviceID	M	Specifies the new called (transferredTo) device.
userData	UserData	O	Specifies the user data to be sent to parties in the call.
callFacilities (private)	List	O	Values: <ul style="list-style-type: none"> <li>SeamlessHandover: Specifies if the request must be executed seamlessly, i.e. without any additional audible or visible notification inbetween</li> <li>AutoAnswer: requests an automatic answer to the request</li> </ul>

#### Positive Response

**Table 85: Single Step Transfer Call Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
transferredCall	ConnectionID	M	Specifies the ConnectionID of the transferredTo device 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.

Parameter Name	Content	M/C/O	Comments
callFacilities (private)	List	O	Values: <ul style="list-style-type: none"> <li>SeamlessHandover: Specifies if the request must be executed seamlessly, i.e. without any additional audible or visible notification inbetween</li> <li>AutoAnswer: requests an automatic answer to the request</li> </ul>

### Negative Response

**Table 86: Single Step Transfer Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallIdentifier	Call ID contains an invalid value.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	invalidDestination	The service request contains a destination that is invalid.
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.
	resourceLimitExceeded	The service is supported by the server, but is unavailable because it would exceed the internal usage limit of the resource. <ul style="list-style-type: none"> <li>There are no Generic Data buffers to handle userData</li> </ul>
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

**Usage Notes**

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) If the transferring device's Device ID refers to an extension number that appears on more than one device (multiple appearance), the Single Step Transfer Call service is only attempted on the device where the specified extension number is configured as its primary device, and only if no other appearances are active on that device.
- 3) Prior to the Single Step Transfer, the local connection states of the active call at the transferring device and the transferred device must both be connected or a negative response is returned. This service is only supported for two-party calls; the active call cannot be a conference call. (Previously the OpenScape 4000 did not supported this feature from consultation call neither, but from OpenScape 4000 V4 it is possible).
- 4) If the Transfer-to device is outside the switching domain and is in one of the following conditions: busy, out-of-service, DND active, Invalid, or has class mark and administrative restrictions, a positive response is generated, but the transfer does not take place and no events are generated.
- 5) In the case of an All Trunks Busy condition in which the call cannot be transferred because there is no ISDN or CorNet trunk available, a positive response is generated, but the transfer does not take place and no events are generated.
- 6) In the above two cases it is possible to configure the CSTA so that it will deliver the response of the request execution based on the reality (the CSTA waits for the next ACL event, which indicates the right execution result in its cause). It is working with OpenScape 4000 V6 only with the following configuration settings of the corresponding Connectivity Adapter: `CSTA3_DELAY_SST_CALL_RESP=1`
- 7) If the transferredTo device is a CorNet or a PRI trunk, it is not immediately known if a trunk is available and a positive response is returned.
- 8) If the transferredTo device was call forwarded to another local party, the local Forwarded-To party will be indicated in the service response.
- 9) After successful execution of the Single Step Transfer, the connection at the transferred device will be still connected.
- 10) After the successful execution of the Single Step transfer, the transferring device will transition to failed and then to null. Depending on the device type(s), the following will occur:
  - Analog devices transition to failed (receive blocked tone) and are inaccessible to other services. These devices remain inaccessible until the user goes on-hook.
  - Digital devices using the handset transition to failed (receive blocked tone) and transition to null after a time-out or going on-hook.
  - Digital devices using the headset or speakerphone will transition to failed and then to null immediately.
- 11) If the transferredTo device is an extension or a personal attendant that is busy or is undergoing routine test (RTO), a negative response is returned.
- 12) To handle the user data OpenScape 4000 needs a sufficient number of configured Generic Data buffers.

**13) Miscellaneous Characteristics:**

- DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
- MultipleDevices - The switching function does not support transferring multiple devices or a conference call.
- Service Request Acknowledgement Model (Multi-Step [TRUE])

**3.8.16 Transfer Call**

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).

**Request Message****Table 87: Transfer Call Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
heldCall	ConnectionID	M	Specifies the held connection.
activeCall	ConnectionID	M	Specifies the active connection.

**Positive Response****Table 88: Transfer Call Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
transferredCall	ConnectionID	M	Specifies the ConnectionID of the transferredTo 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.

**Negative Response****Table 89: Transfer Call Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallIdentifier	Call ID contains an invalid value.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  - No link to OpenScape 4000
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Connections on manual, exclusive, background or hard hold cannot be transferred. Conferences on hold cannot be transferred.
- 3) Prior to the transfer, the transferring device must have a connection in the hold connection state, and a connection in the connected connection state, or a negative response is returned.
- 4) Prior to the transfer, the connection at the transferred-to device must be either connected, hold (background hold only), alerting, or queued (camp-on only), or a negative response is returned.
- 5) Prior to the transfer, the connection at the transferred device must be connected or a negative response is returned.
- 6) After the successful execution of the transfer, the connection at the transferred device will be still connected and the connection at the transferred-to-device will be connected, hold (background hold only), alerting, or queued (camp-on only).
- 7) After the successful execution of the transfer, the transferring device will transition to failed then to null. Depending on the device type(s), the following occur:
  - Analog devices transition to failed (receive blocked tone) and are inaccessible to other services. These devices remain inaccessible until the user goes on-hook.
  - Digital devices using the handset transition to failed (receive blocked tone) and then to null after a time-out or going on-hook.
  - Digital devices using the headset or speakerphone will transition to failed and then to null immediately.
- 8) If the transferring device's Device ID refers to an extension number that appears on more than one device (multiple appearances), the Transfer Call service is attempted only on the device where the specified extension number is configured as its primary device, and only if no other appearances are active on that device.

- 9) If the transferred-to device was call forwarded to another party, the forwarded-to party will be indicated in the service response.
- 10) Miscellaneous Characteristics:
- DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
  - MultipleDevices - The switching function does not support transferring multiple devices or a conference call.
  - Service Request Acknowledgement Model (Atomic [FALSE]).

## 3.9 Call Associated Feature Services

### Call Associated Feature Services Summary

Table 90: Support of Call Associated Feature Services

Call Associated Feature Services	Service Description Section
Generate Digits	<a href="#">Section 3.4.3</a>
Generate Telephony Tones	<a href="#">Section 3.9.2</a>
Send User Information	<a href="#">Section 3.9.3</a>

### Call Associated Feature Services Descriptions

The entire ECMA CSTA III standard covering call associated feature Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.9.1 Generate Digits

The Generate Digits service causes a series of digits to be sent on behalf of a connection in a call. The digits may be sent in the form of DTMF tones or rotary pulses. This service also supports optional parameters to control digit generation.

This service is used for generating end-to-end information that is to be sent to a device in a call (i.e., not to address/select a device).

This service does not affect the state or progress of a call.

#### Request Message

Table 91: Generate Digits Service Request Parameters

Parameter Name	Content	M/C/O	Comments
connectionToSendDigits	ConnectionID	M	Connection of the device which is generating the digits for the call.

Parameter Name	Content	M/C/O	Comments
digitMode	Enumerated	O	Specifies the signalling format. The complete set of possible values is: <ul style="list-style-type: none"> <li>rotaryPulse - rotary signalling.</li> <li>DTMF - DTMF signalling (default).</li> </ul>
charactersToSend	Characters (64)	M	Specifies the string of characters to send. Shall consist of the following set: <ul style="list-style-type: none"> <li>For rotary digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D</li> <li>For DTMF digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *, #, A, B, C, D</li> </ul> <p>A comma "," may be included in the parameter string to indicate a pause between characters. The length of the pause is switching function specific and may be determined using the capabilities exchange services.</p> <p>This parameter type is a character string. The maximum length supported by the switching function is provided via the capabilities exchange services.</p>

### Positive Response

Since the Generate Digits service positive response does not contain any service specific parameter, no table is provided.

A response is provided after initiation of this service, not at the completion.

### Negative Response

**Table 92: Generate Digits Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	Device ID contains an unsupported device ID type.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.  OpenScape 4000:  The object is not in a connected state.  The primary device associated with the object is currently active on a non-primary line.  Another device is using an appearance of the objects line.
SystemResourceAvailability	generic	Internal resource error
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The connection state of both devices in the call must be connected.
- 3) After digit generation has commenced, it cannot be interrupted.
- 4) Successive requests to generate digits will be rejected until the active request is complete.
- 5) Only DTMF mode is supported (pulse mode is not supported).
- 6) Valid characters for DTMF mode are '0' through '9', 'A', 'B', 'C', 'D', '\*', '#', and ',' (comma). A comma injects a two second delay between the signaling of the previous and next digits it separates. Multiple commas can be used to inject longer pauses.
- 7) A maximum of 44 ASCII characters is supported per request.
- 8) The duration of DTMF digits and interdigit spacing is determined by the switch according to the OpenScape 4000 configuration.
- 9) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function accepts or supports the DeviceID only format of the Connection ID for this service.
  - The switching function supports the DTMF tones "A, B, C, D"
  - The switching function supports the pause tone character of ","
  - Service Request Acknowledgement Model (Multi-Step [TRUE])

## 3.9.2 Generate Telephony Tones

The Generate Telephony Tones service causes a telephony tone such as a beep, busy, or ringback to be sent on behalf of a connection in a call. This service also supports optional parameters to control tone generation.

This service does not affect the state or progress of a call.

## Request Message

Table 93: Generate Telephony Tones Service Request Parameters

Parameter Name	Content	M/C/O	Comments
connectionToSendTone	ConnectionID	M	Connection of the device which is generating the telephony tones for the call
toneToSend	Enumerated	M	<p>Specifies the tone to send. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• beep (OpenScape 4000: station only)</li> <li>• billing</li> <li>• busy</li> <li>• carrier</li> <li>• confirmation (OpenScape 4000: station only)</li> <li>• dial</li> <li>• faxCNG</li> <li>• hold</li> <li>• howler</li> <li>• intrusion</li> <li>• modemCNG</li> <li>• park</li> <li>• record warning (indicates call may be being recorded)</li> <li>• reorder</li> <li>• ringback</li> <li>• silence</li> <li>• SIT VC</li> <li>• SIT-IG</li> <li>• SIT-RØ</li> <li>• SIT-NC</li> <li>• SwitchSpecified0 through SwitchSpecified100 – reserved for switch specified tones.</li> </ul>

## Positive Response

Since the Generate Telephony Tones service positive response does not contain any service specific parameter, no table is provided.

## Negative Response

Table 94: Generate Telephony Tones Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidConnectionID	Connection identifier or some component of the connection identifier is invalid.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	Device ID contains an unsupported device ID type.
	valueOutOfRange	Parameter (other than a CSTA object) has a value that is not in the enumeration or range specified for that parameter.
StateIncompatibility	generic	Server is unable to be any more specific about the cause of the error.
	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error
	internalResourceBusy	An internal resource is in use.
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) Limitations for station devices: this service supports analog stations and digital stations without a display only. The device has to be in an I/O session initiated by a Fast Data service with dataPathType = voice.
- 3) The support of the Generate Telephony Tones service in context with RCGs depends on the special ACD configuration (ACD Routing Tables) in OpenScape 4000.
- 4) The Generate Telephony Tones service may be rejected even though the Services Permitted parameter in previously events indicated that it was allowed. This may happen at RCGs, caused by special ACD handling (ACD Routing Tables) without providing events.
- 5) For RCGs the tone is applied to the RCGs partner, but for stations the tone is applied to the station itself.
- 6) For RCGs the connectionToSendTone has to include the CallID and the deviceID, for stations the deviceID is sufficient when only one call exists at that device.
- 7) The requested tone continues to be generated until it is replaced by a switching function generated tone, or until the call is cleared.
- 8) Miscellaneous Characteristics:
  - DeviceIDOnly - The switching function accepts or supports the DeviceID only format of the Connection ID for this service (see limitations above).
  - Service Request Acknowledgement Model (Atomic [FALSE]).

### 3.9.3 Send User Information

The Send User Information service is used to send user data information.

This service does not affect the state or progress of a call.

#### Request Message

**Table 95: Send User Information Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
existingCall	ConnectionID	M	Specifies the connection on whose behalf user data is to be sent.
userData	UserData	M	Specifies the user data to send.

#### Positive Response

Since the Send User Information service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 96: Send User Information Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidConnectionID	Connection identifier or some components of the connection identifier is invalid.  OpenScape 4000: ConnectionID doesn't contain both, a CallID and a DeviceID. CallID consists of more than 4 bytes.
	invalidDeviceID	DeviceID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	DeviceID contains an unsupported device ID type.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	valueOutOfRange	Parameter (other than a CSTA object) has a value that is not in the enumeration or range specified for that parameter.  OpenScape 4000:  OpenScape 4000 generates this error, when the length of the received UserData is higher than the supported.
StateIncompatibility	generic	Internal resource error
	invalidObjectState	Object is in an incorrect state for the service.
	noActiveCall	No connection for the given parameter exists.
SystemResourceAvailability	generic	Internal resource error
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.
	resourceLimitExceeded	The service is supported by the server, but is unavailable because it would exceed the internal usage limit of the resource.  <ul style="list-style-type: none"> <li>There are no Generic Data buffers to handle userData</li> </ul>
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The Send User Information service can be used in any state after the called device has been alerted, except failed.
- 3) To handle the user data OpenScape 4000 needs a sufficient number of configured Generic Data buffers.
- 4) OpenScape 4000 supports a maximum length of 32 bytes UserData per request.
- 5) If OpenScape 4000 is currently processing a Deflect Call service or a Call Forward No Reply for the specified call (ConnectionID), a negative response with cause invalidObjectState will be sent to the application.
- 6) A positive acknowledgement does not necessarily mean that the User Data has been successfully received by the devices in a different node. It only indicates that the User Data has been sent.

7) Miscellaneous Characteristics:

- DeviceIDOnly - The switching function does not accept or support the DeviceID only format of the Connection ID for this service.
- Service Request Acknowledgement Model (Atomic [FALSE])

## 3.10 Routing Registration Services

### Routing Registration Services Summary

Table 97: Support of Routing Registration Services

Routing Registration Services	Service Description Section
Route Register	<a href="#">Section 3.10.1</a>
Route Register Abort	<a href="#">Section 3.10.2</a>
Route Register Cancel	<a href="#">Section 3.10.3</a>

### Routing Registration Services Descriptions

The entire ECMA CSTA III standard covering Routing Registration Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.10.1 Route Register

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.

#### Request Message

Table 98: Route Register Service Request Parameters

Parameter Name	Content	M/C/O	Comments
routingDevice	DeviceID	O	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.

## Positive Response

Table 99: Route Register Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
routeRegisterReqID	RouteRegisterReqID	M	Specifies the routing registration request identifier for this registration.

## Negative Response

Table 100: Route Register Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidDeviceId	Device ID contains an unsupported device ID type or is not known to the switching function.  OpenScape 4000: Device is not of type "RCG". A dialling number other than *888 is used.
SubscribedResourceAvailability	objectRegistrationLimitExceeded	This service request would exceed the switching function's limit on the number of registrations for this device.  OpenScape 4000: There is already a registration for the specified routing device.
SystemResourceAvailability	generic	internal resource error
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  • No link to OpenScape 4000.

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) If the parameter routingDevice is not present then the request is for all OpenScape 4000 routing devices.
- 3) Dialling number "\*888" as requested routingDevice activates registration for all RCGs with one routeRegisterReqID.
- 4) Only one registration is allowed per routingDevice. A dialling number "\*888" or no deviceId in the service request will register all routing devices that have not already been single registered.

- 5) If application1 has registered for a single routing device A and application2 (on any application link) has afterwards registered for all devices, application2 receives route requests of device B and application1 receives route requests of device A. After application1 has unregistered for routing services, application2 receives route requests of device A also.
- 6) The registration for a routing device has to be unique for a OpenScape 4000 system. This means that all CSTA III applications running against a OpenScape 4000 system have to be synchronized regarding the OpenScape 4000 routing devices they serve (there is no synchronization concerning routing data over all OpenScape 4000 CSTA).
- 7) The device numbers of the routing device must be encoded using the appropriate high-order byte encoding as specified in Table 5-1 on page 5-5, part 2, or a negative response is returned.
- 8) In OpenScape 4000 an applications registration for routing services doesn't set the service into effect. A subsequent Set Routing Mode service request from the application is necessary to enable the service (see [Section 3.13.9, "Set Routeing Mode", on page 238](#)).
- 9) The registration is deleted, if the link to the computing function (host) fails or if OpenScape 4000 CSTA restarts. The registration remains if the link between OpenScape 4000 CSTA and OpenScape 4000 fails.
- 10) Miscellaneous Characteristics:
  - AllroutingDevices - The switching function supports the ability to register for all routing devices within the switching function.

### 3.10.2 Route Register Abort

This service is used by the switching function to asynchronously cancel an active routing registration. This service invalidates a current routing registration. There is no positive acknowledgement defined for this service.

#### Request Message

**Table 101: Route Register Abort Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
routeRegisterReqID	RouteRegisterReqID	M	Specifies the routing registration request identifier for the routingregistration that was aborted.

#### Positive Response

There is no positive acknowledgement defined for this service.

#### Negative Response

Negative acknowledgement is not supported by CAP Inside.

#### Usage Notes

- 1) The registration is aborted by the switching function if the link between OpenScape 4000 CSTA and OpenScape 4000 fails.

### 3.10.3 Route Register Cancel

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.

#### Request Message

**Table 102: Route Register Cancel Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
routeRegisterReqID	RouteRegisterReqID	M	Specifies the routing registration request identifier for which the routing registration is to be cancelled.

#### Positive Response

Since the Route Register Cancel service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 103: Route Register Cancel Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error
	invalidRouteRegistrationCrossReferenceIdentifier	The route registration request identifier is invalid.
SystemResourceAvailability	generic	internal resource error
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000.</li> </ul>

#### Usage Notes

- 1) The registration is deleted, if the link to the computing function (host) fails or if OpenScape 4000 CSTA restarts. The registration remains if the link between OpenScape 4000 CSTA and OpenScape 4000 fails.
- 2) If application1 has registered for a single routing device A and application2 (on any application link) has afterwards registered for all devices, application2 receives route requests of device B and application1 receives route requests of device A. After application1 has unregistered for routing services, application2 receives route requests of device A also.

## 3.11 Routing Services

### Routing Services Summary

**Table 104: Support of Routing Services**

Routing Services	Service Description Section
Re-Route	<a href="#">Section 3.11.1</a>
Route End	<a href="#">Section 3.11.2</a>
Route Reject	<a href="#">Section 3.11.3</a>
Route Request	<a href="#">Section 3.11.4</a>
Route Select	<a href="#">Section 3.11.5</a>

### Routing Services Descriptions

The entire ECMA CSTA III standard covering Routing Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.11.1 Re-Route

The Re-Route service requests an alternate destination from the one provided by a previous Route Select service based on previous information provided for the call.

#### Request Message

**Table 105: Re-Route Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
crossRefIdentifier	routingCrossRefID	M	Specifies the cross reference identifier associated with the routing dialogue.
routeRegisterReqID	RouteRegisterReqID	C M	Specifies the route register request identifier associated with the route registration for this routing dialogue.
callLinkageData	CallLinkageData	C M	Specifies the global call data and thread data associated with the call. This parameter is mandatory if the switching function supports the call linkage feature otherwise it shall not be provided.

#### Positive Response

There is no positive acknowledgment associated with this service.

### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided. A negative response from any application has to comply the universalFailure FROM CSTA-error-definition { iso( 1) identified-organization( 3) icd-ecma( 12) standard( 0) csta3( 285) error-definition( 120) }.

### Usage Notes

- 1) Since OpenScape 4000 requires registration for routing services, the routeRegisterReqID parameter is mandatory.
- 2) The routeRegisterReqID and crossRefIdentifier values used are the same as in a previous Route Request.
- 3) This service is only sent to the application that has set the routing mode "on" for the RCG that is controlling the call.
- 4) OpenScape 4000 restarts the routing timer when it sends this message. This gives the application additional time to issue another Route Select message. The routing timer should be configured with sufficient time to allow for a route select attempt.
- 5) The delay ringback timer must be configured with sufficient time to allow for multiple route select attempts.
- 6) The application should send a Route End request to terminate the routing dialog. However, if a negative acknowledge is sent, the routing dialog is not terminated until the routing timer expires, at which time a Route End request is sent to the application.

## 3.11.2 Route End

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 time-out 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.

### Request Message

**Table 106: Route End Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
crossRefIdentifier	routingCrossRefID	M	Specifies the cross reference identifier associated with the routing dialogue.
routeRegisterReqID	RouteRegisterReqID	C M	Specifies the route register identifier associated with the route registration for this routing dialogue.
errorValue	ErrorValue	O	<p>Specifies the reason for the route end request</p> <p>OpenScape 4000:</p> <p>This parameter is provided only in case of an error sent by the OpenScape 4000 to the application.</p> <ul style="list-style-type: none"> <li>• The routing timer or delay ringback timer expired.</li> <li>• The caller has abandoned the call.</li> <li>• The OpenScape 4000 encountered a resource problem indicated in the Route Select message and is aborting the routing dialog.</li> </ul> <p>This parameter is not provided in case:</p> <ul style="list-style-type: none"> <li>• the call was successfully routed.</li> <li>• the application does not want to route the call.</li> </ul>

### Positive Response

There is no positive acknowledgment associated with this service.

### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided. A negative response from any application has to comply the universalFailure FROM CSTA-error-definition { iso( 1) identified-organization( 3) icd-ecma( 12) standard( 0) csta3( 285) error-definition( 120) }.

The following negative acknowledgement error codes sent by the OpenScape 4000 CSTA are possible:

**Table 107: Route End Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidRouteRegistrationCrossReferenceIdentifier	The route registration request identifier is invalid.
	invalidroutingCrossReferenceIdentifier	The service request specified a routing cross reference identifier that is not in use.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  - No link to OpenScape 4000

#### Usage Notes

- 1) Since OpenScape 4000 requires registration for routing services, the routeRegisterReqID parameter is mandatory.
- 2) This service is only accepted if the call is in a routing dialog, that is crossRefIdentifier and routeRegisterReqID parameters are valid.
- 3) The routeRegisterReqID and crossRefIdentifier values used within a Route End request form OpenScape 4000 are the same as in a previous Route Request or Re-Route request.
- 4) The OpenScape 4000 sends this message either because
  - a Route Select request is successfully completed,
  - the application did not respond before the expiration of the routing timer or the delay ringback timer,
  - a Route Reject request from the computing domain was successful.
  - the caller has abandoned the call,
  - OpenScape 4000 encounters a resource problem indicated by a Route Select request and aborts the routing dialog.
- 5) This service can only be sent by the application that has set the routing mode "on" for the RCG that is controlling the call.
- 6) Miscellaneous Characteristics:
  - Switching function supports sending this service request to the computing function.
  - Switching function supports receiving this service request from the computing function.

### 3.11.3 Route Reject

The Route Reject service request is sent to the switching function during a routing 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 routing request was issued from) for alternate routing.

#### Request Message

**Table 108: Route Reject Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
crossRefIdentifier	routingCrossRefID	M	Specifies the cross reference identifier associated with the routing dialogue.
routeRegisterReqID	RouteRegisterReqID	C M	Specifies the route register identifier associated with the route registration for this routing dialogue.

#### Positive Response

There is no positive acknowledgment associated with this service.

#### Negative Response

**Table 109: Route Reject Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidRouteRegistrationCrossReferenceIdentifier	The route registration request identifier is invalid.
	invalidroutingCrossReferenceIdentifier	The service request specified a routing cross reference identifier that is not in use.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.  OpenScape 4000:  The call is not in a routing dialog.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  • No link to OpenScape 4000

#### Usage Notes

- 1) Since OpenScape 4000 requires registration for routing services, the routeRegisterReqID parameter is mandatory.

- 2) Successful completion of a Route Reject request is indicated by a Route End message from the OpenScape 4000.
- 3) This service is only accepted if the call is in a routing dialog, that is crossRefIdentifier and routeRegisterReqID parameters are valid.
- 4) This service can only be sent by the application that has set the routing mode "on" for the routing device that is controlling the call.

### 3.11.4 Route Request

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.

#### Request Message

**Table 110: Route Request Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
crossRefIdentifier	routingCrossRefID	M	Specifies the cross reference identifier associated with the routing dialogue.
routeRegisterReqID	RouteRegisterReqID	C M	Specifies the route register identifier associated with the route registration for this routing dialogue.
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.
routingDevice	SubjectDeviceID	O	Specifies the device that initiated the Route Request service.  OpenScape 4000:  The Device ID is the device number of the RCG.
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.  OpenScape 4000:  The Device ID is the device number of the RCG.  Call ID is the call identifier of the call.

Parameter Name	Content	M/C/O	Comments
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.
callLinkageData	CallLinkageData	C M	Specifies the global call data and thread data associated with the call. This parameter is mandatory if the switching function supports the call linkage feature otherwise it shall not be provided.

### Positive Response

There is no positive acknowledgment associated with this service.

### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided. A negative response from any application has to comply the universalFailure FROM CSTA-error-definition { iso( 1) identified-organization( 3) icd-ecma( 12) standard( 0) csta3( 285) error-definition( 120) }.

### Usage Notes

- 1) Since OpenScape 4000 requires registration for routing services, the routeRegisterReqID parameter is mandatory.
- 2) This service is sent only to the application that has set the routing mode "on" for the RCG that is controlling the call.
- 3) If a negative acknowledge is sent, the routing dialog is not terminated until the routing timer expires, at which time a Route End message is sent to the application.
- 4) For the Route Request to be sent, the first step in the ACD Routing Table (ART) must be the delay ringback step.

**5) The Route Request is not sent:**

- If the callingDevice is consulting the RCG.
- If the call is transferred to the RCG.
- If the call reaches a second or further ARTs within the RCG.
- If the call is routed or re-routed to a new RCG.

**6) There are two timers that are related to the routing services:**

- The configurable delay ringback timer assures that ringback will be sent to the trunk within a configurable period of time. This timer must be set to a value that gives the application enough time to respond to the Route Request service request (and possible Re-Route requests as well). Note that the caller will experience silence until delay ringback is sent. If this timer expires during a routing dialog, the OpenScape 4000 terminates the routing dialog, issues a Route End request and sends an alert indication to the caller, and continues with the next step in the configured ACD Routing Table (ART).
- When the OpenScape 4000 sends a Route Request service request, it starts a configurable timer called the routing timer. This timer should be set to a value that gives the application enough time to respond to the Route Request. If this timer expires before the application issues a successful routing instruction, the OpenScape 4000 terminates the routing dialog and issues a Route End request, and continues with the next step in the configured ACD Routing Table (ART). In this case, because the delay ringback timer has not expired, ringback is not automatically sent to the caller.

**7) Miscellaneous Characteristics:**

- The switching function doesn't use the Route Request for non-call related routing.

### 3.11.5 Route Select

The Route Select service is used by the computing function to provide the destination requested by a previous Route Request or Re-Route service.

#### Request Message

**Table 111: Route Select Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
crossRefIdentifier	routingCrossRefID	M	Specifies the cross reference identifier associated with the routing dialogue.
routeRegisterReqID	RouteRegisterReqID	C M	Specifies the route register identifier associated with the route registration for this routing dialogue.
routeSelected	DeviceID	M	Specifies the primary selected destination of the call for which a route was requested.

#### Positive Response

There is no positive acknowledgment associated with this service.

## Negative Response

Table 112: Route Select Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error.
	invalidRouteRegistrationCrossReferenceIdentifier	The route registration request identifier is invalid.
	invalidroutingCrossReferenceIdentifier	The service request specified a routing cross reference identifier that is not in use.
	invalidDeviceId	Device ID contains an unsupported device ID type or is not known to the switching function.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.  OpenScape 4000:  The call is not in a routing dialog.  A previous Route Select service request is in progress.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  - No link to OpenScape 4000

## Usage Notes

- 1) Since OpenScape 4000 requires registration for routing services, the routeRegisterReqID parameter is mandatory.
- 2) This service is only accepted if the call is in a routing dialog, that is crossRefIdentifier and routeRegisterReqID parameters are valid.
- 3) This service can only be sent by the application that has set the routing mode "on" for the RCG that is controlling the call.
- 4) Successful completion of a Route Select service request is indicated by a Route End message from OpenScape 4000.
- 5) If the OpenScape 4000 cannot route the call to the specified destination, it resets the routing timer, and sends a Re-Route Request. This allows the application to try alternate destinations in case the original destination was busy, out of service, or resources were unavailable.
- 6) Routing to an extension does not honor do-not-disturb (DND) or immediate types of forwarding configured for that extension since these are considered features for non-ACD calls coming to agents. If the agent does not want to receive ACD calls, the agent needs to change agent state to unavailable. Routing to an extension does honor non-immediate types, for example, forward-no-answer. Routing to destination in another switching domain honors all types of forwarding.

## 3.12 Physical Device Feature Services

### Physical Device Feature Services Summary

**Table 113: Support of Physical Device Feature Services**

Physical Device Feature Services	Service Description Section
Get Message Waiting Indicator	<a href="#">Section 3.12.1</a>
Set Display	<a href="#">Section 3.12.2</a>
Set Lamp Mode	<a href="#">Section 3.12.3</a>
Set Message Waiting Indicator	<a href="#">Section 3.12.4</a>
Button Press	<a href="#">Section 3.12.5</a>
Get Microphone Mute	<a href="#">Section 3.12.6</a>
Set Microphone Mute	<a href="#">Section 3.12.7</a>

### Physical Device Feature Services Descriptions

The entire ECMA CSTA III standard covering Physical Device Feature Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.12.1 Get Message Waiting Indicator

The Get Message Waiting Indicator service provides the message waiting feature status at a specified device. The message waiting feature is typically used to notify a user (typically via a dedicated lamp on a phone device) when messages are available.

#### Request Message

**Table 114: Get Message Waiting Indicator Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device's physical element.

#### Positive Response

**Table 115: Get Message Waiting Indicator Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
messageWaitingOn	Boolean	M	Specifies the value of the requested feature. The complete set of possible values is: <ul style="list-style-type: none"><li>FALSE - Message waiting off.</li><li>TRUE - Message waiting on.</li></ul>

## Negative Response

Table 116: Get Message Waiting Indicator Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	requestIncompatibleWithSubjectDevice	The service request is not compatible with the subject device.  OpenScape 4000: Device does not have a display. Device does not have the Mailbox authorization set in its COS.
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices" on page 45](#).
- 2) The lamp of a OpenScape 4000 digital phone is also used for other features like PhoneMail, CallBack, External PhoneMail, etc. Therefore, the message waiting lamp may be ON even if there is no "application message waiting" set. Thus the messageWaitingOn parameter in the Get Message Waiting Indicator Service response is set to FALSE.

## 3.12.2 Set Display

The Set Display service allows the computing function to set a display associated with a device.

~~The Get Display service can be used to determine the size of a specific display.~~  
The capabilities exchange services can be used to determine the number of displays associated with a device.

## Request Message

Table 117: Set Display Service Request Parameters

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device's physical element.
contentsOfDisplay	Characters (240)	M	Specifies the text to place on the display as a string of characters consisting of the text on each row of the display (including spaces) concatenated together. If a null string is sent, the display will be cleared.  OpenScape 4000:  This parameter should have a maximum length of 227 characters, further characters will be truncated.

## Positive Response

Since the Set Display service positive response does not contain any service specific parameters, no table is provided.

## Negative Response

Table 118: Set Display Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.  OpenScape 4000:  For this service is could also mean that the device ID contains an unsupported device ID type.
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

**Usage Notes**

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The application has to send the text to display in an unformatted string, since OpenScape 4000 do the formatting according to the size of the display (e.g. OPTISET supports two lines of 24 characters each). Line-exceeding characters are printed in the following line. After the last line of the display, the string is truncated.
- 3) When using this service to write to a display of a device of OpenScape 4000, the text will be displayed without an audible indication and only temporarily (i.e. not static but only for a limited time). If the application requires a different behavior it shall use the CSTA III Fast Data service.
- 4) If the application wants to reset the display (before the temporary timer expires) it shall send a contentsOfDisplay with zero length. To blank a display it shall send the number of blank characters necessary.
- 5) Miscellaneous Characteristics:
  - The Switching Functions does not support modifying the relative positions of the logical and physical displays (i.e. scrolling) via this service.
  - Service Request Acknowledgement Model (Atomic [FALSE])

### 3.12.3 Set Lamp Mode

The Set Lamp Mode service allows a computing function to control how a specified lamp is lit at a specified device.

**Request Message****Table 119: Set Lamp Mode Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device's physical element.

Parameter Name	Content	M/C/O	Comments
lamp	LampID	M	<p>Specifies the LampID of the lamp on the device.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>1..19 - Lamps on the phone itself</li> <li>20 - NOT USED</li> <li>21 .. 36 - Lamps on the first add-on key module</li> <li>37 .. 40 - NOT USED</li> <li>41 .. 56 - Lamps on the second add-on key module</li> <li>57 .. 60 - NOT USED</li> <li>61 .. 76 - Lamps on the third add-on key module</li> <li>77 .. 80 - NOT USED</li> <li>81 .. 96 - Lamps on the fourth add-on key module</li> <li>97 .. 127 - NOT USED</li> </ul> <p>For analog devices always Lamp Id 1 shall be used.</p>
lampMode	Value	M	<p>Specifies how the lamp associated with the specified device should be lit. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li><del>0 - Broken flutter. Superposition of wink and flutter.</del></li> <li>1 - Flutter. Fast on and off.</li> <li>2 - Off. Lamp is off.</li> <li>3 - Steady. Lamp is continuously lit.</li> <li>4 - Wink. Lamp is winking.</li> <li><del>5 - not used</del></li> <li><del>All other values (6-100) are switching function specific.</del></li> </ul>

LampId IE value	Key Module	Physical Key Number	Logical Key Number
1 .. 32	0 (PHONE ITSELF)	1 .. 32	1 .. 32
41 .. 56	1 (1st DSS)	1 .. 16	33 .. 48
61 .. 76	2 (2nd DSS)	1 .. 16	49 .. 64
81 .. 96	3 (3rd DSS)	1 .. 16	65 .. 80
101 .. 116	4 (4th DSS)	1 .. 16	81 .. 96

Table 118.1 Special mapping of LampID for CP400 and CP600 phones

The above table shows the relation between the passed LampId IE value and the equivalent Key Module, physical Key Number and Logical Key Number for CP400/CP600 devices.

For instance, Lamp ID #41 corresponds to physical key #01 in Key Module #1 and logical key #33

#### Positive Response

Since the Set Lamp Mode service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 120: Set Lamp Mode Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	invalidLampIdentifier	The lamp identifier is invalid.
	invalidLampMode	The lamp mode is invalid.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.  OpenScape 4000:  - For this service it could also mean that device ID contains an unsupported device ID type.
	requestIncompatibleWithSubjectDevice	The service request is not compatible with the subject device.
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

#### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) This service will be allowed only if the lamp ID indicates a key (led) number that was either configured as a name key or that was not assigned to any function by OpenScape 4000 configuration.
- 3) For analog devices the lamp ID must always be 1.

## 4) Miscellaneous Characteristics:

- Service Request Acknowledgement Model (Atomic [FALSE])

### 3.12.4 Set Message Waiting Indicator

The Set Message Waiting Indicator service allows a computing function to control the status of the message waiting feature at a specified device. The message waiting feature is typically used to notify a user (typically via a dedicated lamp on a phone device) when messages are available.

#### Request Message

**Table 121: Set Message Waiting Indicator Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device's physical element.
messageWaitingOn	Boolean	M	Specifies the setting of the message waiting feature. The complete set of possible values is: <ul style="list-style-type: none"> <li>• OFF - Message waiting off.</li> <li>• ON - Message waiting on.</li> </ul>

#### Positive Response

Since the Set message Waiting Indicator service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 122: Set Message Waiting Indicator Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	requestIncompatibleWithSubjectDevice	The service request is not compatible with the subject device.  OpenScape 4000:  Device does not have a display.  Device does not have the Mailbox authorization set in its COS.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The corresponding Message-Waiting-Event is sent only if the messageWaitingOn parameter value of a new Set Message Waiting Indicator service request for a specific device has changed.
- 3) The Set Message Waiting Indicator service will use the existing message waiting lamp to indicate if there is any message.
- 4) The lamp of a OpenScape 4000 digital phone is also used for other features like PhoneMail, CallBack, External PhoneMail, etc. Therefore, the message waiting lamp may be ON even if there is no "application message waiting" set.
- 5) Miscellaneous Characteristics:
  - Service Request Acknowledgement Model (Atomic [FALSE])

## 3.12.5 Button Press

The Button Press service allows a computing function to simulate the activation of a specified button at a specified device.

### Request Message

**Table 123: Button Press Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device's physical element.
button	ButtonID	M	Specifies the button on the device.

### Positive Response

Since the Button Press service positive response does not contain any service specific parameters, no table is provided.

## Negative Response

Table 124: Button Press Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.  Device belongs to the supported device type, button index is invalid.
	invalidCalledDeviceID	PressButton service request for an invalid device.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	requestIncompatibleWithObject	Device is valid, but does not belong to the supported device types (e.g. AnaTe).
SystemResourceErrors	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScope 4000 devices"](#).
- 2) The button to be pressed is to be defined by its index (number). For OpenScope 4000 switch it is already exactly defined for each type of phone which index (number) belongs to which button. The implementation in CSTA for single domain native interface will also follow this definition.
- 3) Restrictions:  
OpenScope 4000 will not support the following keys:
  - +, - (volume keys)
  - Left, Right, OK keys (menu navigation).

## 3.12.6 Get Microphone Mute

The Get Microphone Mute service provides the microphone mute feature status of a microphone associated with an auditory apparatus at a specified device. While a device's microphone is muted, no audio information is transmitted over the device microphone. This feature is used when it is desired to prevent the other party(s) in a call from hearing a conversation through the device.

## Request Message

Table 125: Get Microphone Mute Service Request Parameters

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device's physical element.

**Positive Response**

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

**Table 126: Get Microphone Mute Service Response Parameters**

Parameter Name	Content	M/C/O	Comments
microphoneMuteList	List of Structures	M	<p>Specifies information about the specified auditory apparatus or about all auditory apparatuses associated with the device if no auditoryApparatus was provided in the request. Each entry contains the following:</p> <ul style="list-style-type: none"> <li>auditoryApparatus (M) AuditoryApparatusID - Specifies the auditory apparatus that the microphone belongs to.</li> <li>microphoneMuteOn (M) Boolean - Specifies whether the microphone is muted or not. The possible values are: FALSE - Microphone is activated. TRUE - Microphone is muted.</li> </ul>

**Negative Response****Table 127: Get Microphone Mute Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	invalidMicrophoneMute	A microphone mute setting is invalid.
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### 3.12.7 Set Microphone Mute

The Set Microphone Mute service allows the computing function to control the microphone mute status of the microphone associated with one auditory apparatus at a specified device. While a device's microphone is muted, no audio information is transmitted over the microphone. This is used when it is desired to prevent the other device(s) in a call from hearing a conversation.

#### Request Message

**Table 128: Set Microphone Mute Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device's physical element.
microphoneMuteOn	Boolean	M	Specifies the microphone mute setting of a particular microphone. The complete set of possible values is: <ul style="list-style-type: none"> <li>OFF - Microphone is activated.</li> <li>ON - Microphone is muted.</li> </ul>

#### Positive Response

Since the Set Microphone Mute service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

**Table 129: Set Message Waiting Indicator Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	invalidMicrophoneMute	A microphone mute setting is invalid.
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

#### Usage Notes

This request generates a microphone mute event.

## 3.13 Logical Device Feature Services

### Logical Device Feature Services Summary

**Table 130: Support of Logical Device Feature Services**

Logical Device Feature Services	Service Description Section
Cancel Call Back	<a href="#">Section 3.13.1</a>
Get Agent State	<a href="#">Section 3.4.3</a>
Get Do Not Disturb	<a href="#">Section 3.13.3</a>
Get Forwarding	<a href="#">Section 3.13.4</a>
Get Routing Mode	<a href="#">Section 3.13.5</a>
Set Agent State	<a href="#">Section 3.13.6</a>
Set Do Not Disturb	<a href="#">Section 3.13.7</a>
Set Forwarding	<a href="#">Section 3.13.8</a>
Set Routing Mode	<a href="#">Section 3.13.9</a>

#### Logical Device Feature Services Descriptions

The entire ECMA CSTA III standard covering Logical Device Feature Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.13.1 Cancel Call Back

The Cancel Call Back service allows the computing function to cancel a previous (or all) Call Back feature at a device.

Note that this service cancels call backs that were created with either call related or non-call related Call Back features.

**Request Message****Table 131: Cancel Call Back Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
originatingDevice	DeviceID	M	The DeviceID of the device who initiated the original Call Back service.
targetDevice	DeviceID	M	The DeviceID of the target of the original Call Back service. If the switching function supports clearing of all Call Back features (as indicated by the capability exchange services) and a null format DeviceID (a DeviceID with 0 characters) is provided, all of the Call Back features at the originatingDevice are cancelled.

**Positive Response**

Since the Cancel Call Back service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 132: Cancel Call Back Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCalledDeviceIdentifier	The called device parameter is invalid.
	invalidCallingDeviceIdentifier	The calling device parameter is invalid.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request. <ul style="list-style-type: none"><li>Device ID contains an unsupported device ID type.</li></ul>
StateIncompatibility	generic	Internal resource error.  OpenScape 4000:  Cancel Callback Party's check subunit is not idle.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	invalidObjectState	Object is in an incorrect state for the service.  OpenScape 4000:  Cancel Callback Party is in an improper call state.  Cancel Callback Party's prime line is in use at another device.  Cancel Callback Party device is active on a non-primary line
SystemResourceAvailability	generic	Internal resource error.  OpenScape 4000:  The device has its check/program key active
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) A positive response is send to the application even there is no callback set at the originatingDevice.
- 3) The Call Back event is sent only if the service is successful (positive acknowledgment) and at least one callback was cancelled.
- 4) In order to cancel all callbacks at the originating device, the application needs to send a service request containing a targetDevice parameter with number of digits = 0.
- 5) Miscellaneous Characteristics:
  - Service Request Acknowledgement Model (Multi-Step [TRUE]).
  - The switching function supports clearing of all Call Back features at a device.

## 3.13.2 Get Agent State

The Get Agent State service provides the agent state at a specified device.

### Request Message

**Table 133: Get Agent State Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the DeviceID of the device on which the agent state is being queried.

## Positive Response

Table 134: Get Agent State Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
agentStateList	List of Structures	M	<p>This parameter specifies a list of agent identifiers, and/or their corresponding agent states and ACD groups for a given device. This list parameter has a maximum list size of 32 entries. Each entry contains the following components:</p> <ul style="list-style-type: none"> <li>agentID (C) AgentID - Indicates the agentID of the agent with respect to the associated ACD device or ACD group. This component shall be provided if there are multiple agentIDs associated with the agent device. OpenScape 4000: Only supplied when the loggedOnState is TRUE.</li> <li>loggedOnState (M) Boolean - Indicates the logged on state of the agent. The complete set of possible values is:  TRUE - Agent is logged on.  FALSE - Agent is not logged on.</li> <li>agentInfo (O) List of Structures - A specific agent may be associated with one or more agent states. The following components are associated with each agent state:  acdGroup (C) DeviceID - This component is mandatory in an entry when there is more than one entry in the list. OpenScape 4000: Only supplied when the loggedOnState is TRUE.  agentState (M) Enumerated- The complete set of possible values is Busy, Not Ready, Null, Ready, and Working After Call.  pendingAgentState (C) Enumerated - Indicates the pending agent state if the agentState is Busy or Working After Call. This component shall be provided if the switching function is delaying the transition to the pendingAgentState until the agent is no longer Busy or Working After Call, otherwise the parameter is optional. The possible complete set of possible values is: Working After Call, Not Ready, Ready, Null.  agentStateCondition (O) Enumerated - Indicates the agent state condition associated with the agent state. The complete set of possible values is: Forced Pause, Other.</li> </ul>

## Negative Response

Table 135: Get Agent State Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	requestIncompatibleWithSubjectDevice	The service request is not compatible with the subject device.
System resource availability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) The *typical* sequence of agent states is:
  - a) **Null** - The agent is not logged on to an ACD Group.
  - b) **Not ready** - The agent logs on to an ACD Group by using the telephone feature or CSTA service. The agent cannot receive ACD calls. If an agent device is configured for auto logon, the initial state is ready.
  - c) **Ready** - The agent goes into available state by using the telephone feature or CSTA service. The agent is now ready to receive ACD calls.
  - d) **Ready** - If an agent is connected with a call, the state remains ready.
  - e) **WorkingAfterCall** - WorkingAfterCall means that the agent is occupied and cannot receive ACD calls.
  - f) **Ready** - The agent indicates that *work* has been completed by telephone, with CSTA service, or through a OpenScape 4000 timer expiration. The agent can receive ACD calls.
  - g) **Not ready** - The agent indicates through a telephone feature or a CSTA service that a break period is in effect. The agent cannot receive ACD calls.

### 3.13.3 Get Do Not Disturb

The Get Do Not Disturb service provides the do not disturb feature status at a specified device. The do not disturb feature is used to prevent incoming calls at a device.

#### Request Message

**Table 136: Get Do Not Disturb Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the DeviceID of the device on which the do not disturb feature is being queried.

#### Positive Response

**Table 137: Get Do Not Disturb Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
doNotDisturbOn	Boolean	M	Specifies the value of the requested feature. The complete set of possible values is: <ul style="list-style-type: none"> <li>FALSE - Do not disturb feature is not enabled.</li> <li>TRUE - Do not disturb feature is enabled.</li> </ul>

#### Negative Response

**Table 138: Get Do Not Disturb Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	requestIncompatibleWithSubjectDevice	The service request is not compatible with the subject device.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
System resource availability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).

## 3.13.4 Get Forwarding

The Get Forwarding service provides the forwarding feature status at a specified device. The status returned may consist of one or more forwarding types that are active at the specified device based on user defined conditions.

The forwarding feature is used to redirect calls that arrive at a specified device to an alternate destination.

### Request Message

**Table 139: Get Forwarding Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device on which to query.

## Positive Response

Table 140: Get Forwarding Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
forwardList	List of Structure	M	<p>The list contains one structure per forwardingType/forwardDN combination. The structure has the following components:</p> <ul style="list-style-type: none"> <li>forwardingType (C) Enumerated. Specifies the type of forwarding. It shall be provided for user specified settings and it is optional for switching function default settings. The "internal" and "external" types refer to the type of call origination (for example an external call) that will be forwarded if it matches a forwarding type (for example forwardImmExt) enabled at the device. The complete set of possible values is:   forwardImmediate  forwardBusy  forwardDND  forwardNoAns  forwardBusyInt  forwardBusyExt  forwardDNDInt  forwardDNDExt  forwardNoAnsInt  forwardNoAnsExt  forwardImmInt  forwardImmExt</li> <li>forwardStatus (M) Boolean. Indicates the status of the forwarding type. The complete set of possible values is:   FALSE - the forwarding type is deactivated.   TRUE - the forwarding type is active.</li> <li>forwardDN (C) DeviceID. Specifies the destination to</li> </ul>

Parameter Name	Content	M/C/O	Comments
			<p>which calls are forwarded.</p> <p>It shall be provided for user specified settings and it is optional for switching function default settings</p>
forwardList (continued)			<ul style="list-style-type: none"> <li>forwardDefault (O) Enumerated. Specifies that the provided forwardingType and/or the forwardDN is a default setting. If the forwardDefault parameter is supported by the switching function and it is not present, the information is not a default setting. The complete set of possible values is:  defaultForwardingTypeAndForwardDN OpenScape 4000: Switching function default Fixed (US).  defaultForwardingType OpenScape 4000: Switching function default Fixed-IM.  defaultForwardDN</li> <li>ringCount (O) Value (1...100). It specifies the number of times that the device rings prior to forward no answer.</li> </ul>
privateData <sup>21</sup>	CSTAPrivateData	O	<p>Specifies non-standardized information.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>staticONDAActive</li> <li>staticONDDN</li> </ul> <p>(see <a href="#">Chapter 9, "Appendix C - Private Data"</a>)</p>

### Negative Response

**Table 141: Get Forwarding Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operations	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.

<sup>21</sup> Supported starting with OpenScape 4000 V8R1

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
System resource availability	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	requestIncompatibleWithSubjectDevice	The service request is not compatible with the subject device.
	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.
		<ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) There are two possible levels of forwarding settings that are supported by the switching function. (refer to the capability exchange services for the levels supported by an implementation):
  - switching function default settings - a single set of forwarding type/ forwarding destination combinations that can be activated/deactivated as a set. OpenScape 4000: Fixed (US) and Fixed-IM.
  - user specified settings - individual forwarding type/forwarding destination combinations that can be activated/deactivated one at a time.
- 3) The forwardDefault parameter in the forwardList indicates if information in the forwardList is the default information associated with the device (in the case where the forwarding settings have never been changed, for example) and what forwarding level the information is associated with. The possible values for forwardDefault are:
  - defaultForwardingTypeAndForwardDN - indicates that the information is associated with the switching function default settings. OpenScape 4000: Fixed (US).
  - defaultForwardingType - indicates that forwardingType is a default value associated with user specified settings. OpenScape 4000: Fixed-IM
  - forwardDefault not present indicates that the forwardDN is associated with user specified settings.
- 4) OpenScape 4000 supports three different kinds of forwarding:
  - User Specified (Variable) forwarding: Destination and the ForwardingType can be changed in the Set Forwarding Service Request.
  - Switching Function Default Forwarding (Fixed-IM): The configured Forwarding can only be activated or deactivated by the Service Request. The activated forwarding has the forwardingType forwardImmediate to the configured destination. The configured destination can not be changed by the service request.

- **Switching Function Default Forwarding (Fixed (US)):** The configured Forwarding can only be activated or deactivated by the Service Request. A maximum of eight destinations can be configured.

These three kinds of Call Forwarding can be activated simultaneously, but the Fixed (US) forwarding has lower priority than the other ones.

**Table 142: Forwarding Types**

User Specified Forwarding Types	Switching Function Default Forwarding Types (Fixed (US))	Switching Function Default Forwarding Types (Fixed-IM)
<ul style="list-style-type: none"> <li>• Forward immediate</li> <li>• Forward immediate internal</li> <li>• Forward immediate external</li> <li>• Forward no answer</li> <li>• Forward busy</li> </ul>	<ul style="list-style-type: none"> <li>• Forward immediate</li> <li>• Forward immediate internal</li> <li>• Forward immediate external</li> <li>• Forward no answer</li> <li>• Forward busy</li> <li>• Forward no answer internal</li> <li>• Forward no answer external</li> <li>• Forward busy internal</li> <li>• Forward busy external</li> <li>• Forward DND</li> <li>• Forward DND internal</li> <li>• Forward DND external</li> </ul>	<ul style="list-style-type: none"> <li>• Forward immediate</li> </ul>

5) The Get Forwarding service is possible in all device states except when the device is out-of-service.

6) If Forwarding is not active at the device, this service reports the default forwarding type *forward immediate* associated with a forwardStatus set to FALSE.

### 3.13.5 Get Routeing Mode

This **Get Routeing Mode** service indicates if a device is able to make routing requests to the computing function.

#### Request Message

**Table 143: Get Routing Mode Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the DeviceID of the device on which the routeing mode is being queried.

## Positive Response

Table 144: Get Routing Mode Service Response; Positive Result

Parameter Name	Content	M/C/O	Comments
routeingMode	Boolean	M	<p>Indicates the routeing mode of the device. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>TRUE - the device will request routeing instructions when a call arrives at the device.</li> <li>FALSE - the device will not request routeing instructions.</li> </ul>

## Negative Response

Table 145: Get Routing Mode Service Response Error Codes

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	<p>Default or internal error.</p> <p>OpenScape 4000:</p> <p>Device not registered or not registered for this port.</p>
	invalidDeviceID	<p>Device ID contains an unsupported device ID type or is not known to the switching function.</p> <p>OpenScape 4000:</p> <p>Dialing number is not "*888"</p> <p>Not a valid Route Control Group number (out of range)</p> <p>Device number does not indicate to be a Route Control Group.</p>
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

## Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) RouteingMode is set by the Set Routeing Mode service only.
- 3) If the Get Routeing Mode service is invoked for a device which hasn't been registered for routing services yet, a negative response will be returned by OpenScape 4000.

### 3.13.6 Set Agent State

The Set Agent State service requests a new agent state at a specified device. In the case where an ACD agent is involved with an ACD call, the transition to the requested state may or may not occur until the current connection transitions to the null state.

#### Request Message

**Table 146: Set Agent State Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the DeviceID for the ACD agent for which the state is to be changed. The device may also be an ACD device or an ACD group device if allowed by the switching function, as indicated by the capability exchange services.
requestedAgentState	Enumerated:	M	<p>Specifies the requested agent state. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>loggedOn - Requests that the agent be logged on.</li> <li>loggedOff - Requests that the agent be logged off.</li> <li>notReady - Requests that the agent be placed into the notReady agent state.</li> <li>ready - Requests that the agent be placed into the ready state.</li> <li>workingAfterCall - Requests that the agent be placed into the workingAfterCall state.</li> </ul> <p>Note that the list of values in this parameter is different from the list of values in the agentState parameter in the Get Agent State service.</p>
agentID	AgentID	C	<p>Specifies the agent identifier. This parameter must be provided if there are multiple agentIDs associated with the device.</p> <p>OpenScape 4000: AgentID is mandatory for the requestedAgentState loggedOn and optional for loggedOff. It is not supported for other states.</p>
group	DeviceID	C	<p>Specifies the agent ACD group that the agent is logging in or out of. The presence or absence of this parameter indicates the type of agent log on model.</p> <p>OpenScape 4000: This parameter is optional for the loggedOn and loggedOff states. It is not supported for other states.</p>

**Positive Response****Table 147: Set Agent State Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
pendingAgentState	Enumerated	C	<p>Indicates the agent state that the agent will transition to after the agent state is no longer Busy or Working After Call. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• Working After Call</li> <li>• Not Ready</li> <li>• Ready</li> <li>• Null</li> </ul> <p>This parameter shall be provided if the switching function is delaying the transition to the pendingAgentState until the agent is no longer Busy or Working After Call, otherwise the parameter is optional.</p>

**Negative Response****Table 148: Set Agent State Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>The ACD feature is not enabled in the OpenScape 4000.</li> <li>Subject device is not configured as an ACD agent.</li> <li>The subject device is a phantom line, and it is an appearance on more than one device.</li> <li>The agent ID is already in use.</li> <li>The agent ID is configured as the auto-logon agent ID of another station.</li> <li>The request is for logon, and the specified agent ID is not associated with the specified ACD group number.</li> <li>The request is for logoff, and the subject device is not logged on with the specified Agent ID.</li> <li>The request is for logoff, and the subject device is not logged on to the specified ACD group.</li> <li>In the request is for an ACD Group and the ACD group is a PhoneMail ACD group.</li> </ul>
	invalidDeviceID	<p>Device ID contains an unsupported device ID type or is not known to the switching function.</p> <ul style="list-style-type: none"> <li>ACD group is not configured or is in the wrong format.</li> </ul>
State incompatibility	invalidObjectState	<p>Object is in an incorrect state for the service.</p> <ul style="list-style-type: none"> <li>The agent is blocked for configuration changes.</li> <li>The agent's state is already the requested agent state.</li> <li>The logon was requested and the subject device has an active or held call.</li> </ul>
System resource availability	Generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#). Additionally this service is provided for ACD Groups.
- 2) If this service is issued while the agent is Busy, the transition to the requestedAgentState may be delayed until the agent is no longer Busy. If there is a monitor on the Agent Busy or the Agent Working After Call event, the switching function generates a Agent Busy event with the pendingAgentState parameter that reflects the requestedAgentState.
- 3) The agent must be logged on or the following feature values will not be accepted:
  - Log out
  - Not ready
  - Ready
  - WorkingAfterCall
- 4) The switching domain verifies whether the specified device can log on/off the specified group (as indicated by the group parameter). If this is not possible, a negative response is returned.
- 5) All agents are logged off after a hard restart or reload. Agents configured for autologon are automatically logged back on and changed to the *loggedOn*.
- 6) For agents connected to phantom lines, the phantom line should be configured for one device only. If the phantom line is configured on more than one device, the request for logging on causes a negative response.
- 7) In case of setting agent state of an ACD Group, the group parameter will be ignored.
- 8) The possible states that can be set for an ACD Group are:
  - Logged Off
  - Ready
  - Not-ready
- 9) A positive response is also returned if no agents are logged on to a specified ACD Group.

**10) Miscellaneous Characteristics:**

- Service Request Acknowledgement Model (Atomic [FALSE])
- The switching function allows a group (ACD group) device in the service request (i.e. service applies to all agents associated with the ACD group).
- The switching function does not allow an ACD device in the service request (i.e. service applies to all agents associated with the ACD device).
- The switching function delays transition to the requestedAgentState if it is Busy (i.e. support the pending agent state).
- The switching function does not delay transition to the requestedAgentState if it is WorkingAfterCall (i.e. support the pending agent state).

**3.13.7 Set Do Not Disturb**

The Set Do Not Disturb service allows the computing function to control the do not disturb feature at a specified device. The do not disturb feature is typically used to prevent a specified device from being alerted.

**Request Message****Table 149: Set Do Not Disturb Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device on which to set the feature.
doNotDisturbOn	Boolean	M	Specifies whether the do not disturb feature is enabled. The complete set of possible values is: <ul style="list-style-type: none"> <li>• FALSE- Do not disturb feature is not enabled.</li> <li>• TRUE- Do not disturb feature is enabled.</li> </ul>

**Positive Response**

Since the Set Do Not Disturb service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 150: Set Do Not Disturb Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>• Ringer Cutoff (RCOFF key) is not configured for the digital telephone.</li> <li>• The subject device is configured for originate only.</li> </ul>
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.
System resource availability	generic	Internal resource error.
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>• No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) This service invokes the do-not-disturb feature for the specified Device ID. In addition, no ringing shall occur at the device where the specified Device ID is configured as the primary appearance of the device.
- 3) For digital telephones, a user manually enables the DND feature by using the ringer cutoff (RCOFF) key. For manual operation, all devices with ringing line appearances of the subject device must also have ringer cutoff activated for DND to be enabled for the line.
- 4) For digital telephones, a user manually disables the DND feature by using the ringer cutoff (RCOFF) key. For manual operation, DND can be disabled by a user pressing the RCOFF key at either the subject device or at a device with a ringing line appearance of the subject device.
- 5) For digital telephones, if the ringer cutoff (RCOFF) key is not configured, a negative response is returned.
- 6) Miscellaneous Characteristics:
  - Service Request Acknowledgement Model (Atomic [FALSE])

### 3.13.8 Set Forwarding

The Set Forwarding service allows the computing function to control the forwarding feature at a specified device based on user defined conditions. The forwarding feature is used to redirect calls that arrive at a specified device to an alternate destination.

This service allows only one user-specified setting (forwarding type/forward-destination combination) to be changed per service invocation.

#### Request Message

**Table 151: Set Forwarding Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device on which to set the feature.
forwardingType	Enumerated	C	<p>Specifies the type of forwarding. The "internal" and "external" types refer to the type of call origination (for example an external call) that will be forwarded if it matches a forwarding type (for example, forwardImmExt) enabled at the device. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• forwardImmediate</li> <li>• forwardBusy</li> <li>• forwardDND</li> <li>• forwardNoAns</li> <li>• forwardBusyInt</li> <li>• forwardBusyExt</li> <li>• forwardDNDInt</li> <li>• forwardDNDExt</li> <li>• forwardNoAnsInt</li> <li>• forwardNoAnsExt</li> <li>• forwardImmInt</li> <li>• forwardImmExt</li> </ul> <p>This parameter shall be provided for user specified forwarding settings and shall not be provided for switching function default settings.</p>
activateForward	Boolean	M	<p>Indicates the status of the forwarding type. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• True: Activate forwarding</li> <li>• False: Deactivate forwarding</li> </ul>

Parameter Name	Content	M/C/O	Comments
forwardDN	DeviceID	C	<p>Specifies the device to which new calls are forwarded. This parameter shall be provided for variable forwarding settings when activateForward is TRUE or FALSE.</p> <p>It shall not be provided for switching function default settings.</p> <p>OpenScape 4000:</p> <p>In the case of Fixed -IM Forwarding this parameter contains a dialable String with zero length.</p> <p>In the case of Fixed (US) Forwarding this parameter is not present.</p>
privateData <sup>22</sup>	CSTAPrivateData	O	<p>Specifies non-standardized information.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>staticONDAActive</li> <li>staticONDDN</li> </ul> <p>(see <a href="#">Chapter 9, "Appendix C - Private Data"</a>)</p>

### Positive Response

Since the Set Forwarding service positive response does not contain any service specific parameters, no table is provided.

### Negative Response

**Table 152: Set Forwarding Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	Generic	Default or internal error.

<sup>22</sup> Supported starting with OpenScape 4000 V8R1

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	requestIncompatibleWithObject	<p>The service request is not compatible with the corresponding object specified in the service request.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>The call forwarding system feature is not configured.</li> <li>The subject device is originate only.</li> <li>The subject device equals forwarding directory number.</li> <li>The subject device has Station Hunt configured (for system call forwarding system only).</li> <li>ITR (Internal Traffic Restrictions) does not allow subject device to call the forwarding directory number.</li> <li>The forwardDN is not a supported device type.</li> <li>The forwardDN is not originate only.</li> </ul>
	invalidDeviceID	The requested device category is not identified by the switching domain.
	invalidDestination	The forwarding directory number is not configured as a device is not configured in the dialing plan.
System resource availability	Generic	Internal resource error.
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) To activate or deactivate user specified settings, the computing function shall specify the forwardingType and the forwardDN. To activate or deactivate switching function default settings, the computing function shall not specify the forwardingType. The computing function should use the capabilities exchange services to determine which level(s) of forwarding settings are supported by the switching function.

- 3) There are two possible levels of forwarding settings that are supported by the switching function. (refer to the capability exchange services for the levels supported by an implementation):
  - switching function default settings - a single set of forwarding type/forwarding destination combinations that can be activated/deactivated as a set. OpenScape 4000: Fixed (US) and Fixed-IM.
  - user specified settings - individual forwarding type/forwarding destination combinations that can be activated/deactivated one at a time.
- 4) The forwardDN indicates the three different kinds of forwarding in OpenScape 4000:
  - User Specified (Variable) Forwarding: Destination and the ForwardingType can be changed in the service request. The forwardDN is a mandatory parameter and contains a dialing string. The ForwardDN in the deactivate request must be present and contains the dialing string with either the calling DeviceID or the originating DeviceID.
  - Switching Function Default Forwarding (Fixed-IM) configured Forwarding can only be activated or deactivated by the service request. The activated forwarding has the forwardingType forwardImmediate to the configured destination. The configured destination can not be changed by the service request. The forwardDN is a mandatory parameter and must contain a dialing number with zero length.
  - Switching Function Default Forwarding (Fixed (US)): The configured Forwarding can only be activated or deactivated by the service request. A maximum of eight destinations can be configured. The forwardDN is not present in the service request.
- 5) These three kinds of Call Forwarding can be activated simultaneously, but the Fixed (US) forwarding has lower priority than the other ones. Non-delayed forwarding (immediate, busy, DND) also takes precedence over delayed forwarding (no answer).
- 6) Refer to [Table 142 "Forwarding Types"](#) for possible types of user specified forwarding types and switching function default forwarding types.
- 7) Miscellaneous Characteristics:
  - Service Request Acknowledgement Model (Atomic [FALSE]).

### 3.13.9 Set Routeing Mode

The **Set Routeing Mode** service allows the computing function to control the routing mode at a specified device.

The routing mode indicates if a device is able to make routing requests to the computing function.

#### Request Message

**Table 153: Set Routing Mode Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
device	DeviceID	M	Specifies the device on which to set the feature.

Parameter Name	Content	M/C/O	Comments
routingMode	Boolean	M	<p>Specifies the routing mode of the device. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>TRUE - the device will request routing instructions from the computing function when a call arrives at the device.</li> <li>FALSE - the device will not request routing instructions.</li> </ul>

### Positive Response

Since the Set Routeing Mode positive response does not contain any service specific parameters, no table is provided.

### Negative Response

**Table 154: Set Routing Mode Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	<p>Device ID contains an unsupported device ID type or is not known to the switching function.</p> <p>OpenScape 4000:</p> <p>Dialing number is not ""888"</p> <p>Not a valid Route Control Group number (out of range)</p> <p>Device number does not indicate to be a Route Control Group.</p> <p>Device is registered for another application - invalid port</p>
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	<p>Object is in an incorrect state for the service.</p> <p>OpenScape 4000:</p> <p>Service is already active for the requesting application.</p> <p>Service is already cleared for the requesting application.</p> <p>The Route Control Group a registration has been done for by an application is not in an appropriate state.</p>
SystemResourceAvailability	generic	Internal resource error.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 2) RoutingMode is set by the Set Routing Mode service only.
- 3) RoutingMode can be set only for devices which have been registered by the Route Register service.
- 4) In correlation with the Route Register service, the Set Routing Mode service for an appropriate device has to be unique for a OpenScape 4000 system. This means that all CSTA III applications running against a OpenScape 4000 system have to be synchronized regarding the OpenScape 4000 routing devices they serve (there is no synchronization concerning routing data over all OpenScape 4000 CSTA).
- 5) If application1 has activated routing for a single routing device A and application2 has afterwards activated routing for all devices, application2 receives route requests of device B and application1 receives route requests of device A. After application1 has deactivated for routing services, application2 receives route requests of device A also. A - GetRoutingMode Service request for device A therefor will respond with - RoutingMode=True.
- 6) RoutingMode can be set on individual devices or all applicable devices in OpenScape 4000 simultaneously (global), corresponding to a previous registration.
- 7) Global setting of RoutingMode (set RoutingMode for all applicable devices in OpenScape 4000 simultaneously) is done by selecting \*888 as dialingNumber or 0 as the deviceNumber for the device parameter in the Set Routing Mode Service request (according to the registration (Route Register) of the devices).
- 8) Individual and global actions may be used together with RCGs in the same OpenScape 4000. When this is the case, Set Routing Mode requests on individual RCGs take precedence over global trigger actions. For example, if an application sends a SetRoutingMode -TRUE request to an individual RCG A, and then it sends a global (\*888) SetRoutingMode TRUE, followed by a global SetRoutingMode FALSE request, only those RCGs that were triggered globally are turned off. RCG A remains triggered and needs to be turned off with an individual SetRoutingMode FALSE request.
- 9) The value of the routingMode parameter within the Set Routing Mode Service request has to toggle. E.g. if *routingMode = True* is sent to an individual or all applicable devices that already has been set to True by the same or a different application, then a negative response is returned.
- 10) RoutingMode is set to False automatically for all routing devices which have been triggered by an application if its (TCP-)Link to OpenScape 4000 breaks down.
- 11) If an application unregisters for routing (Route Register Cancel service), routingMode is set to False for all routing devices which correspond to the registrationID.

12) Miscellaneous Characteristics:

Service Request Acknowledgement Model (Atomic [FALSE])

## 3.14 I/O Registration Services

### I/O Registration Services Summary

Table 155: Support of I/O Registration Services

I/O Registration Services	Service Description Section
I/O Register	<a href="#">Section 3.14.1</a>
I/O Register Abort	<a href="#">Section 3.14.2</a>
I/O Register Cancel	<a href="#">Section 3.14.3</a>

### I/O Registration Services Descriptions

The entire ECMA CSTA III standard covering I/O Registration Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.14.1 I/O Register

The I/O Register service is used to register the computing function as an I/O server for a specific device or as an I/O server for all devices within the switching sub-domain. The computing function may be required to register for I/O services before it can receive I/O service requests for a device from the switching function. A computing function may register to be the I/O server for more than one I/O device.

Note that an I/O registration is not applicable when a data path is initiated by the computing function (i.e., a computing function can initiate a I/O data path after an I/O registration but the ioRegisterReqID parameter is not provided in the I/O services related to a data path that has been initiated by the computing function).

### Request Message

Table 156: I/O Register Service Request Parameters

Parameter Name	Content	M/C/O	Comments
ioDevice	DeviceID	O	Specifies the device for which the computing function requests to be the I/O server. This parameter is mandatory if the switching function does not support the option of registering for all devices in the switching sub-domain. Otherwise, the parameter is optional and if not present, indicates the registration is to be for all devices in the switching sub-domain.

Parameter Name	Content	M/C/O	Comments
privateData	CSTAPrivateData	O	Specifies non-standardized information.  OpenScape 4000: <ul style="list-style-type: none"> <li>applicationId</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

### Positive Response

**Table 157: I/O Register Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
ioRegisterReqID	IORegisterReqID	M	Specifies the I/O registration request identifier for this registration.

### Negative Response

**Table 158: I/O Register Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.  OpenScape 4000:  For this service invalidDeviceID might also mean an invalid subaddress.
SubscribedResourceAvailability	objectRegistrationLimitExceeded	This service request would exceed the switching function's limit on the number of registrations for this device.  OpenScape 4000:  Device is already registered for the given subaddress and applicationId.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>

### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).

- 2) The ioDevice (object) should be encoded using the switching function representation format of the device ID. This is an IA5 (ASCII) string of the syntax

Ndn!sa or N<dn!sa>nm

with dn being the directory number, sa the subaddress and nm the name string representing the person associated with the device (e.g. "N<4711!2>Mustermann").

OpenScape 4000 supports the following (one character) subaddresses:

- "0" for phone (default if not present)
  - "1" for smartCard
  - "2" for mobileUser
- 3) OpenScape 4000 supports only one registration for I/O services for each OpenScape 4000 device.
- 4) If one application registers for a OpenScape 4000 device then other applications won't be able to do so regardless of application ID or subaddress.
- 5) Omitting the application ID is treated like a registration for all application IDs in OpenScape 4000. Therefore this service request might get rejected if another application has successfully registered without an application ID for the same subaddress of a OpenScape 4000 device.
- 6) If the parameter ioDevice is not present then the request is for all devices. Since all CSTA III parameters of the I/O Register service request are optional the same applies if the service is used without any parameters. Therefore this service request might get rejected if another application has successfully registered without a device ID or any parameter at all.
- 7) If application1 has registered for the phone subaddress of device A and application2 (on any application link) has afterwards registered for all devices, application2 receives I/O data of device B and application1 receives I/O data of device A. After application1 has unregistered for I/O services, application2 receives I/O data of device A also.
- 8) The registration is deleted, if the link to the computing function (host) fails or if OpenScape 4000 CSTA restarts. The registration remains if the link between OpenScape 4000 CSTA and OpenScape 4000 fails.
- 9) Miscellaneous Characteristics:
- allIODEvices - The switching function supports the ability to register for all I/O devices within the switching function.

### 3.14.2 I/O Register Abort

This service is used by the switching function to asynchronously cancel an active I/O registration. This service invalidates a current I/O registration.

#### Request Message

**Table 159: I/O Register Abort Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
ioRegisterReqID	IORegisterReqID	M	Specifies the I/O registration request identifier for the I/O registration that was aborted.

**Positive Response**

There is no positive acknowledgement defined for this service.

**Negative Response**

Negative acknowledgement is not supported by Connectivity Adapter.

**Usage Notes**

- 1) The registration is aborted by the switching function if the link between OpenScape 4000 CSTA and OpenScape 4000 fails.

**3.14.3 I/O Register Cancel**

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

**Request Message****Table 160: I/O Register Cancel Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
ioRegisterReqID	IORegisterReqID	M	Specifies the I/O registration to be cancelled.

**Positive Response**

Since the I/O Register Cancel service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 161: I/O Register Cancel Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCrossRefID	The service request specified a Cross Reference Identifier ioRegisterReqID that is not in use.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	<p>The service is supported by the server, but is unavailable due to a resource that is out of service.</p> <ul style="list-style-type: none"> <li>• No link to OpenScape 4000</li> </ul>

#### Usage Notes

- 1) The registration is deleted, if the link to the computing function (host) fails or if OpenScape 4000 CSTA restarts. The registration remains if the link between OpenScape 4000 CSTA and OpenScape 4000 fails.
- 2) If application1 has registered for the phone subaddress of device A and application2 (on any application link) has afterwards registered for all devices, application2 receives I/O data of device B and application1 receives I/O data of device A. After application1 has unregistered for I/O services, application2 receives I/O data of device A also.

## 3.15 Input / Output Services

### Input / Output Services Summary

**Table 162: Support of Input / Output Services**

Input / Output Services	Service Description Section
Data Path Resumed	<a href="#">Section 3.15.1</a>
Data Path Suspended	<a href="#">Section 3.15.2</a>
Fast Data	<a href="#">Section 3.15.3</a>
Send Data	<a href="#">Section 3.15.4</a>
Start Data Path	<a href="#">Section 3.15.5</a>
Stop Data Path	<a href="#">Section 3.15.6</a>

### Input / Output Services Descriptions

The entire ECMA CSTA III standard covering Input / Output Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 3.15.1 Data Path Resumed

The Data Path Resumed service provides information that a previously suspended data path has been resumed.

## Request Message

Table 163: Data Path Resumed Service Request Parameters

Parameter Name	Content	M/C/O	Comments
ioCrossRefID	IOCrossRefID	M	Specifies the cross reference identifier associated with the data path and whether the computing function or switching function started the data path.  OpenScape 4000:  Only the computing function can start a data path, therefore the identifier is always switch-provided
privateData	CSTAPrivateData	O	Specifies non-standardized information.  OpenScape 4000: <ul style="list-style-type: none"> <li>mobileUserDirectoryNumber</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

## Positive Response

Since the Data Path Resumed service positive response does not contain any service specific parameters, no table is provided.

## Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided.

## Usage Notes

- 1) Although OpenScape 4000 CSTA supports this CSTA III service it's usage with OpenScape 4000 (i.e. the data path) does not work without private data. Therefore the Switching Function Capabilities will not indicate its support in Get Switching Function Capabilities service positive response.
- 2) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#) on page 45.

## 3.15.2 Data Path Suspended

The Data Path Suspended service provides information that a data path has been suspended.

**Request Message****Table 164: Data Path Suspended Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
ioCrossRefID	IOCrossRefID	M	<p>Specifies the cross reference identifier associated with the data path and whether the computing function or switching function started the data path.</p> <p>OpenScape 4000:</p> <p>Only the computing function can start a data path, therefore the identifier is always switch-provided</p>

**Positive Response**

Since the Data Path Suspended service positive response does not contain any service specific parameters, no table is provided.

**Negative Response**

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided.

**Usage Notes**

- 1) Although OpenScape 4000 CSTA supports this CSTA III service it's usage with OpenScape 4000 (i.e. the data path) does not work without private data. Therefore the Switching Function Capabilities will not indicate its support in Get Switching Function Capabilities service positive response.
- 2) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices" on page 45](#). The suspend mechanism is started whenever the user answers a call on the physical device.

**3.15.3 Fast Data**

The Fast Data service transfers data to/from a specified CSTA object. The services results in a data path to be created for only the duration of sending information contained in the Fast Data service request.

This is a bi-directional service.

## Request Message

Table 165: Fast Data Service Request Parameters

Parameter Name	Content	M/C/O	Comments
ioRegisterReqID	IORegisterReqID	C	<p>Specifies the I/O register request identifier associated with the registration for I/O services.</p> <p>This parameter is mandatory if the switching function supports I/O registration and the I/O service was requested from the switching function, and shall not be provided otherwise.</p>
object	Choice Structure	M	<p>Specifies the object with which a data path should be initiated. This shall be one of the following choices:</p> <ul style="list-style-type: none"> <li>Device (DeviceID) - specifies the device upon which the data path is to be started.</li> <li>Call (ConnectionID) - specifies the call (connection) upon which the data path is to be started.</li> </ul>
dataPathType	Enumerated	O	<p>Specifies the data-type of the data path. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>text - a digitally encoded text stream</li> <li>voice - a digitally encoded voice stream</li> </ul> <p>OpenScape 4000:</p> <p>This parameter is only supported for S -&gt; C. It is ignored if present for C -&gt; S.</p>
ioData	Characters (240)	M	Specifies the data to be sent.
privateData	CSTAPrivateData	O	<p>Specifies non-standardized information.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>applicationID (S -&gt; C)</li> <li>audibleIndication (C -&gt; S)</li> </ul> <p>(see <a href="#">Chapter 9, "Appendix C - Private Data"</a>)</p>

## Positive Response

Since the Fast Data service positive response does not contain any service specific parameters, no table is provided.

## Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided.

The following negative acknowledgement error codes sent by the OpenScape 4000 CSTA are possible:

**Table 166: Fast Data Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.  - For this service it might also mean an invalid subaddress.
	invalidObjectType	Object contains an unsupported object type.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.  <ul style="list-style-type: none"> <li>For this service it might also mean that the Device ID contains an unsupported device ID type.</li> </ul>
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	deviceOutOfService	A device that is needed to carry out the service is out of service.
PrivateDataInformation	cSTAPrivateDataInfoError	An error occurred in the privateData parameter. The reason for this error is implementation specific.  <ul style="list-style-type: none"> <li>The reason for this error could be that the private data was of the wrong service category.</li> </ul>

#### Usage Notes

- 1) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).

- 2) The device (object) should be encoded using the switching function representation format of the device ID. This is an IA5 (ASCII) string of the syntax

`Ndn!sa or N<dn!sa>nm`

with dn being the directory number, sa the subaddress and nm the name string representing the person associated with the device (e.g. "N<4711!2>Mustermann").

OpenScape 4000 supports the following (one character) subaddresses:

- "0" for phone (default if not present)
  - "1" for smartCard
  - "2" for mobileUser
- 3) The application can send I/O data to the display of a phone using this service (C -> S), but it should code the I/O data according to the Generic Display Protocol (see ACL-C Interface Specification of OpenScape 4000).
- 4) The voice I/O session is rejected for keysystem stations. The display of the device will show 'not possible' instead.
- 5) If an application registered for I/O services and therefore receives a Fast Data message but cannot interpret the ioData or privateData information, it shall not return a negative response, since this information is switching function specific anyway.
- Private Data: To avoid receiving this information in the first place it should register for I/O services including the OpenScape 4000 Application ID (see [Section 3.14.1, "I/O Register"](#)).
- 6) Private Data: If an application receives ioData of zero length OpenScape 4000 wants the application to start a data path for that device (see [Section 3.15.5, "Start Data Path"](#)).
- 7) Private Data: The audibleIndication is optional. If not present the OpenScape 4000 sets it to "silent".
- 8) Miscellaneous Characteristics:
- The switching function does not support modifying the relative positions of the logical and physical displays (i.e., scrolling) via this service.

### 3.15.4 Send Data

The Send Data service sends data to a specified data path.

This is a bi-directional service.

#### Request Message

**Table 167: Send Data Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
ioCrossRefID	IOCrossRefID	M	Specifies the cross reference identifier associated with the data path and whether the computing function or switching function started the data path.

Parameter Name	Content	M/C/O	Comments
ioData	Characters (240)	M	<p>Specifies the data to be sent.</p> <p>When writing to a display on a device, this specifies the data to place on the display as a string of characters consisting of the text on each row of the display (including spaces) concatenated together. If a null string is sent, the display will be cleared.</p>
ioCause	EventCause	O	<p>Specifies the reason why data is being sent. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>terminationCharacterReceived - the specified termination character was received.</li> <li>characterCountReached - the specified number of characters was reached.</li> <li>timeout - a timeout occurred.</li> <li>switchingFunctionTerminated - the switching function terminated collection before other termination conditions were encountered.</li> </ul> <p>Note that even though ioCause parameter is used in a service, the EventCause parameter type is used to represent the reason why the data is being sent.</p> <p>OpenScape 4000:</p> <p>This parameter is only supported for S -&gt; C</p>
privateData	CSTAPrivateData	O	<p>Specifies non-standardized information.</p> <p>OpenScape 4000:</p> <ul style="list-style-type: none"> <li>displayMode (C -&gt; S)</li> <li>audibleIndication (C -&gt; S)</li> <li>mobileUserDirectoryNumber (S -&gt; C)</li> </ul> <p>(see <a href="#">Chapter 9, "Appendix C - Private Data"</a>)</p>

### Positive Response

Since the Send Data service positive response does not contain any service specific parameters, no table is provided.

### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided.

The following negative acknowledgement error codes sent by the OpenScape 4000 CSTA are possible:

**Table 168: Send Data Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCrossRefID	The service request specified a Cross Reference Identifier that is not in use.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	InvalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  - No link to OpenScape 4000
PrivateDataInformation	cSTAPrivateDataInfoError	An error occurred in the privateData parameter. The reason for this error is implementation specific.  - The reason for this error could be that the private data was of the wrong service category.

**Usage Notes**

- 1) Although OpenScape 4000 CSTA supports this CSTA III service it's usage with OpenScape 4000 (i.e. the data path) does not work without private data. Therefore the Switching Function Capabilities will not indicate its support in Get Switching Function Capabilities service positive response.
- 2) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 3) The privateData fields displayMode and audibleIndication are optional. If not present the OpenScape 4000 CSTA sets them to "temporaryMode" and "silentIndication".
- 4) The application can send I/O data to the display of a phone using this service (C -> S), but it should code the I/O data according to the Generic Display Protocol (see ACL-C Interface Specification of OpenScape 4000).
- 5) Miscellaneous Characteristics:
  - The switching function does not support modifying the relative positions of the logical and physical displays (i.e., scrolling) via this service.

### 3.15.5 Start Data Path

The Start Data Path service starts a data path on the specified object.

~~This is a bi-directional service.~~ OpenScape 4000: This service is only supported for C -> S.

#### Request Message

**Table 169: Start Data Path Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
object	Choice Structure	M	Specifies the object to which a data path should be initiated. The complete set of possible objects is: <ul style="list-style-type: none"> <li>Device (DeviceID) - specifies the device upon which the data path is to be started.</li> <li><del>Call (ConnectionID) - specifies the call (connection) upon which the data path is to be started.</del></li> </ul>
numberOfCharsToCollect	Value	O	Specifies the number of characters to collect before sending collected characters on the data path.
terminationCharacter	Character(1)	O	Specifies an ASCII (IA5) character that, if received, causes the switching function to send collected characters on the data path.
privateData	CSTAPrivateData	O M	Specifies non-standardized information. OpenScape 4000: <ul style="list-style-type: none"> <li>applicationID (M)</li> <li>localIDMode</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

#### Positive Response

**Table 170: Start Data Path Service Response; Positive Result**

Parameter Name	Content	M/C/O	Comments
ioCrossRefID	IOCrossRefID	M	Specifies the cross reference identifier associated with the data path that has been created.  OpenScape 4000:  Only the computing function can start a data path, therefore the identifier is always switch-provided
numberOfCharsToCollect	Value	O	Specifies the number of characters to collect before sending collected characters on the data path.

Parameter Name	Content	M/C/O	Comments
terminationCharacter	Character(1)	O	Specifies an ASCII (IA5) character that, if received, causes the switching function to send collected characters on the data path.
privateData	CSTAPrivateData	O	Specifies non-standardized information.  OpenScape 4000: <ul style="list-style-type: none"> <li>mobileUserDirectoryNumber</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

### Negative Response

**Table 171: Start Data Path Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidDeviceID	Device ID contains an unsupported device ID type or is not known to the switching function.  For this service it might also mean an invalid subaddress.
	invalidObjectType	Object contains an unsupported object type.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.  For this service it might also mean that the Device ID contains an unsupported device ID type.
StateIncompatibility	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	deviceOutOfService	A device that is needed to carry out the service is out of service.

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
PrivateDataInformation	cSTAPrivateDataInfoError	<p>An error occurred in the privateData parameter. The reason for this error is implementation specific.</p> <ul style="list-style-type: none"> <li>• The reason for this error could be that the private data was of the wrong service category.</li> <li>• For this service it might also mean that the applicationID is missing.</li> </ul>

### Usage Notes

- 1) Although OpenScape 4000 CSTA supports this CSTA III service its usage with OpenScape 4000 (i.e. the data path) does not work without private data. Therefore the Switching Function Capabilities will not indicate its support in Get Switching Function Capabilities service positive response.
- 2) The following procedure applies to start a data path with a OpenScape 4000 device:
  - a) The application [Section 3.14.1, "I/O Register"](#) should register for I/O services (see chapter [Section 3.14.1, "I/O Register"](#)), including the ApplicationID (private data) as configured in OpenScape 4000.
  - b) If requested by the user from a OpenScape 4000 device the application will receive a Fast Data request (see [Section 3.15.3, "Fast Data"](#)). This message will include the initiating device, an ioData parameter of length zero, and the ApplicationID (private data) the application used in the I/O Register. The ioData of zero length indicates that OpenScape 4000 wants the application to start a data path for that device.
  - c) Now the application should send a Start Data Path request, including the device and the ApplicationID (private data) of the received Fast Data request. Note that the ApplicationID (private data) is mandatory, contrary to the ECMA-269 standard.
- 3) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices"](#).
- 4) The standard parameter ioRegisterReqID (see ECMA-269) is conditional, i.e. mandatory if the switching function supports I/O registration and the I/O Data Path was requested from the switching function, and shall not be provided otherwise. OpenScape 4000 CSTA only supports Start Data Path C -> S, therefore ioRegisterReqID shall not be provided for this service and all other data path messages. Instead the ioCrossRefID is used to identify the data path.

- 5) The device (object) should be encoded using the switching function representation format of the device ID. This is an IA5 (ASCII) string of the syntax  
`Ndn!sa or N<dn!sa>nm`
- with dn being the directory number, sa the subaddress and nm the name string representing the person associated with the device (e.g. "N<4711!2>Mustermann").
- OpenScape 4000 supports the following (one character) subaddresses:
- "0" for phone (default if not present)
  - "1" for smartCard
  - "2" for mobileUser
- 6) Although the standard parameter `dataPathType` (see ECMA-269) is not supported for this service OpenScape 4000 expects that the data path will be of the data-type as specified in the previously received Fast Data request.

### 3.15.6 Stop Data Path

The Stop Data Path service terminates an existing data path.

This is a bi-directional service.

#### Request Message

**Table 172: Stop Data Path Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
<code>ioCrossRefID</code>	<code>IOCrossRefID</code>	M	<p>Specifies the cross reference identifier associated with the data path to be terminated.</p> <p>OpenScape 4000:</p> <p>Only the computing function can start a data path, therefore the identifier is always switch-provided</p>

#### Positive Response

Since the Stop data Path service positive response does not contain any service specific parameters, no table is provided.

#### Negative Response

The negative acknowledgement error codes sent by the computing domain are application specific. Therefore no table is provided.

The following negative acknowledgement error codes sent by the OpenScape 4000 CSTA are possible:

**Table 173: Stop Data Path Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCrossRefID	The service request specified the Cross Reference Identifier that is not in use.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
StateIncompatibility	InvalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	generic	Internal resource error.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	deviceOutOfService	A device that is needed to carry out the service is out of service.

#### Usage Notes

- 1) Although OpenScape 4000 CSTA supports this CSTA III service it's usage with OpenScape 4000 (i.e. the data path) does not work without private data. Therefore the Switching Function Capabilities will not indicate its support in Get Switching Function Capabilities service positive response.
- 2) This service is supported for devices listed in [Table 2 "Services supported by OpenScape 4000 devices" on page 45](#).
- 3) It is not necessary for a suspended data path to be resumed for this service to be carried out.
- 4) If during a data path session the link to OpenScape 4000 fails, OpenScape 4000 CSTA will send a Stop Data Path request to the application.

## 3.16 Vendor Specific Extensions Services

### Vendor Specific Extensions Services Summary

**Table 174: Support of Vendor Specific Extensions Services**

Vendor Specific Extensions Services	Service Description Section
Escape	<a href="#">Section 3.16.1</a>
Private Data Version Selection	<a href="#">Section 3.16.2</a>

### Vendor Specific Extensions Services Descriptions

The entire ECMA CSTA III standard covering Vendor Specific Extensions Services is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

## 3.16.1 Escape Services

The Escape service is used by an implementation to send a non-standardized (implementation specific) feature using the CSTA protocol. This service shall not be used for features that can be invoked with standardized services.

The Escape service allows an implementation to "escape" from standard operations in order to exploit some special feature of an implementation. This mechanism also allows manufacturers to experiment with new features that may, at a later date, become standardized.

### Escape Services Summary

**Table 175: Support of Escape Services**

Escape Services	Service Description Section	Call Scenario Section
Set lower Class of Service	<a href="#">Section 3.16.1.1</a>	
Get lower Class of Service	<a href="#">Section 3.16.1.2</a>	

### Usage Notes

Miscellaneous Characteristics:

- 1) The switching function does not generate this service request.
  - The switching function supports receiving this service request.

### 3.16.1.1 Set lower Class of Service

This service allows an application to switch ON/OFF the lower class of service (COS2) for a specific device.

### Request Message

**Table 176: Escape Service Request Parameters: Set lower Class of Service**

Parameter Name	Content	M/C/O	Comments
privateData	CSTAPrivateData	M	Specifies non-standardized information.  OpenScape 4000: <ul style="list-style-type: none"> <li>• lowerCosOn</li> <li>• extensionNumber</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

**Positive Response**

Since the Escape service positive response (Set lower Class of Service) does not contain any service specific parameters, no table is provided.

**Negative Response****Table 177: Escape Service Response Error Codes: Set lower Class of Service**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallingDeviceIdentifier	The calling device parameter is invalid.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
	requestIncompatibleWithSubjectDevice	The service request is not compatible with the subject device.
StateIncompatibility	generic	Internal resource error.
	invalidObjectState	Object is in an incorrect state for the service.
SystemResourceAvailability	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.  - No link to OpenScape 4000
	generic	Internal resource error.
	resourceBusy	The service is supported by the server, but is unavailable due to a resource that is busy.

**Usage Notes**

1) This service is supported for the following OpenScape 4000 devices:

- Station Analog
- Station Digital
- CMI
- Phantom

**3.16.1.2 Get lower Class of Service**

This service allows an application to get the status of "lower class of service" (COS2) for a specific device.

**Request Message****Table 178: Escape Service Request Parameters: Get lower Class of Service**

Parameter Name	Content	M/C/O	Comments
privateData	CSTAPrivateData	M	Specifies non-standardized information.  OpenScape 4000: <ul style="list-style-type: none"> <li>extensionNumber</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

**Positive Response****Table 179: Escape Service Response; Positive Result: Get lower Class of Service**

Parameter Name	Content	M/C/O	Comments
privateData	CSTAPrivateData	M	Specifies non-standardized information.  OpenScape 4000: <ul style="list-style-type: none"> <li>lowerCosOn</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

**Negative Response****Table 180: Escape Service Response Error Codes: Get lower Class of Service**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error.
	invalidCallingDeviceIdentifier	The calling device parameter is invalid.
	requestIncompatibleWithObject	The service request is not compatible with the corresponding object specified in the service request.
SystemResourceAvailability	deviceOutOfService	A device that is needed to carry out the service is out of service.
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service. <ul style="list-style-type: none"> <li>No link to OpenScape 4000</li> </ul>
	generic	Internal resource error.

**Usage Notes**

1) This service is supported for the following OpenScape 4000 devices:

- Station Analog
- Station Digital
- CMI
- Phantom

**3.16.2 Private Data Version Selection**

The Private Data Version Selection service provides the switching function with the selected version for Private Data.

**Request Message****Table 181: Private Data Version Selection Service Request Parameters**

Parameter Name	Content	M/C/O	Comments
privateDataVersion	PrivateDataVersion	M	Represents the version number to be used for future Private Data. A value of 0 means Private Data is not to be used.

**Positive Response**

Since the Private Data Version Selection service positive response does not contain any service specific parameters, no table is provided.

**Negative Response****Table 182: Private Data Version Selection Service Response Error Codes**

CSTA Error Codes		Possible Causes
CSTA Error Category	CSTA Error Value	
Operation	generic	Default or internal error  OpenScape 4000: Private Data Version value is out of range.
SystemResourceAvailability	generic	Internal resource error
	resourceOutOfService	The service is supported by the server, but is unavailable due to a resource that is out of service.

**Usage Notes**

- 1) A list of possible values for the PrivateDataVersion parameter can be obtained from the response of the Get Switching Function Capability service.
- 2) A value of "0" for the PrivateDataVersion parameter is not supported. Therefore an application has to ignore unknown Private Data elements contained in requests and responses from OpenScape 4000.

- 3) In OpenScape 4000 CSTA the private data version is set to 1 by default.

## 4 Events

### Events Summary

This chapter describes the CSTA III events provided by OpenScape 4000 CSTA in cooperation with OpenScape 4000 and includes:

- A description of each CSTA events
- A list of supported devices

### Events Descriptions

The following chapters provide the following types of information about each event:

- An explanation of the events based on ECMA 269
- A table listing the supported parameters of the request message.
  - The original parameter name and content
  - An indication of whether the parameter is required or optional
  - A description of each parameter based on ECMA 269, which gives detailed OpenScape 4000 specific information.
- Usage notes providing important additional information about the events.
  - OpenScape 4000 specific behaviour
  - Chosen options of ECMA 269 and additions to functional requirements
  - Miscellaneous characteristics

Events	Description	Station Analog	Station Digital	CMI	Phantom	Funct- ional (ISDN)	Attendant Console	General Attendant	RCG	Hunt Group	Trunk
Call Control Event											
Conferenced	Indicates that the conferencing device has conferenced itself or another device with an existing call.	X	X	X	X		X				X
Connection Cleared	Indicates that a device in a call has disconnected or dropped out from a call.	X	X	X	X		X	X	X	X	X
Delivered	Indicates that a call is being presented to a device in either the Ringing or Entering Distribution modes of the alerting state.	X	X	X	X		X	X	X	X	X
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.	X	X	X	X						
Diverted	Indicates that a call has been diverted from a device.	X	X	X	X		X	X	X	X	X

Events	Description	Station Analog	Station Digital	CMI	Phantom	Functional (ISDN)	Attendant Console	General Attendant	RCG	Hunt Group	Trunk
Established	Indicates that a device has answered or has been connected to a call.	X	X	X	X		X		X		X
Failed	Indicates that a call cannot be completed and/or a connection has entered the Fail state.	X	X	X	X		X				X
Held	Indicates that an existing call has been put on hold.	X	X	X	X		X				X
Network Reached	Indicates that a call has been connected to an external network using a Network Interface Device (e.g., trunk, CO Line).	X	X	X	X		X	X	X	X	X
Offered	Indicates that the connection is in the Offered mode of the Alerting state.		X								
Originated	Indicates that a call is being attempted from a device.	X	X	X	X		X				X
Queued	Indicates that a call has been queued.	X	X	X	X		X	X	X	X	X
Retrieved	Indicates that a previously held call has been retrieved.	X	X	X	X		X				X
Service Initiated	Indicates that a device has gone off-hook for service or is being prompted to go off-hook.	X	X	X	X		X		X		X
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.	X	X	X	X		X	X	X	X	X
Call Associated Event											
Call Information	Indicates that call associated information (such as account code, services permitted, call linkage data, etc.) has been collected for a call.	X	X	X	X		X	X	X	X	X
Digits Generated	Indicates that DTMF digits have been generated.	X	X								
Telephony Tones Generated	Indicates that telephony tones have been generated.	X	X						X		
Physical Device Feature Events											

## Events

Events	Description	Station Analog	Station Digital	CMI	Phantom	Functional (ISDN)	Attendant Console	General Attendant	RCG	Hunt Group	Trunk
Message Waiting	The message waiting status has changed.		X								
Logical Device Event Supported											
Agent Busy	An agent is occupied with serving an ACD call.	X	X	X	X						X
Agent Logged Off	An agent has logged off of an ACD group or an ACD device.	X	X	X	X						X
Agent Logged On	An agent has logged on to an ACD group or an ACD device.	X	X	X	X						X
Agent Not Ready	An agent is unavailable and cannot receive incoming ACD calls.	X	X	X	X						X
Agent Ready	An agent is available for an ACD call.	X	X	X	X						X
Agent Working After Call	An agent is involved with after call work and cannot receive ACD calls.	X	X	X	X						X
Call Back	The call back feature status has changed.		X								
Do Not Disturb	The do not disturb status has changed.	X	X	X			X				
Forwarding	The forwarding status has changed.	X	X	X				X		X	
Routeing Mode	The routeing mode status has changed.								X		
Maintenance Events											
Back In Service	Indicates that the device has been returned to service.	X	X	X			X				X
Out Of Service	Indicates that the device has entered a maintenance state (i.e., has been taken out of service).	X	X	X			X				X
Vendor Specific Extensions Events											
Mobile User Status ( Private Event )	The switching function sends this event in case of a mobile user identification	X	X	X							

## 4.1 Call Control Events

### Call Control Events Summary

**Table 183: Support of Call Control Events**

Call Control Event	Event Description Section
Conferenced	<a href="#">Section 4.1</a>
Bridged	<a href="#">Section 4.1.1</a>
Connection Cleared	<a href="#">Section 4.1.2</a>
Delivered	<a href="#">Section 4.1.3</a>
Digits Dialed	<a href="#">Section 4.1.4</a>
Diverted	<a href="#">Section 4.1.5</a>
Established	<a href="#">Section 4.1.6</a>
Failed	<a href="#">Section 4.1.7</a>
Held	<a href="#">Section 4.1.8</a>
Network Reached	<a href="#">Section 4.1.9</a>
Offered	<a href="#">Section 4.1.10</a>
Originated	<a href="#">Section 4.1.11</a>
Queued	<a href="#">Section 4.1.12</a>
Retrieved	<a href="#">Section 4.1.13</a>
Service Initiated	<a href="#">Section 4.1.14</a>
Transferred	<a href="#">Section 4.1.15</a>

### Call Control Events Descriptions

The entire ECMA CSTA III standard covering call control events is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

Additionally to the mandatory parameters the following conditional parameters are mandatory :

- 1) localConnectionInfo The switching function supports only device-type monitors.
- 2) servicesPermitted The switching function supports the Dynamic Feature Availability option.
- 3) callLinkageDataList, CallLinkagedata The switching function supports the call linkage feature.

#### 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.

## Events

Common situations that generate this event include:

- Two step conferencing situations (manual and service initiated)

### Parameters

**Table 184: Conferenced Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
primaryOldCall	ConnectionID	M	<p>The switching function provides the local view option.</p> <p>The primaryOldCall specifies the held call at the conferencing device, otherwise at other participating devices it is the only call involved in the conference from the perspective of that device.</p> <p>See Usage Note #2</p>
secondaryOldCall	ConnectionID	C	<p>The switching function provides the local view option.</p> <p>The secondaryOldCall specifies the active call at the conferencing device, otherwise it is not provided.</p> <p>See Usage Note #2</p>
conferencingDevice	SubjectDeviceID	M	<p>Specifies the device ID of the conferencing device</p> <p>The device ID may be represented by Diallable Digits or Device Number.</p>
addedParty	SubjectDeviceID	M	<p>The addedParty specifies the device ID of the device, that belongs to the active call of the conference.</p> <p>The device ID may be represented by Switching Function Representation or Device Number or Not Known.</p> <p>See Usage Note #2</p>
conferenceConnections	ConnectionList	M	<p>The conferenceConnections parameter is a list that contains the new ConnectionIDs and the old ConnectionIDs of the conference and for externally located devices the associated Network Interface DeviceID.</p> <p>The endPoint DeviceID parameter is provided only for external calls with network information. It is the representation of the externally located device.</p> <p>See Usage Note #2</p>

Parameter Name	Contents	M/C/O	Comments
localConnectionInfo	LocalConnectionState	M	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. Conferencing device - Connected  Other devices - See Table 17-143 in ECMA 269
userData	UserData	C	Specifies user information that is related to the call.
cause	EventCause	M	The cause parameter specifies the reason for the event. Possible values are:  Normal  Network Signal  In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
callLinkageDataList	List	M	Specifies the global call data and thread data associated with the call. The parameter consists of the following components:  <ul style="list-style-type: none"> <li>• newCallLinkageData (M) CallLinkageData - specifies the call linkage data associated with the resulting call.</li> <li>• oldCallLinkageData (M) CallLinkageData - specifies the call linkage data that was discarded as the result of the transfer.</li> </ul>

### Usage Notes

- 1) The contents of the primaryOldCall and the secondaryOldCall parameters is a "local view" of the connections at a device before the conference has been completed.
- 2) According to the ECMA 269 standard the primaryOldCall and the secondaryOldCall should specify the first / second call visible at the monitored device. Due to switching function limitations the OpenScape 4000 interpretation of primaryOldCall, secondaryOldCall and addedParty is not fully compliant with the CSTA phase III standard. However the switching function provides full ConferenceConnections list to enable applications to properly track the conference.
- 3) When a conference of more then 3 parties is created, the switching function reuses the callID of the 3-party conference as the new callID in the conferenced event.
- 4) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The

combination of these two components provides call linkage data that is globally unique. Restrictions:

- inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
- new Global ID may be created in a few cases where the Global ID should remain the same (e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

### 4.1.1 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 PART 3 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.

#### Parameters

**Table 185: Bridged Parameters**

Parameter Name	Content	M/C/O	Comments
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	Specifies the local connection state of the device associated with the Monitor Cross Reference ID. Conferencing device - Connected  Other devices - See Table 17-143 in ECMA 269
correlatorData	CorrelatorData	O	Specifies the correlator data associated with the call. Possible values
cause	EventCause	M	Specifies the reason for the event.

Parameter Name	Content	M/C/O	Comments
servicesPermitted	ServicesPermitted	C	Specifies a list of the call control services that can be applied to the local connection.
callLinkageData	CallLinkageData	O	Specifies the global call data and thread data associated with the call.
privateData	CSTAPrivateData	O	Specifies the non-standardized information attached to the event.

## 4.1.2 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.

---

**NOTICE:** This event is not used when a device is removed from a call due to the call being transferred or diverted to another device. (The Transferred and Diverted events are used in these cases.)

---

### Parameters

**Table 186: Conferenced Event Parameters**

Parameter Name	Content	M/C/O	Comments
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.  When the subject device ID is not known, the device ID component of the Connection ID is not provided either.
releasingDevice	SubjectDeviceID	M	Specifies the device that dropped from the call.  The device ID may be represented by Switching Function Representation or Device Number or Not Known..

## Events

Parameter Name	Content	M/C/O	Comments
localConnectionInfo	LocalConnectionState	M	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"> <li>For the clearing device: Null</li> <li>For the other devices left in the call: (unaffected)</li> </ul>
userData	UserData	C	Specifies user information that is related to the call.
cause	EventCause	M	The cause parameter specifies the reason for the event. Possible values are: <ul style="list-style-type: none"> <li>Call Not Answered</li> <li>Normal Clearing</li> </ul> In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
callLinkageData	CallLinkageData	O	Specifies the global call data and thread data associated with the call.
privateEventCause (privateData)	PrivateEventCause	O	Specifies an additional cause for the event. Possible values are: <ul style="list-style-type: none"> <li>Single Step Call Transfer</li> </ul>

**Table 187: Connection Cleared Event Parameters**

Parameter Name	Content	M/C/O	Comments
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. <p>When the subject device ID is not known, the device ID component of the Connection ID is not provided either.</p>
releasingDevice	SubjectDeviceID	M	Specifies the device that dropped from the call. <p>The device ID may be represented by Switching Function Representation or Device Number or Not Known..</p>

Parameter Name	Content	M/C/O	Comments
localConnectionInfo	LocalConnectionState	M	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"> <li>For the clearing device: Null</li> <li>For the other devices left in the call: (unaffected)</li> </ul>
userData	UserData	C	Specifies user information that is related to the call.
cause	EventCause	M	The cause parameter specifies the reason for the event. Possible values are: <ul style="list-style-type: none"> <li>Call Not Answered</li> <li>Normal Clearing</li> </ul> In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
callLinkageData	CallLinkageData	O	Specifies the global call data and thread data associated with the call.
privateEventCause (privateData)	PrivateEventCause	O	Specifies an additional cause for the event. Possible values are: <ul style="list-style-type: none"> <li>Single Step Call Transfer</li> </ul>

### Usage Notes

- 1) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same (e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.1.3 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.

## Events

- A call has been assigned to a distribution device such as an ACD or hunt group.

### Parameters

**Table 188: Delivered Event Parameters**

Parameter Name	Content	M/C/O	Comments
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.  The device ID may be represented by Switching Function Representation or Device Number.
callingDevice	CallingDeviceID	M	For internal or outgoing calls the callingDevice parameter includes the Switching Function Representation of the calling device, for incoming calls the callingDevice parameter may include the Automatic Number Identification ( ANI ), Station Identification ( SID ), or in the case of a private network , information provided by satellite operations ( SATOPS ).  Not Known indicates that the switching function cannot identify the calling party.
calledDevice	CalledDeviceID	M	The calledDevice is either the originally dialed digits or the internal representation of the originally dialed number ( after digit translation ) or the DNIS in case of an incoming call.  The format of the calledDevice is Diallable Digits.  "Not Known" indicates that the switching function cannot identify the called party.

Parameter Name	Content	M/C/O	Comments
lastRedirectionDevice	RedirectionDeviceID	M	<p>The lastRedirectionDevice is the last known device from which the current call was routed. It is represented by only one device identifier and in actuality the call might have been routed several times before arriving at the final destination, and the last known device is not necessarily the last in the chain.</p> <p>The switching function provides it in only the first event after the redirection.</p> <p>"Not Known" indicates that the call has been redirected but the switching function cannot identify DeviceID or it will be present in case of an Immediate Forwarding, where forwarding is triggered before the call is delivered.</p> <p>"Not Specified" indicates that the switching function cannot determine whether or not the call has ever been redirected.</p>
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from.
localConnectionInfo	LocalConnectionState	M	<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"> <li>• For the alerting device: Alerting</li> <li>• For the calling device: (unaffected)</li> </ul>
userData	UserData	C	Specifies user information that is related to the call.

## Events

Parameter Name	Content	M/C/O	Comments
cause	EventCause	M	<p>The cause parameter specifies the reason for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>• Call Back</li> <li>• Call Forward</li> <li>• Call Forward -Busy</li> <li>• Call Forward -Immediate</li> <li>• Call Forward -No Answer</li> <li>• Distributed</li> <li>• Entering Distribution</li> <li>• Multiple Alerting</li> <li>• Network Signal</li> <li>• No Available Agents</li> <li>• Normal</li> <li>• Remains in Queue</li> <li>• Recall</li> <li>• Single Step Transfer</li> <li>• Overflow</li> </ul> <p>In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.</p>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.
associatedCallingDevice	AssociatedCallingDeviceID	C	<p>The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.</p> <p>If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.</p>

Parameter Name	Content	M/C/O	Comments
associatedCalledDevice	AssociatedCalledDeviceID	C	<p>The associatedCalledDevice parameter specifies the Network Interface Device associated with the called device if the call is outgoing.</p> <p>If the Network Interface Device leaves the call, the associatedCalledDevice will not be provided any more.</p> <p>If the call has more than one outgoing connection ( for ex. Make Predictive Call with external destination and external ACD agents ) , the associatedCalledDevice will be set to "Not Known".</p> <p>For incoming external calls it might provide a deviceID, that is associated with the originally called device. ( e.g. internal representation of an RCG )</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.
executiveDeviceID (privateData)	CallingDeviceID	O	Specifies the Executive device in case of a CHESE call. See further information in ADG Volume 2 Call Scenarios.

#### Usage Notes

- 1) As the switching function does not provide the Diverted event for all devices in a call, the computing function needs the alertingDevice, calledDevice, lastRedirectionDevice and cause parameters to properly track the progress of the call. However the switching function may provide "Not Specified" lastRedirectiondevice in specific situations. For example when an RCG distributes a call to an agent.
- 2) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
- 3) • inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.  
• new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).
- 4) Delivered event is sent only when alerting state reached. Alerting state might be missing on in some network wide call scenarios.

### 4.1.4 Digits Dialed

The Digits Dialed event indicates that a call or feature is being attempted from a device. It is sent for example, when there is partial dialling in a MakeCall request, so the called directory number is not complete.

Digits Dialed event will be sent also, when a call is being attempted from a device.

It is implementation specific how many digits are buffered before they are sent in a CSTA Digits

All digits are buffered and sent together in one event.

This provides more information for the application in the following cases:

- The called device in the CSTA events contains the already resolved number, not the really dialed digits.
- In case of incomplete dialling the application has no information about the dialed digits.
- If an invalid number has been dialed, the application does not receive them.

The Digits Dialed event is used to inform the application about each number dialed by the user. The dialed digits must be reported in case dialling incomplete numbers (the user starts dialing a number, then goes onhook), dialing invalid numbers, dialing non-existent numbers, dialing a device which is in OOS.

### Parameters

**Table 189: Digits Dialed Event Parameters**

Parameter Name	Content	M/C/O	Comments
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 ( Switching Function Representation ).
diallingSequence	DeviceID	M	Specifies the sequence of digits that was dialed.
localConnectionInfo	LocalConnectionState	M	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"> <li>• For the device dialling the digits: Initiated.</li> </ul>
cause	EventCause	M	The cause parameter specifies the reason for the event. The only Possible value is :  <ul style="list-style-type: none"> <li>• Normal</li> </ul>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.

Parameter Name	Content	M/C/O	Comments
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.
associatedCallingDevice	AssociatedCallingDeviceID	C	<p>The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.</p> <p>If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.</p>
associatedCalledDevice	AssociatedCalledDeviceID	C	<p>The associatedCalledDevice parameter specifies the Network Interface Device associated with the called device if the call is outgoing.</p> <p>If the Network Interface Device leaves the call, the associatedCalledDevice will not be provided any more.</p> <p>If the call has more than one outgoing connection ( for ex. Make Predictive Call with external destination and external ACD agents ) , the associatedCalledDevice will be set to "Not Known".</p> <p>For incoming external calls it might provide a deviceID, that is associated with the originally called device. ( e.g. internal representation of an RCG )</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.

### Usage Notes

- 1) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same (e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

### 4.1.5 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, Recall, etc.
- A call leaves an ACD/Hunt group device to be redirected to an agent, an extension, another ACD/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 Deflect Service is successfully invoked.

#### Parameters

**Table 190: Diverted Event Parameters**

Parameter Name	Content	M/C/O	Comments
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.  The device ID may be represented by Switching Function Representation or Device Number.
newDestination	SubjectDeviceID	M	Specifies the device to which the call was diverted.  The device ID may be represented by Switching Function Representation or Device Number or Not Known.
callingDevice	CallingDeviceID	O	For internal or outgoing calls the callingDevice parameter includes the Switching Function Representation of the calling device, for incoming calls the callingDevice parameter may include the Automatic Number Identification ( ANI ), Station Identification ( SID ), or in the case of a private network , information provided by satellite operations ( SATOPS ).  In case the switching function cannot identify the calling party, it will be not present.

Parameter Name	Content	M/C/O	Comments
calledDevice	CalledDeviceID	O	<p>The calledDevice is either the originally dialed digits or the internal representation of the originally dialed number ( after digit translation ) or the DNIS in case of an incoming call.</p> <p>The format of the calledDevice is Diallable Digits.</p> <p>In case the switching function cannot identify the called party, it will be not present.</p>
lastRedirectionDevice	RedirectionDeviceID	M	<p>The lastRedirectionDevice is the last known device from which the current call was routed. It is represented by only one device identifier and in actuality the call might have been routed several times before arriving at the final destination, and the last known device is not necessarily the last in the chain.</p> <p>The switching function provides it in only the first event after the redirection.</p> <p>"Not Known" indicates that the call has been redirected but the switching function cannot identify DeviceID or it will be present in case of an Immediate Forwarding, where forwarding is triggered before the call is delivered.</p> <p>"Not Specified" indicates that the switching function cannot determine whether or not the call has ever been redirected.</p>
localConnectionInfo	LocalConnectionState	M	<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"> <li>• For the diverting device: Null</li> <li>• For the other devices left in the call: (unaffected)</li> </ul>

## Events

Parameter Name	Content	M/C/O	Comments
cause	EventCause	M	<p>The cause parameter specifies the reason for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>• Call Forward -No Answer</li> <li>• Call Pickup</li> <li>• Distributed</li> <li>• Normal</li> <li>• Recall</li> <li>• Overflow</li> </ul> <p>In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.</p>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.
associatedCallingDevice	AssociatedCallingDeviceID	C	<p>The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.</p> <p>If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.</p>

Parameter Name	Content	M/C/O	Comments
associatedCalledDevice	AssociatedCalledDeviceID	C	<p>The associatedCalledDevice parameter specifies the Network Interface Device associated with the called device if the call is outgoing.</p> <p>If the Network Interface Device leaves the call, the associatedCalledDevice will not be provided any more.</p> <p>If the call has more than one outgoing connection ( for ex. Make Predictive Call with external destination and external ACD agents ) , the associatedCalledDevice will be set to "Not Known".</p> <p>For incoming external calls it might provide a deviceID, that is associated with the originally called device. ( e.g. internal representation of an RCG )</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.

### Usage Notes

- 1) The switching function sends the Diverted event only to the divertingDevice (as indicated through the capabilities exchange services).
- 2) CallingDevice and calledDevice are optional for the Diverted event, because the switching function never sends Diverted events in the case of Immediate Forwarding.
- 3) As the switching function does not provide the Diverted event for all devices in a call, the computing function needs the alertingDevice, calledDevice, lastRedirectionDevice and cause parameters to properly track the progress of the call. However the switching function may provide "Not Specified" lastRedirectiondevice in specific situations. For example when an RCG distributes a call to an agent.
- 4) The switching function does not provide userData with this event, although it is required by the standard.
- 5) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.1.6 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.

### Parameters

**Table 191: Established Event Parameters**

Parameter Name	Content	M/C/O	Comments
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.  The device ID may be represented by Switching Function Representation or Device Number.
callingDevice	CallingDeviceID	M	For internal or outgoing calls the callingDevice parameter includes the Switching Function Representation of the calling device, for incoming calls the callingDevice parameter may include the Automatic Number Identification ( ANI ), Station Identification ( SID ), or in the case of a private network , information provided by satellite operations ( SATOPS ).  Not Known indicates that the switching function cannot identify the calling party.
calledDevice	CalledDeviceID	M	The calledDevice is either the originally dialed digits or the internal representation of the originally dialed number ( after digit translation ) or the DNIS in case of an incoming call.  The format of the calledDevice is Diallable Digits.  "Not Known" indicates that the switching function cannot identify the called party.

Parameter Name	Content	M/C/O	Comments
lastRedirectionDevice	RedirectionDeviceID	M	<p>The lastRedirectionDevice is the last known device from which the current call was routed. It is represented by only one device identifier and in actuality the call might have been routed several times before arriving at the final destination, and the last known device is not necessarily the last in the chain.</p> <p>The switching function provides it in only the first event after the redirection.</p> <p>"Not Known" indicates that the call has been redirected but the switching function cannot identify DeviceID or it will be present in case of an Immediate Forwarding, where forwarding is triggered before the call is delivered.</p> <p>"Not Specified" indicates that the switching function cannot determine whether or not the call has ever been redirected.</p>
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from.
localConnectionInfo	LocalConnectionState	M	<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"> <li>For the answering device: Connected</li> <li>For the calling device: (unaffected - never Null)</li> </ul>
userData	UserData	C	Specifies user information that is related to the call.
cause	EventCause	M	<p>The cause parameter specifies the reason for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>Call Pickup</li> <li>Park</li> <li>Normal</li> <li>Network Signal</li> <li>Single Step Transfer</li> </ul> <p>In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.</p>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.

## Events

Parameter Name	Content	M/C/O	Comments
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.
associatedCallingDevice	AssociatedCallingDeviceID	C	<p>The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.</p> <p>If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.</p>
associatedCalledDevice	AssociatedCalledDeviceID	C	<p>The associatedCalledDevice parameter specifies the Network Interface Device associated with the called device if the call is outgoing.</p> <p>If the Network Interface Device leaves the call, the associatedCalledDevice will not be provided any more.</p> <p>If the call has more than one outgoing connection ( for ex. Make Predictive Call with external destination and external ACD agents ) , the associatedCalledDevice will be set to "Not Known".</p> <p>For incoming external calls it might provide a deviceID, that is associated with the originally called device. ( e.g. internal representation of an RCG )</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.
executiveDeviceID (privateData)	CallingDeviceID	O	Specifies the Executive device in case of a CHESE call. See further information in ADG Volume 2 Call Scenarios.

### Usage Notes

- 1) As the switching function does not provide the Diverted event for all devices in a call, the computing function needs the alertingDevice, calledDevice, lastRedirectionDevice and cause parameters to properly track the progress of the call. However the switching function may provide "Not Specified" lastRedirectiondevice in specific situations. For example when an RCG distributes a call to an agent.
- 2) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.

- new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.1.7 Failed

The Failed event indicates that a call cannot be completed or a connection has entered the Fail state for any reasons.

### Parameters

**Table 192: Failed Event Parameters**

Parameter Name	Content	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
failedConnection	ConnectionID	M	Specifies the connection that failed. See Usage Notes #1
failingDevice	SubjectDeviceID	M	Specifies the device that failed.  The device ID may be represented by Switching Function Representation or Device Number.
callingDevice	CallingDeviceID	M	For internal or outgoing calls the callingDevice parameter includes the Switching Function Representation of the calling device, for incoming calls the callingDevice parameter may include the Automatic Number Identification ( ANI ), Station Identification ( SID ), or in the case of a private network , information provided by satellite operations ( SATOPS ).  Not Known indicates that the switching function cannot identify the calling party.
calledDevice	CalledDeviceID	M	The calledDevice is either the originally dialed digits or the internal representation of the originally dialed number ( after digit translation ) or the DNIS in case of an incoming call.  The format of the calledDevice is Diallable Digits.  "Not Known" indicates that the switching function cannot identify the called party.

## Events

Parameter Name	Content	M/C/O	Comments
lastRedirectionDevice	RedirectionDeviceID	M	<p>The lastRedirectionDevice is the last known device from which the current call was routed. It is represented by only one device identifier and in actuality the call might have been routed several times before arriving at the final destination, and the last known device is not necessarily the last in the chain.</p> <p>The switching function provides it in only the first event after the redirection.</p> <p>"Not Known" indicates that the call has been redirected but the switching function cannot identify DeviceID or it will be present in case of an Immediate Forwarding, where forwarding is triggered before the call is delivered.</p> <p>"Not Specified" indicates that the switching function cannot determine whether or not the call has ever been redirected.</p>
originatingNIDConnection	ConnectionID	O	Specifies the connection of the Network Interface Device (NID) that the call originated from.
localConnectionInfo	LocalConnectionState	M	<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> <p>For the failing device: Fail</p> <p>For the other devices left in the call: (unaffected)</p>
cause	EventCause	M	<p>The cause parameter specifies the reason for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>Blocked ( Blocked is reported for blocked situations and instead of Destination Not Obtainable and Invalid Number Format )</li> <li>Busy</li> </ul> <p>In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.</p>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.

Parameter Name	Content	M/C/O	Comments
associatedCallingDevice	AssociatedCallingDeviceID	C	<p>The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.</p> <p>If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.</p>
associatedCalledDevice	AssociatedCalledDeviceID	C	<p>The associatedCalledDevice parameter specifies the Network Interface Device associated with the called device if the call is outgoing.</p> <p>If the Network Interface Device leaves the call, the associatedCalledDevice will not be provided any more.</p> <p>If the call has more than one outgoing connection ( for ex. Make Predictive Call with external destination and external ACD agents ) , the associatedCalledDevice will be set to "Not Known".</p> <p>For incoming external calls it might provide a deviceID, that is associated with the originally called device. ( e.g. internal representation of an RCG )</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.
executiveDeviceID (privateData)	CallingDeviceID	O	Specifies the Executive device in case of a CHESE call. See further information in ADG Volume 2 Call Scenarios.

### Usage Notes

- 1) A complete connection identifier is provided as the failedConnection parameter and the Failed event is reported to all active monitors associated with the call.
- 2) The switching function does not provide userData with this event, although it is required by the standard.
- 3) As the switching function does not provide the Diverted event for all devices in a call, the computing function needs the alertingDevice, calledDevice, lastRedirectionDevice and cause parameters to properly track the progress of the call. However the switching function may provide "Not Specified" lastRedirectiondevice in specific situations. For example when an RCG distributes a call to an agent.
- 4) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.

## Events

- new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

### 4.1.8 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).

#### Parameters

**Table 193: Hold Event Parameter**

Parameter Name	Content	M/C/O	Comments
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.  The device ID may be represented by Switching Function Representation or Device Number.
localConnectionInfo	LocalConnectionState	M	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"><li>• For the holding device: Hold</li><li>• For the other devices left in the call: (unaffected)</li></ul>
cause	EventCause	M	The cause parameter specifies the reason for the event. Possible values are: <ul style="list-style-type: none"><li>• Alternate</li><li>• Consultation</li><li>• Normal</li></ul> In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
callLinkageData	CallLinkageData	O	Specifies the global call data and thread data associated with the call.

### Usage Notes

- 1) When generated due to silent monitoring , the Held event indicates only that the silent monitored agent is not currently active in a call.
- 2) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.1.9 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.

### Parameters

**Table 194: Network Reached Event Parameters**

Parameter Name	Content	M/C/O	Comments
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.

## Events

Parameter Name	Content	M/C/O	Comments
callingDevice	CallingDeviceID	M	<p>For internal or outgoing calls the callingDevice parameter includes the Switching Function Representation of the calling device, for incoming calls the callingDevice parameter may include the Automatic Number Identification ( ANI ), Station Identification ( SID ), or in the case of a private network , information provided by satellite operations ( SATOPS ).</p> <p>Not Known indicates that the switching function cannot identify the calling party.</p>
calledDevice	CalledDeviceID	M	<p>The calledDevice is either the originally dialed digits or the internal representation of the originally dialed number ( after digit translation ) or the DNIS in case of an incoming call.</p> <p>The format of the calledDevice is Diallable Digits.</p> <p>"Not Known" indicates that the switching function cannot identify the called party.</p>
lastRedirectionDevice	RedirectionDeviceID	M	<p>The lastRedirectionDevice is the last known device from which the current call was routed. It is represented by only one device identifier and in actuality the call might have been routed several times before arriving at the final destination, and the last known device is not necessarily the last in the chain.</p> <p>The switching function provides it in only the first event after the redirection.</p> <p>"Not Known" indicates that the call has been redirected but the switching function cannot identify DeviceID or it will be present in case of an Immediate Forwarding, where forwarding is triggered before the call is delivered.</p> <p>"Not Specified" indicates that the switching function cannot determine whether or not the call has ever been redirected.</p>
originatingNIDConnection	ConnectionID	Θ	<p>Specifies the connection of the Network Interface Device (NID) that the call originated from.</p>

Parameter Name	Content	M/C/O	Comments
localConnectionInfo	LocalConnectionState	M	<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"> <li>For the Network Interface Device: Connected</li> <li>For the other devices left in the call: (unaffected)</li> </ul>
userData	UserData	C	Specifies user information that is related to the call.
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> <ul style="list-style-type: none"> <li>networkType (M) - The complete set of possible values is:</li> </ul> <p>ISDN public</p> <p>Non-ISDN public</p> <p>ISDN private</p> <p>Non-ISDN private</p> <p>Other</p>
cause	EventCause	M	<p>The cause parameter specifies the reason for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>Call Forward - Immediate</li> <li>Call Forward - Busy</li> <li>Distributed</li> <li>No Agents Available</li> <li>Normal</li> <li>Transfer</li> </ul> <p>In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.</p>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.

## Events

Parameter Name	Content	M/C/O	Comments
associatedCallingDevice	AssociatedCallingDeviceID	C	The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.  If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.

### Usage Notes

- 1) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.1.10 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.

### Parameters

**Table 195: Offered Event Parameters**

Parameter Name	Content	M/C/O	Comments
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.

Parameter Name	Content	M/C/O	Comments
calledDevice	CalledDevice ID	M	Specifies the originally called device.
lastRedirectionDevice	RedirectionDeviceID	M	Specifies the previously known redirected from device.
localConnectionInfo	LocalConnectionState	M	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.  For the device initiating the call: Connected
cause	EventCause	M	Specifies the reason for the event.
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.
security	CSTASecurityData	O	Specifies timestamp information, message sequence number, and security information.
privateData	CSTAPrivateData	O	Non-standardized information attached to the event.

### 4.1.11 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).

#### Parameters

**Table 196: Originated Event Parameters**

Parameter Name	Content	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
originatedConnection	ConnectionID	M	Specifies the connection at which the call originated.

## Events

Parameter Name	Content	M/C/O	Comments
callingDevice	SubjectDeviceID	M	For internal or outgoing calls the callingDevice parameter includes the Switching Function Representation of the calling device, for incoming calls the callingDevice parameter may include the Automatic Number Identification ( ANI ), Station Identification ( SID ), or in the case of a private network , information provided by satellite operations ( SATOPS ).
calledDevice	CalledDeviceID	M	The calledDevice is the originally dialed digits.  The format of the calledDevice is Diallable Digits.
localConnectionInfo	LocalConnectionState	M	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.  For the device initiating the call: Connected
cause	EventCause	M	The cause parameter specifies the reason for the event. Possible values are: <ul style="list-style-type: none"><li>• Call Back</li><li>• Make Call</li><li>• Normal</li><li>• Consultation</li></ul> In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.
associatedCallingDevice	AssociatedCallingDeviceID	C	The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.  If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.

Parameter Name	Content	M/C/O	Comments
associatedCalledDevice	AssociatedCalledDeviceID	C	<p>The associatedCalledDevice parameter specifies the Network Interface Device associated with the called device if the call is outgoing.</p> <p>If the Network Interface Device leaves the call, the associatedCalledDevice will not be provided any more.</p> <p>If the call has more than one outgoing connection ( for ex. Make Predictive Call with external destination and external ACD agents ) , the associatedCalledDevice will be set to "Not Known".</p> <p>For incoming external calls it might provide a deviceID, that is associated with the originally called device. ( e.g. internal representation of an RCG )</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.

### Usage Notes

- 1) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.1.12 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 a group device.
- A call is queued (camped on or parked, for example) at a device.

### Parameters

**Table 197: Queued Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.

## Events

Parameter Name	Contents	M/C/O	Comments
queuedConnection	ConnectionID	M	Specifies the queued connection.
queue	SubjectDeviceID	M	Specifies the queue device.  The device ID may be represented by Switching Function Representation or Device Number.
callingDevice	CallingDeviceID	M	For internal or outgoing calls the callingDevice parameter includes the Switching Function Representation of the calling device, for incoming calls the callingDevice parameter may include the Automatic Number Identification ( ANI ), Station Identification ( SID ), or in the case of a private network , information provided by satellite operations ( SATOPS ).  Not Known indicates that the switching function cannot identify the calling party.
calledDevice	CalledDeviceID	M	The calledDevice is either the originally dialed digits or the internal representation of the originally dialed number ( after digit translation ) or the DNIS in case of an incoming call.  The format of the calledDevice is Diallable Digits.  "Not Known" indicates that the switching function cannot identify the called party.
lastRedirectionDevice	RedirectionDeviceID	M	The lastRedirectionDevice is the last known device from which the current call was routed. It is represented by only one device identifier and in actuality the call might have been routed several times before arriving at the final destination, and the last known device is not necessarily the last in the chain.  The switching function provides it in only the first event after the redirection.  "Not Known" indicates that the call has been redirected but the switching function cannot identify DeviceID or it will be present in case of an Immediate Forwarding, where forwarding is triggered before the call is delivered.  "Not Specified" indicates that the switching function cannot determine whether or not the call has ever been redirected.

Parameter Name	Contents	M/C/O	Comments
localConnectionInfo	LocalConnectionState	M	<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> <p>For the queue device: Queued</p> <p>For the other devices left in the call: (unaffected)</p>
userData	UserData	C	Specifies user information that is related to the call.
cause	EventCause	M	<p>The cause parameter specifies the reason for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>• Camp On</li> <li>• No Available Agents</li> <li>• Normal</li> <li>• Overflow</li> <li>• Park</li> <li>• Remains in Queue</li> </ul> <p>In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.</p>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	O	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	O	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.
associatedCallingDevice	AssociatedCallingDeviceID	C	<p>The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.</p> <p>If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.</p>

## Events

Parameter Name	Contents	M/C/O	Comments
associatedCalledDevice	AssociatedCalledDeviceID	C	<p>The associatedCalledDevice parameter specifies the Network Interface Device associated with the called device if the call is outgoing.</p> <p>If the Network Interface Device leaves the call, the associatedCalledDevice will not be provided any more.</p> <p>If the call has more than one outgoing connection ( for ex. Make Predictive Call with external destination and external ACD agents ) , the associatedCalledDevice will be set to "Not Known".</p> <p>For incoming external calls it might provide a deviceID, that is associated with the originally called device. ( e.g. internal representation of an RCG )</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.
executiveDeviceID (privateData)	CallingDeviceID	O	Specifies the Executive device in case of a CHESE call. See further information in ADG Volume 2 Call Scenarios.

### Usage Notes

- 1) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

### 4.1.13 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.

## Parameters

Table 198: Retrieved Event Parameters

Parameter Name	Content	M/C/O	Comments
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.  The device ID may be represented by Switching Function Representation or Device Number.
localConnectionInfo	LocalConnectionState	M	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.  For the retrieving device: Connected  For the other devices left in the call: (unaffected)
cause	EventCause	M	The cause parameter specifies the reason for the event. Possible values are: <ul style="list-style-type: none"><li>• Alternate</li><li>• Normal</li></ul> In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
callLinkageData	CallLinkageData	O	Specifies the global call data and thread data associated with the call.

## Usage Notes

- 1) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - a) inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - b) new Global ID may be created in a few cases where the Global ID should remain the same(e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

### 4.1.14 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).

#### Parameters

**Table 199: Service Initiated Event Parameters**

Parameter Name	Content	M/C/O	Comments
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.  The device ID may be represented by Switching Function Representation or Device Number.
localConnectionInfo	LocalConnectionState	M	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.  When the device is initiating a service of some form: Initiated
cause	EventCause	M	The cause parameter specifies the reason for the event. Possible values are: <ul style="list-style-type: none"> <li>• Normal</li> <li>• Call back</li> <li>• Make Call</li> </ul> In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.

Parameter Name	Content	M/C/O	Comments
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
networkCallingDevice	NetworkCallingDeviceID	Ø	The networkCalling parameter specifies the ANI number if it is provided by the network for incoming calls. Otherwise it is not present.
networkCalledDevice	NetworkCalledDeviceID	Ø	The networkCalled parameter specifies the DNIS number if it is provided by the network for incoming calls. Otherwise it is not present.
associatedCallingDevice	AssociatedCallingDeviceID	C	<p>The associatedCallingDevice parameter specifies the Network Interface Device associated with the calling device if the call is incoming.</p> <p>If the Network Interface Device leaves the call, the associatedCallingDevice will not be provided any more.</p>
callLinkageData	CallLinkageData	M	Specifies the global call data and thread data associated with the call.
privateEventCause (privateData)	PrivateEventCause	O	<p>Specifies an additional cause for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>Single Step Call Transfer</li> </ul>

### Usage Notes

- 1) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The combination of these two components provides call linkage data that is globally unique. Restrictions:
  - inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
  - new Global ID may be created in a few cases where the Global ID should remain the same (e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.1.15 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).

## Parameters

Table 200: Transferred Event Parameters

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
primaryOldCall	ConnectionID	M	<p>The switching function provides the local view option.</p> <p>The primaryOldCall specifies the held call at the transferring device, otherwise at other participating devices it is the only call involved in the transfer from the perspective of that device.</p> <p>See Usage Note #2</p>
secondaryOldCall	ConnectionID	C	<p>The switching function provides the local view option.</p> <p>The secondaryOldCall specifies the active call at the transferring device, otherwise it is not provided.</p> <p>See Usage Note #2</p>
transferringDevice	SubjectDeviceID	M	<p>Specifies the device that transferred the call.</p> <p>The device ID may be represented by Switching Function Representation or Device Number.</p>
transferredToDevice	SubjectDeviceID	M	<p>Specifies the transferred to device.</p> <p>The device ID may be represented by Switching Function Representation or Device Number.</p>
transferredConnections	ConnectionList	M	<p>For the transferring device the transferredConnections parameter is a list that contains the new ConnectionIDs and the old ConnectionIDs of the transfer and for externally located devices the associated Network Interface DeviceID.</p> <p>For the other participating devices the old ConnectionID, that does not belong to the device will not be provided.</p> <p>The endPoint DeviceID parameter is provided only for external calls with network information. It is the representation of the externally located device.</p>

Parameter Name	Contents	M/C/O	Comments
localConnectionInfo	LocalConnectionState	M	<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> <p>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</p> <p>For the other devices associated with the transfer: (unaffected)</p>
userData	UserData	C	Specifies user information that is related to the call.
cause	EventCause	M	<p>The cause parameter specifies the reason for the event. Possible values are:</p> <ul style="list-style-type: none"> <li>• Normal</li> <li>• Single Step Transfer</li> <li>• Transfer</li> </ul> <p>In general, a cause code ( apart from Normal ) is only reported in the first event after the condition has been detected.</p>
servicesPermitted	ServicesPermitted	M	Specifies a list of the call control services that can be applied to the local connection.
callLinkageDataList	List	M	<p>Specifies the global call data and thread data associated with the call.</p> <p>newCallLinkageData (M) CallLinkageData - specifies the call linkage data associated with the resulting call.</p> <p>oldCallLinkageData (M) CallLinkageData - specifies the call linkage data that was discarded as the result of the transfer.</p>

### Usage Notes

- 1) The contents of the primaryOldCall and the secondaryOldCall parameters is a "local view" of the connections at a device before the conference has been completed.
- 2) According to the ECMA 269 standard the primaryOldCall and the secondaryOldCall should specify the first / second call visible at the monitored device. Due to switching function limitations the OpenScape 4000 interpretation of primaryOldCall, secondaryOldCall is not fully compliant with the CSTA phase III standard.
- 3) Transferred events are generated also in case of Route Optimization. See ADG Volume 2 Call Scenarios for further information.
- 4) The switching function that creates the callLinkageData ensures that it is globally unique by providing the switching sub-domain name component along with a switching sub-domain unique identifier component. The

## Events

### Call Associated Feature Events

combination of these two components provides call linkage data that is globally unique. Restrictions:

- inconsistent Call Linkage Data (Thread ID and Global ID) over networking conferences.
- new Global ID may be created in a few cases where the Global ID should remain the same (e.g. redirection related scenarios : Deflect, Call Forwarding scenarios).

## 4.2 Call Associated Feature Events

### Call Associated Events Summary

**Table 201: Support of Call Associated Events**

Call Associated Event	Event Description Section
Call Information	<a href="#">Section 4.2.1</a>
Digits Generated	<a href="#">Section 4.2.2</a>
Telephony Tones Generated	<a href="#">Section 4.2.3</a>

### Call Associated Events Descriptions

The entire ECMA CSTA III standard covering call associated events is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

### 4.2.1 Call Information

The Call Information event indicates that call associated information has been collected/updated for a call.

Common situations that generate this event include:

- The account code has been included or changed via a call control service request, or has been manually entered.
- User data data has been received independent of call activity (via the Send User Information service or as a result of non-call related activity from an external network).
- Services permitted information has been updated as a result of another connection changing state.
- Call Linkage Data (Global Call Id and Thread Id) has been changed in a remote node, but has not caused any state transition in the local node. A Call Information event is sent by the switching function whenever the call linkage data for a particular call is changed asynchronously, i.e. NOT at the same time as a callstate change for the monitored device. This occurs e.g. when the call is transferred or conferenced on the remote node.

### Parameters

**Table 202: Call Information Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
connection	ConnectionID	M	Indicates the connection of the device responsible for associating the information with the call
device	Device ID	M	Indicates the device responsible for associating the information with the call
accountInfo	AccountInfo	C	Indicates the account code associated with the call.
servicesPermitted	ServicesPermitted	C	Indicates a list of services that can be applied to the specified connection.
userData	UserData	C	Provides information that has been sent from another device.
callLinkageDataList	List	C	<p>Specifies the global call data and thread data associated with the call. The parameter consists of the following components:</p> <ul style="list-style-type: none"> <li>newCallLinkageData (M) CallLinkageData - specifies the call linkage data associated with the resulting call.</li> <li>oldCallLinkageData (M) CallLinkageData - specifies the call linkage data that was discarded as the result of the transfer.</li> </ul>

## 4.2.2 Digits Generated

The Digits Generated event indicates that DTMF digits have been generated at a device.

Common situations that generate this event include:

- The switching function generates DTMF digits for the device of a given connection as the result of the Generate Digits service.

### Parameters

**Table 203: Digits Generated Event Parameters**

Parameter Name	Contents	M/C/O	Comments
MonitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.

## Events

Parameter Name	Contents	M/C/O	Comments
connection	ConnectionID	M	The connection at the device
digitGeneratedList	Characters (64)	M	<p>The sequence of digits generated.</p> <ul style="list-style-type: none"><li>For DTMF digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *, #, A, B, C, D</li></ul> <p>This parameter shall only include previously unreported digits.</p>

### 4.2.3 Telephony Tones Generated

The Telephony Tones Generated event indicates that telephony tones have been generated at a device.

Common situations that generate this event include:

- The switching function generates telephony tones (via the Generate Telephony Tones service) for the device of a given connection.

#### Parameters

**Table 204: Telephony Tones Generated Event Parameters**

Parameter Name	Contents	M/C/O	Comments
MonitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
connection	ConnectionID	M	The connection at the device

Parameter Name	Contents	M/C/O	Comments
toneGenerated	Enumerated	C	<p>Specifies the generated tone. Possible values are :</p> <ul style="list-style-type: none"> <li>• beep</li> <li>• billing</li> <li>• busy</li> <li>• carrier</li> <li>• confirmation</li> <li>• dial</li> <li>• faxCNG</li> <li>• hold</li> <li>• howler</li> <li>• intrusion</li> <li>• modemCNG</li> <li>• park</li> <li>• record-warning (indicates call may be being recorded)</li> <li>• reorder</li> <li>• ringback</li> <li>• silence</li> <li>• SIT-VC</li> <li>• SIT-IC</li> <li>• SIT-RO</li> <li>• SIT-NC</li> <li>• SwitchSpecified0 through SwitchSpecified100 – reserved for switch-specified tones.</li> <li>• other</li> <li>• unknown</li> </ul> <p>This parameter shall not be provided when a previously reported tone has stopped being generated.</p>

### Usage Notes

- 1) For RCG's the tone is applied to the RCG's partner, but for stations the tone is applied to the station itself.
- 2) The event is reported only in an active I/O Session for anate devices and digite devices without display and for RCGs in a special routing step ( wait for application = WTAPPL ).
- 3) It is only sent when the device of the specified connection generates telephony tones via the GenerateTelephonyTones service, not for switching function initiated telephony tones.

## 4.3 Logical Device Feature Events

### Logical Device Events Summary

**Table 205: Support of Logical Device Events**

Logical Device Event Supported	Event Description Section
Agent Busy	<a href="#">Section 4.3.1</a>
Agent Logged Off	<a href="#">Section 4.3.2</a>
Agent Logged On	<a href="#">Section 4.3.3</a>
Agent Not Ready	<a href="#">Section 4.3.4</a>
Agent Ready	<a href="#">Section 4.3.5</a>
Agent Working After Call	<a href="#">Section 4.3.6</a>
Call Back	<a href="#">Section 4.3.7</a>
Do Not Disturb	<a href="#">Section 4.3.8</a>
Forwarding	<a href="#">Section 4.3.9</a>
Routeing Mode	<a href="#">Section 4.3.10</a>

### Logical Device Events Descriptions

The entire ECMA CSTA III standard covering logical device events is not reproduced here. Changes, limitations, and additions are described as well as those portions of the specification that are supported.

Additionally to the mandatory parameters the following conditional parameters are mandatory:

- 1) `pendingAgentState` The switching function delays the transition to the `pendingAgentState` until the agent is no longer `Busy` or `WorkingAfterCall`.

### 4.3.1 Agent Busy

The Agent Busy event indicates that an agent has entered the Busy state. In this state an agent is involved with an existing ACD call at a device, even if that call is on hold at the device. It also implies that the agent may be able to accept non-ACD calls. Calls between agents, calls between supervisors and agents and private calls may or may not cause this transition.

An example of when this event is generated is when an agent is connected to an ACD call.

## Parameters

Table 206: Agent Busy Event Parameters

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
agentDevice	SubjectDeviceID	M	Indicates the device at which the agent entered the Agent Busy state.
pendingAgentState	Enumerated	M	<p>Indicates the agent state that the agent will transition to after the agent state is no longer Busy. The complete set of possible values is:</p> <ul style="list-style-type: none"> <li>• Working After Call</li> <li>• Not Ready</li> <li>• Ready</li> <li>• Null</li> </ul> <p>This parameter shall be provided if the switching function is delaying the transition to the pendingAgentState until the agent is no longer Busy, otherwise the parameter is optional.</p>
cause	EventCause	O	Indicates a reason for the event. Always Normal.

## 4.3.2 Agent Logged Off

The Agent Logged Off event indicates that an agent has logged off an ACD device or an ACD group.

Typical examples of when this event may be generated are:

- An agent logs off using the telephone.
- An agent is logged off via the Set Agent State service.
- A supervisor logs off an agent on behalf of the agent.
- A logged on device leaves a switching sub-domain or becomes an invalid device in a switching sub-domain (is deconfigured, for example).

## Parameters

Table 207: Agent Logged Off Event Parameters

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
agentDevice	AgentID	M	The device ID specifies the directory number of the agent device.

## Events

Parameter Name	Contents	M/C/O	Comments
agentID	DeviceID	C	Indicates the agent identifier.
acdGroup	Device number	C	The device number specifies the number of the ACD group from which the agent logged off.
cause	EventCause	O	Indicates a reason for the event. Always Normal.

### 4.3.3 Agent Logged On

The Agent Logged On event indicates that an agent is logged-on at a particular device to an ACD device or ACD group and is ready to contribute to the activities of the ACD device or ACD group. It does not indicate that the agent is ready to accept ACD calls.

Typical examples of when this event may be generated are:

- The agent logged on using the telephone.
- The agent logged on using the Set Agent State service.
- During system start-up, if the agent is configured for logging on automatically.

#### Parameters

**Table 208: Agent Logged On Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
agentDevice	Device ID	M	The device ID specifies the directory number of the agent device.
agentID	DeviceID	C	Indicates the agent identifier.
acdGroup	Device number	C	The device number specifies the number of the ACD group from which the agent logged off.
cause	EventCause	O	Indicates a reason for the event. Always Normal.

### 4.3.4 Agent Not Ready

The Agent Not Ready event indicates that an agent has entered the Agent Not Ready state. In this state an agent is logged-on at a particular device to an ACD device or ACD group but is not prepared to handle calls that the ACD distributes. While in this state an agent may receive calls that are not ACD calls.

Typical examples of when this event may be generated are:

- An agent logs on using the telephone and is placed into the Not Ready agent state.
- An agent invokes the Agent Not Ready feature on the telephone.
- The agent invoked Agent Not Ready by using the Set Agent State service.
- A supervisor invokes the Agent Not Ready feature on behalf of the agent.

#### Parameters

**Table 209: Agent Not Ready Event Parameter**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
agentDevice	Device ID	M	The device ID specifies the directory number of the agent device.
cause	EventCause	O	Indicates a reason for the event. Always Normal.

#### Usage Notes

- 1) Agent Not Ready events are generated for individual agents when a supervisor invokes the Unavailable feature for an agent-group.

### 4.3.5 Agent Ready

The Agent Ready event indicates that an agent has entered the Ready state. In this state, an agent is logged-on at a particular device to an ACD device or ACD group and is prepared to handle ACD calls even though it may be involved with non-ACD calls.

Typical examples of when this event may be generated are:

- An agent auto-work timer expires. (This is a configuration option.)
- An agent invokes the Agent Ready feature on the telephone.
- The agent invoked Agent Ready by using the Set Agent State service.
- A supervisor invokes the Agent Ready feature on behalf of the agent.

#### Parameters

**Table 210: Agent Ready Event Parameter**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
agentDevice	Device ID	M	The device ID specifies the directory number of the agent device.
cause	EventCause	O	Indicates a reason for the event. Always Normal.

### 4.3.6 Agent Working After Call

The Agent Working After Call event indicates that an agent has entered the Working After Call state. In this state an agent is no longer connected to an ACD call but is still occupied with work related to a previous ACD call. In this state, an agent cannot receive ACD calls but may be able to receive non-ACD calls. The agent may be performing administrative duties (e.g., updating a business order form) for a previous call, or may be involved with a non-ACD call.

Typical examples of when this event may be generated are:

- An agent completes an ACD call and goes into the workingAfterCall state. (This is a configuration option.)
- An agent invokes the Working After Call feature on the telephone.
- An agent invoked workingAfterCall state by using the Set Agent State service.

#### Parameters

**Table 211: Agent Work Not Ready Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
agentDevice	Device ID	M	The device ID specifies the directory number of the agent device.
pendingAgentState	Enumerated	M	Indicates the agent state that the agent will transition to after the agent state is no longer WorkingAfterCall. The complete set of possible values is: <ul style="list-style-type: none"> <li>• Not Ready</li> <li>• Ready</li> <li>• Null</li> </ul>
cause	EventCause	O	Indicates a reason for the event. Always Normal.

### 4.3.7 Call Back

The Call Back event indicates that a call back feature has been set or cancelled between two devices.

Typical examples of when this event may be generated are:

- A call back was set or a pending call back was cancelled manually from a phone.
- The computing function, on behalf of a user, set or cancelled a call back.

### Parameters

**Table 212: Call Back Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
originatingDevice	SubjectDeviceID	M	Indicates the DeviceID of the originating device when the call back relationship was established.
targetDevice	SubjectDeviceID	M	Indicates the DeviceID of the target device when the call back relationship was established.
callBackSet	Boolean	M	Indicates whether a call back was set or cancelled. Shall be one of the following: <ul style="list-style-type: none"> <li>FALSE - Call Back was cancelled.</li> <li>TRUE - Call Back was set.</li> </ul>

### Usage Notes

- 1) An application cannot cancel a callback for a specific trunk, so it must cancel all callbacks for this trunk.
- 2) The originatingDevice can be only a digite.

## 4.3.8 Do Not Disturb

The Do Not Disturb event indicates that the do not disturb feature has been changed for a device.

Typical examples of when this event may be generated are:

- The do not disturb feature has been changed on the telephone.
- The computing function, on behalf of a user, has invoked the Set Do Not Disturb service.
- An Attendant Console enters/leaves night mode or unplugs/plugs handset.

### Parameters

**Table 213: Do Not Disturb Event Parameter**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
device	SubjectDeviceID	M	Indicates the device where the do not disturb feature was changed.

## Events

Parameter Name	Contents	M/C/O	Comments
doNotDisturbOn	True  OR  False	M	Specifies the current state. <ul style="list-style-type: none"><li>• FALSE = Do not disturb feature is not enabled.</li><li>• TRUE = Do not disturb feature is enabled.</li></ul>

### 4.3.9 Forwarding

The Forwarding event indicates that the forwarding feature has been changed for a device.

Typical examples of when this event may be generated are:

- The forwarding feature has been changed on the telephone.
- The computing function, on behalf of a user, has invoked the Set Forwarding service.
- A General Attendant enters/leaves night mode

#### Parameters

**Table 214: Forwarding Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
device	SubjectDeviceID	M	Indicates the device where the forwarding feature was changed.
forwardingType	Enumerated	O	Indicates the type of forwarding: <ul style="list-style-type: none"><li>• forwardImmediate</li><li>• forwardImmInt</li><li>• forwardImmExt</li><li>• forwardDND</li><li>• forwardDNDInt</li><li>• forwardDNDExt</li><li>• forwardNoAns</li><li>• forwardNoAnsInt</li><li>• forwardNoAnsExt</li><li>• forwardBusy</li><li>• forwardBusyInt</li><li>• forwardBusyExt</li></ul> It is only provided when call forwarding is being activated.
forwardStatus	Boolean	M	Indicates the status of the forwarding type. The complete set of possible values is: <ul style="list-style-type: none"><li>• FALSE - the forwarding type is deactivated</li><li>• TRUE - the forwarding type is active</li></ul>

Parameter Name	Contents	M/C/O	Comments
forwardTo	DeviceID	O	Specifies the destination to which calls are forwarded. It is only provided when call forwarding is being activated.
forwardDefault	Enumerated	O	The switching function uses this parameter differently than it is specified in the ECMA 269 standard. The translation from the standard value to the switching function interpretation: <ul style="list-style-type: none"> <li>defaultForwardingType= fixedIM forwarding</li> <li>defaultForwardingTypeAndForwardingDN= fixedUS forwarding</li> <li>not present= variable forwarding</li> </ul>
privateData <sup>23</sup>	CSTAPrivateData	O	Specifies non-standardized information. OpenScape 4000: <ul style="list-style-type: none"> <li>staticONDAActive</li> <li>staticONDDN</li> </ul> (see <a href="#">Chapter 9, "Appendix C - Private Data"</a> )

### Usage Notes

- 1) The terms internal and external (as in "forward immediate internal on" or "forward immediate external on") refer to the type of call which was forwarded if the type was set. For example, internal calls will be forwarded if the station has "forward immediate internal on."
- 2) No event will be generated if the call forwarding system destination or forwarding type is not valid in the OpenScape 4000 configuration.
- 3) No event will be generated if the call forwarding feature has been changed for a device with AMOs.
- 4) Only a single standardized forwarding type is provided for each forwarding event. If multiple forwarding types are invoked for the same device, multiple forwarding events are generated.
- 5) A Forwarding event is also issued, when a :
  - General Attendant enters night mode (last Attendant Console enters night mode)
  - General Attendant leaves night mode (firstAttendant Console leaves night mode).

A GA entering/leaving night mode receives a forwarding event as follows:

Parameter Name	Contents	Comments
device	Device ID	The GA which enters/leaves night mode
forwardingType	Enumerated	forwarding type when GA enters/ leaves nm

<sup>23</sup> Supported starting with OpenScape 4000 V8R1

## Events

Parameter Name	Contents	Comments
forwardStatus	Boolean	<ul style="list-style-type: none"><li>FALSE -GA leaves night mode</li><li>TRUE - GA enters night mode</li></ul>
forwardTo	DeviceID	The GA night station (only when entering night mode)
forwardDefault	Enumerated	not present

### 4.3.10 Routeing Mode

The Routing Mode event indicates that the routing mode has been changed for a device.

Typical examples of when this event may be generated are :

- The computing function, on behalf of a user, has invoked the Set Routing Mode service.
- The switching function has changed the routing mode for a device.

#### Parameters

**Table 215: Routing Mode Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
device	SubjectDeviceID	M	Indicates the device where the routeing mode was changed.
routeingMode	Boolean	M	Indicates the routeing mode of the device. The complete set of possible values is: <ul style="list-style-type: none"><li>TRUE - the device will request routeing instructions (when a call arrives at the device, for example)</li><li>FALSE - the device will not request routeing instructions.</li></ul>

#### Usage Notes

- 1) Device should be registered by invoking routing service RouteRegister (and monitored by invoking monitoring service StartMonitor).

## 4.4 Physical Device Feature Events

### Physical Device Feature Events Summary

**Table 216: Support of Physical Device Feature Events**

Physical Device Feature Events	Event Description Section
Message Waiting	<a href="#">Section 4.4.1</a>

### 4.4.1 Message Waiting

The Message Waiting event indicates that the message waiting status has been changed for a device.

This event may be generated in any one of the following ways:

- The message waiting feature has been changed on the telephone.
- A computing function, on behalf of a user, has invoked the Set Message Waiting service.

#### Parameters

**Table 217: Message Waiting Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
targetDevice	SubjectDeviceID	M	Specifies the device where the message waiting feature has changed.
messageWaitingOn	Boolean	M	Specifies the setting of the message waiting feature. The complete set of possible values is: <ul style="list-style-type: none"> <li>• FALSE - Message waiting off.</li> <li>• TRUE - Message waiting on.</li> </ul>

#### Usage notes

- 1) The "application message waiting" uses the existing message waiting lamp on digital devices to indicate if there is any message waiting. This lamp is also used for features like PhoneMail, CallBack, External Phone Mail, etc. Therefore, the message waiting lamp may be on, even if no message waiting event has been received.
- 2) This event is only reported for digite devices.

## 4.5 Maintenance Events

### Maintenance Events Summary

**Table 218: Support of Maintenance Events**

Maintenance Events	Event Description Section
Back In Service	<a href="#">Section 4.5.1</a>
Out Of Service	<a href="#">Section 4.5.2</a>

### 4.5.1 Back In Service

The Back In Service event indicates that the device has been returned to service and is operating normally.

#### Parameters

**Table 219: Back in Service Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.
device	SubjectDeviceID	M	Indicates the device that is back in service.
cause	EventCause	O	Specifies a reason for the event. Always Normal.

#### Usage Notes

- 1) The Back In Service event does not imply that the capabilities of the device out of service have changed.

### 4.5.2 Out Of Service

The Out Of Service event indicates that the device has entered a maintenance state (i.e., has been taken out of service) and can no longer accept calls and some categories of CSTA service requests (Call Control services, for example).

#### Parameters

**Table 220: Out of Service Event Parameters**

Parameter Name	Contents	M/C/O	Comments
monitorCrossRefID	MonitorCrossRefID	M	Associates the event to an established monitor.

Parameter Name	Contents	M/C/O	Comments
device	SubjectDeviceID	M	Indicates the device that has been taken out of service.
cause	EventCause	O	Specifies a reason for the event, always event cause 'Normal'

#### Usage Notes

- 1) When a device goes out of service, existing monitors are not removed, existing MonitorCrossRefs remain valid.
- 2) Snapshot services will result in a negative acknowledgement (error code: Device Out Of Service) if attempted on a device out of service.

## 4.6 Vendor Specific Extensions Events

### Vendor Specific Extensions Events Summary

**Table 221: Support of Vendor Specific Extensions Events**

Vendor Specific Extensions Events	Event Description Section
Mobile User Status	<a href="#">Section 4.6.1</a>

### 4.6.1 Mobile User Status ( Private Event )

The mobile user feature allows a user to identify himself (e.g. per manually entering a PIN, by inserting a chipcard or via Smart Card and card reader) at another phone, so that e.g. he can use his authorizations for outgoing calls.

#### Parameters

**Table 222: Mobile User Status Event Parameters**

Parameter Name	Contents	M/C/O	Comments
physicalDevice	DeviceID	M	Specifies the device where the Mobile User feature has changed
mobileUserStatusOn	Boolean	M	Specifies the setting of the Mobile User feature. The complete set of possible values is: <ul style="list-style-type: none"> <li>• FALSE: Mobile User feature deactivated</li> <li>• TRUE: Mobile User feature activated</li> </ul>
mobileUserDirectoryNumber	Characters (64)	C	Specifies the Mobile User extension. Mandatory if the Mobile User feature is activated, Not Present if it is deactivated.

## Part 2: Call Scenarios

### 5 Call Scenarios

#### 5.1 Scope

This chapter contains the most common call scenarios of the OpenScape 4000 in OpenScape 4000 CSTA V1. A call scenario is the series of steps that make up a telephony activity. A call scenario describes the actions occurring among all parties involved in a call, in sequence.

Each scenario includes a textual description and an illustration. Illustrations use the same key as described within ECMA-269. For each scenario, message sequences are listed for all device type monitored devices. All devices have device type monitors set with no events masked. The columns in each scenario represent the following:

- The Activity column includes a brief description of the telephony activity. The activity can either be initiated by a service invocation or manually.
- The Monitored Device(s) columns list events generated for the specified device type monitor or a service request and service response.
- The Comments column describes additional information on the activity.

The monitorCrossRefID parameter in events is not shown.

DeviceIDs are illustrated by Dn and ConnectionIDs in the from DnCn.

**Table 223: Monitorable Devices**

DeviceID	Description
Dn	Digital Telepone unless otherwise stated. ( Attendant Console , Analog Telephone )
Rn	Route Control Group ( RCG )
Gn	General Attendant ( GA )
Hn	Hunt Group ( HG )
Nn	Network Interface Device ( NID )
An	Attendant Console

All Device IDs are within the same switching sub-domain unless otherwise indicated or stated. Any exception comments are made in the final column Comments.

We followed ECMA TR/82 as much as possible to make it easier for the application developer to compare the implementation of OpenScape 4000 with the ECMA directive. The document concentrates on the chosen CSTA implementation options and the differences from the ECMA directive.

#### Overview of the main sections:

The first 14 sections have the following structure:

[Section 5.4, "Call Origination Scenarios"](#)

[Section 5.5, "Answering Call Scenarios"](#)

[Section 5.6, "Connection Termination Scenarios"](#)

[Section 5.7, "External outgoing calls"](#)

[Section 5.8, "External incoming calls"](#)

[Section 5.9, "Forwarding Call Scenarios"](#)

[Section 5.10, "Multiple Forwarding Scenarios"](#)

[Section 5.11, "Call Movement Scenarios"](#)

[Section 5.12, "Hold/Retrieving Scenarios"](#)

[Section 5.13, "Consultation Call Scenarios"](#)

[Section 5.14, "Transfer Call Scenarios"](#)

[Section 5.15, "Conference Call Scenarios"](#)

[Section 5.16, "Call Completion Scenarios"](#)

[Section 5.17, "Distribution Call Scenarios"](#)

The ECMA-269 standard gives relative freedom to the implementation of recalls. The below section describes the interpretation of the OpenScape 4000.

[Section 5.18, "Recall Scenarios"](#)

The last section describes OpenScape 4000 features, that are either not described by ECMA 269 or they are not CSTA III standard compliant.

[Section 5.19, "OpenScape 4000 Specific Features"](#)

## 5.2 References

ECMA-269 Services for Computer Supported Telecommunications Applications (CSTA) Phase III, 4th edition (Dec 2011)

ECMA-285 Protocol for Computer Supported Telecommunications Applications (CSTA) Phase III, 2nd edition (Dec 2011)

ECMA TR/72 Glossary of definitions and terminology for Computer Supported Telecommunications Applications (CSTA) Phase III, 3rd edition (June 2009)

ECMA TR/82 Scenarios for Computer Supported Telecommunication Applications (CSTA) Phase III, (June 2009)

## 5.3 Definitions and Abbreviations

The definitions and abbreviations used in this Technical Report are defined in ECMA TR/72.

## 5.4 Call Origination Scenarios

## 5.4.1 Manually dialed call

This scenario illustrates a call originated through manual device activity.

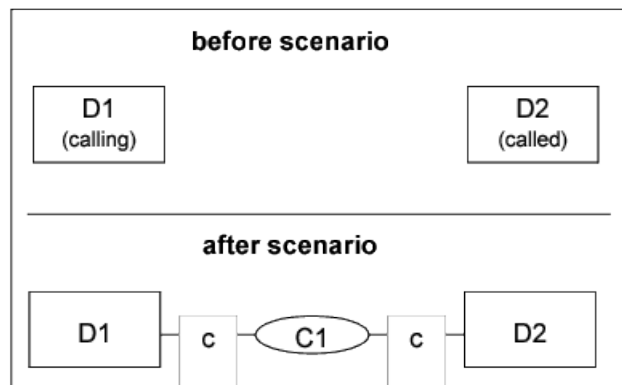


Figure 11: Manually dialed call

Table 224: Manually dialed call

Activity	Monitored Device D1		Monitored Device D2	Comments
1) D1 goes off-hook.	Service Initiated			
	• initiatedConnection	• D1C1		
	• initiatingDevice	• D1		
	• localConnectionInfo	• initiated		
	• cause	• normal		
	• servicesPermitted	• ClearConn, DialDgt		
1) D1 completes dialling D2.	Digits Dialed			Num-ber of D2 is: 1234
	• diallingConnection	• D1C1		
	• diallingDevice	• D1		
	• diallingSequence	• "1234"		
	• localConnectionInfo	• initiated		
	• cause	• normal		
	• servicesPermitted	• none		
	Originated			
	• originatedConnection	• D1C1		
	• callingDevice	• D1		
	• calledDevice	• D2		
	• localConnectionInfo	• connected		

Activity	Monitored Device D1		Monitored Device D2		Comments
	<ul style="list-style-type: none"> <li>cause</li> </ul>	<ul style="list-style-type: none"> <li>normal</li> </ul>			
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	<ul style="list-style-type: none"> <li>ClearConn</li> </ul>			
1) D2 starts ringing.	Delivered		Delivered		
	<ul style="list-style-type: none"> <li>connection</li> </ul>	D2C1	<ul style="list-style-type: none"> <li>connection</li> </ul>	D2C1	
	<ul style="list-style-type: none"> <li>alertingDevice</li> </ul>	D2	<ul style="list-style-type: none"> <li>alertingDevice</li> </ul>	D2	
	<ul style="list-style-type: none"> <li>callingDevice</li> </ul>	D1	<ul style="list-style-type: none"> <li>callingDevice</li> </ul>	D1	
	<ul style="list-style-type: none"> <li>calledDevice</li> </ul>	D2	<ul style="list-style-type: none"> <li>calledDevice</li> </ul>	D2	
	<ul style="list-style-type: none"> <li>lastRedirectionDevice</li> </ul>	NS	<ul style="list-style-type: none"> <li>lastRedirectionDevice</li> </ul>	NS	
	<ul style="list-style-type: none"> <li>localConnectionInfo</li> </ul>	connected	<ul style="list-style-type: none"> <li>localConnectionInfo</li> </ul>	alert	
	<ul style="list-style-type: none"> <li>cause</li> </ul>	normal	<ul style="list-style-type: none"> <li>cause</li> </ul>	normal	
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	CallBack, ClearConn, SendUserInfo	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	AnswerCall, ClearConn, Deflect, SendUserInfo	
1) D2 answers the call.	Established		Established		
	<ul style="list-style-type: none"> <li>establishedConnection</li> </ul>	D2C1	<ul style="list-style-type: none"> <li>establishedConnection</li> </ul>	D2C1	
	<ul style="list-style-type: none"> <li>answeringDevice</li> </ul>	D2	<ul style="list-style-type: none"> <li>answeringDevice</li> </ul>	D2	
	<ul style="list-style-type: none"> <li>callingDevice</li> </ul>	D1	<ul style="list-style-type: none"> <li>callingDevice</li> </ul>	D1	
	<ul style="list-style-type: none"> <li>calledDevice</li> </ul>	D2	<ul style="list-style-type: none"> <li>calledDevice</li> </ul>	D2	
	<ul style="list-style-type: none"> <li>lastRedirectionDevice</li> </ul>	NS	<ul style="list-style-type: none"> <li>lastRedirectionDevice</li> </ul>	NS	
	<ul style="list-style-type: none"> <li>localConnectionInfo</li> </ul>	connected	<ul style="list-style-type: none"> <li>localConnectionInfo</li> </ul>	connected	
	<ul style="list-style-type: none"> <li>cause</li> </ul>	normal	<ul style="list-style-type: none"> <li>cause</li> </ul>	normal	
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	

**Remark:**

- The complete dialled digits sequence is provided in the Originated event.
- Digit Dialed events are never generated for manual activity.
- A more specific event cause in the Service Initiated event cannot be provided.

### 5.4.2 Manually dialled call - Called party is busy

This scenario illustrates a call scenario where a call is made to a busy party.

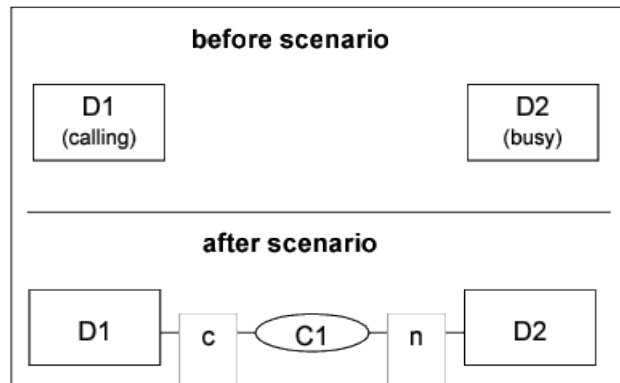


Figure 12: Manually dialled call - Called party is busy

Table 225: Unsuccessful basic call - called party is busy

Activity	Monitored Device D1		Monitored Device D2	Comments
1) D1 goes off-hook.	Service Initiated			
	• initiatedConnection	D1C1		
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn, DialDgt		
1) D1 completes dialling D2.	Digits Dialed			Number of D2 is: 1234
	• diallingConnection	D1C1		
	• diallingDevice	D1		
	• diallingSequence	"1234"		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	none		
	Originated			
	• originatedConnection	D1C1		
	• callingDevice	D1		
	• calledDevice	D2		
	• lastRedirectionDevice	NS		

Activity	Monitored Device D1		Monitored Device D2		Comments
	• localConnectionInfo	connected			
	• cause	normal			
	• servicesPermitted	ClearConn			
1) D2 is busy. The call can not be completed. D1 hears busy tone.	Failed		Failed		This illustrates connection failures that report the Failed event for all devices involved with the call and that will provide a complete connectionID for the failed connection.
	• failedConnection	D2C1	• failedConnection	D2C1	
	• failingDevice	D2	• failingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	fail	
	• cause	busy	• cause	busy	
	• servicesPermitted	ClearConn	• servicesPermitted	none	
1) The busy connection is cleared immediately	Connection Cleared		Connection Cleared		
	• droppedConnection	D2C1	• droppedConnection	D2C1	
	• releasingDevice	D2	• releasingDevice	D2	
	• localConnectionInfo	connected	• localConnectionInfo	null	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	ClearConn	• servicesPermitted	none	

**Remark:**

- The protocol converter of the switching function will immediately send a Connection Cleared event after a connection goes into the busy failed state. This does not necessarily mean, that the connection physically goes to idle.
- The complete dialled digits sequence is provided in the Originated event.
- Digit Dialed events are never generated for manual activity.

### 5.4.3 Manually dialled call - Called party is Out Of Service (OOS)

This scenario illustrates a call scenario where a call is made to an party, which is out of service.

## Call Scenarios

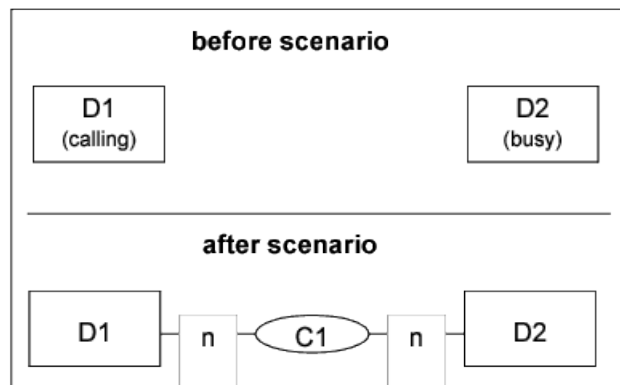


Figure 13: Manually dialed call - Called party is Out of Service (OOS)

Table 226: Unsuccessful basic call - called party is out of service

Activity	Monitored Device D1		Monitored Device D2	Comments	
1) D1 goes offhook	Service Initiated				
	• initiatedConnection	D1C1			
	• initiatingDevice	D1			
	• localConnectionInfo	initiated			
	• cause	normal			
	• servicesPermitted	ClearConn, DialDgt			
1) D1 completes dialling D2's number	Digits Dialed			Number of D2 is: 1234	
	• diallingConnection	D1C1			
	• diallingDevice	D1			
	• diallingSequence	"1234"			
	• localConnectionInfo	initiated			
	• cause	normal			
	• servicesPermitted	none			
	Originated				
	• originatedConnection	D1C1			
	• callingDevice	D1			
	• calledDevice	D2			
	• lastRedirectionDevice				
	• localConnectinfo	connected			
	• cause	normal			

Activity	Monitored Device D1		Monitored Device D2		Comments
	<div>• servicesPermitted</div>	ClearConn			
1) Called party D2 is Out of Service	Failed		Failed		
	<div>• failedConnection</div>	D2C1	<div>• failedConnection</div>	D2C1	
	<div>• failingDevice</div>	D2	<div>• failingDevice</div>	D2	
	<div>• callingDevice</div>	D1	<div>• callingDevice</div>	D1	
	<div>• calledDevice</div>	D2	<div>• calledDevice</div>	D2	
	<div>• lastRedirectionDevice</div>	NS	<div>• lastRedirection-Device</div>	NS	
	<div>• localConnectionInfo</div>	connected	<div>• localConnection-Info</div>	fail	
	<div>• cause</div>	destinationOutOfOrder	<div>• cause</div>	destinationOutOfOrder	
	<div>• servicesPermitted</div>	ClearConn	<div>• services-Permitted</div>	none	
	Connection Cleared		Connection Cleared		
	<div>• droppedConnection</div>	D2C1	<div>• dopped-Connection</div>	D2C1	
	<div>• releasingDevice</div>	D2	<div>• releasingDevice</div>	D2	
	<div>• localConnectionInfo</div>	connected	<div>• localConnection-Info</div>	null	
	<div>• cause</div>	normalClr	<div>• cause</div>	normalClr	
	<div>• servicesPermitted</div>	ClearConn	<div>• services-Permitted</div>	none	
1) D1 goes onhook	Connection Cleared				
	<div>• droppedConnection</div>	D1C1			
	<div>• releasingDevice</div>	D1			
	<div>• localConnectionInfo</div>	null			
	<div>• cause</div>	normalClr			
	<div>• servicesPermitted</div>	none			

**Remark:**

none

**5.4.4 Manually dialled call - Dialed number is invalid**

This scenario illustrates a manually dialled call to an invalid destination. Device D2 is actually an invalid number.

## Call Scenarios

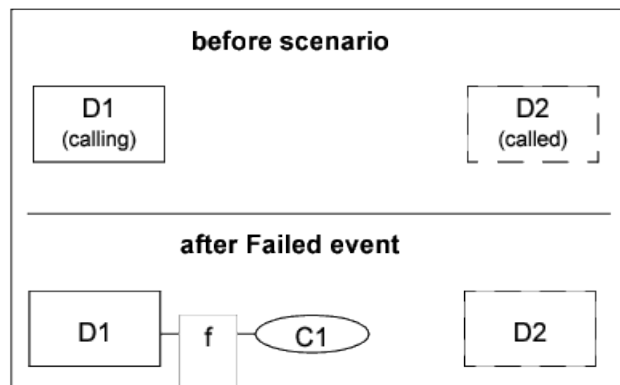


Figure 14: Manually dialed call - Dialed number is invalid

Table 227: Dialed number invalid

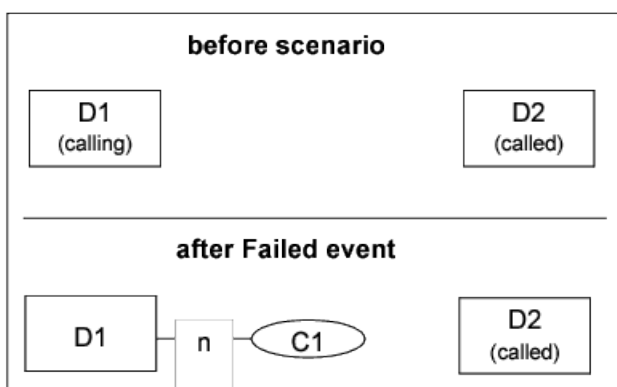
Activity	Monitored Device D1		Monitored Device D2	Comments
1) Device D1 goes offhook.	Service Initiated			
	• initiatedConnection	D1C1		
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn, DialDigits		
1) Since D1 dialled an invalid number, it becomes blocked.	Digits Dialed			D1 dials 9999- it is an invalid number
	• diallingConnection	D1C1		
	• diallingDevice	D1		
	• diallingSequence	"9999"		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	none		
	Failed			The switching function does not provide the Originated event in this case.  D1C1 immediately becomes failed.
	• failedConnection	D1C1		
	• failingDevice	D1		
	• callingDevice	D1		
	• calledDevice	NK		
	• lastRedirectionDevice	NS		
	• localConnectionInfo	fail		
	• cause	normal		

Activity	Monitored Device D1		Monitored Device D2	Comments
1) Device D1 clears its failed call.	• servicesPermitted	ClearConn		
	Connection Cleared			
	• droppedConnection	D1C1		
	• releasingDevice	D1		
	• localConnectionInfo	null		
	• cause	normalClr		
	• servicesPermitted	none		

Remark: None

### 5.4.5 Manually dialled call - Incomplete dialling sequence, calling party goes onhook

This scenario illustrates a manually dialled incomplete call. A starts a call to B. Before dialling the whole number A goes onhook.



**Figure 15: Manually dialled call - Incomplete dialling sequence, calling party goes onhook**

**Table 228: Incomplete dialling sequence, calling party goes onhook**

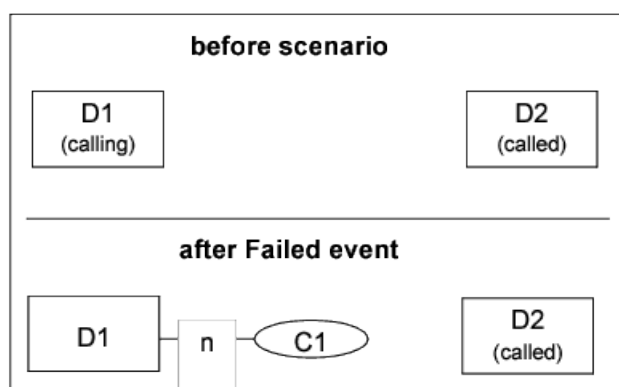
Activity	Monitored Device D1		Monitored Device D2	Comments
1) D1 goes offhook	Service Initiated			
	• initiatedConnection	D1C1		
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn, DialDgt		

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2	Comments
1) D1 does not complete dialling D2's number ("1234")	Digits Dialed			Number of D2 is: 1234
	• diallingConnection	D1C1		
	• diallingDevice	D1		
	• diallingSequence	"123"		
	• localConnectionInfo	initiated		
	• cause	normal		
1) D1 goes onhook	Connection Cleared			
	• droppedConnection	D1C1		
	• releasingDevice	D1		
	• localConnectionInfo	null		
	• cause	normal		
	• servicesPermitted	none		

### 5.4.6 Manually dialled call - Incomplete dialling sequence, dialling has timed out

A starts a call to B. Dialling has timed out.



**Figure 16: Manually dialled call - Incomplete dialling sequence, dialling has timed out**

**Table 229: Manually dialled call - Incomplete dialling sequence, dialling has timed out**

Activity	Monitored Device D1		Monitored Device D2	Comments
1) D1 goes offhook	Service Initiated			
	• initiatedConnection	D1C1		

Activity	Monitored Device D1		Monitored Device D2	Comments
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn, DialDgt		
1) D1 does not complete dialling D2's number ("1234")	Digits Dialed			Number of D2 is: 1234
	• diallingConnection	D1C1		
	• diallingDevice	D1		
	• diallingSequence	"123"		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	none		
1) dialling has timed out	Failed			
	• failedConnection	D1C1		
	• failingDevice	D1		
	• callingDevice	D1		
	• calledDevice	NK		
	• lastRedirectionDevice	NS		
	• localConnectionInfo	fail		
	• cause	normal		
	• servicesPermitted	ClearConn		
	Connection Cleared			
	• droppedConnection	D1C1		
	• releasingDevice	D1		
	• localConnectionInfo	null		
	• cause	normal		
	• servicesPermitted	none		

### 5.4.7 Make Call service

This scenario illustrates a successful Make Call from device D1 to device D2. In this scenario both devices are available and valid, device D1 is permitted to make the call and the call is answered by device D2.

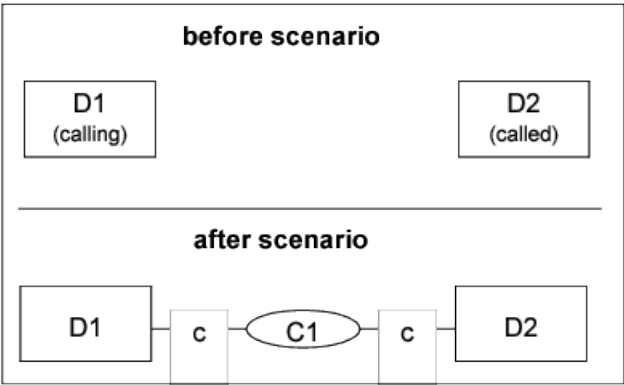


Figure 17: Make call service

Table 230: Make call service

Activity	Monitored Device D1		Monitored Device D2	Comments
1) Make call is invoked on D1.	Make Call Request			The Make Call service specifies that device D1 should be prompted to go off-hook.
	• callingDeviceID	D1		
	• calledDirectoryNumber deviceID	D2		
	• autoAnswer	prompt		
1) Acknowledgement	Make Call Response			
	• connectionID	D1C1		
1) Indication that the service initiated from this device	Service Initiated			The MakeCall cause indicates that the device D1 is being prompted (via ringing, for example) to go off-hook.
	• initiatedConnection	D1C1		
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	MakeCall		
	• servicesPermitted	Answer, ClearConn, DialDgt, SendUserInfo		
Scenario proceeds as shown in <a href="#">Section 7.4.1, "Manually dialled call"</a>				

Remark:  
None

5.4.8 Multi Stage dialling

This scenario illustrates the use of the Dial Digits service to complete dialling a call that was established via a Make Call service. In this scenario both devices are available and valid, device D1 is permitted to make the call and the call is answered by device D2 ( 3160 ).

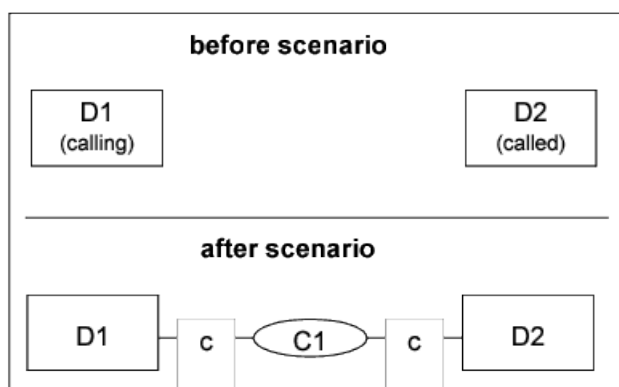


Figure 18: Multi stage dialling

Table 231: Multi Stage Dialling

Activity	Monitored Device D1		Monitored Device D2	Comments
1) Make call is invoked on D1.	Make Call Request			The Make Call service a partial dialling string that includes the first part of the number of D2 ("31") and the partial dialling indicator (":").
	• callingDeviceID	D1		
	• calledDirectoryNumber deviceID	"31;"		
1) Acknowledgement	Make Call Response			
	• connectionID	D1C1		
1) D1 goes off-hook. The event indicates that the service initiated from this device	Service Initiated			The MakeCall cause indicates that the device D1 is being prompted (via ringing, for example) to go off-hook.
	• initiatedConnection	D1C1		
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	Make Call		
1) The event indicates that partial dialling is used.	Digits dialled			A ":" character indicates that there is an incomplete dialling string.
	• diallingConnection	D1C1		
	• diallingDevice	D1		
	• diallingSequence	"31;"		
	• localConnectionInfo	initiated		

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2	Comments
	• cause	normal		
	• servicesPermitted	none		
1) Dial Digits is invoked on D1.	Dial Digits Request			A ";" is not provided in the dialling string since there are no more digits to be dialled.
	• connectionToBeDialled	D1C1		
	• diallingSequence	"60"		
1) Acknowledgement	Dial Digits Response			
1) The dialling sequence is completed and D1 is connected in the call.	Originated			<p>Note, that the last Digit Dialed event is missing.</p> <p>The switching function provides only the Originated event as an indicator of the finished dialling sequence.</p> <p>D2 is the called device. It contains the digits "3160" in this scenario.</p>
	• originatedConnection	D1C1		
	• callingDevice	D1		
	• calledDevice	D2		
	• lastRedirectionDevice	NS		
	• localConnectionInfo	connected		
	• cause	normal		
	• servicesPermitted	ClearConn, SendUserInfo		
Scenario proceeds as shown in <a href="#">Section 7.4.1, "Manually dialled call"</a>				

### Remark:

None

## 5.4.9 Call offered to an application

This scenario illustrates a call offered to an application. Building up the call till Originated can happen either manually or via Make Call Service. Please refer to the previous tables. Offer is supported on monitored digital devices and requires special configuration. Providing Offered event to a calling device is optional.

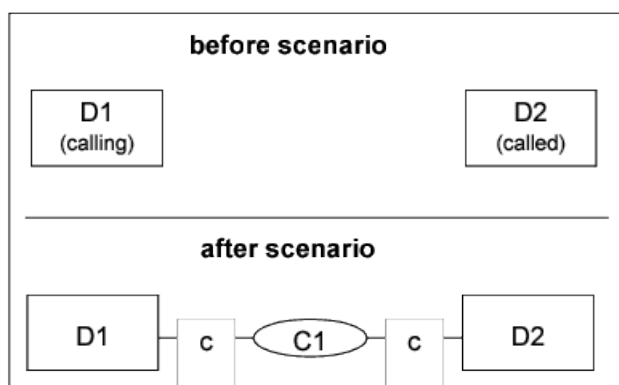


Figure 19: Call offered to an application

Table 232: Call offered

Activity	Monitored Device D1		Monitored Device D2	Comments
1) D1 goes offhook	Service Initiated			
	• initiatedConnection	D1C1		
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn, DialDgt		
1) D1 completes dialling D2	Digits Dialed			Number of D2 is: 1234
	• diallingConnection	D1C1		
	• diallingDevice	D1		
	• diallingSequence	"1234"		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	none		
	Originated			
	• originatedConnection	D1C1		
	• callingDevice	D1		
	• calledDevice	D2		
	• localConnectionInfo	connected		
	• cause	normal		
	• servicesPermitted	ClearConn		

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Comments
1) Call is offered to D2	Offered		Offered		Providing Offered for calling side is optional
	• offeredConnection	D2C1	• offered-Connection	D2C1	
	• offeredDevice	D2	• offeredDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirection-Device	NS	
	• localConnectionInfo	connectd	• localConnectionInfo	alerting	
	• cause	normal	• cause	normal	
	• servicesPermitted	ClearConn	• services-Permitted	AcceptCall, ClearConn, Deflect	
1) Application accepts the call			Accept Cal IRequest		
			• callToBe-Accepted	D2C1	
1) Acknowledged			Accept Call Response		
1) D2 starts ringing.	Delivered		Delivered		
	• connection	D2C1	• connection	D2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirection-Device	NS	• lastRedirectionDevice	NS	
	• localConnection-Info	connected	• localConnection-Info	alert	
	• cause	normal	• cause	normal	
	• services-Permitted	CallBack, ClearConn, SendUserInfo	• services-Permitted	AnswerCall, ClearConn, Deflect, SendUserInfo	
1) D2 answers the call.	Established		Established		
	• establishedConnection	D2C1	• established-Connection	D2C1	
	• answeringDevice	D2	• answeringDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirection-Device	NS	• lastRedirection-Device	NS	

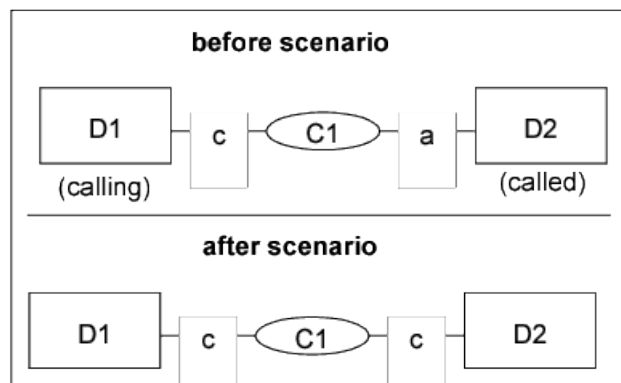
Activity	Monitored Device D1		Monitored Device D2		Comments
	• localConnection-Info	connected	• localConnection-Info	connected	
	• cause	normal	• cause	normal	
	• services-Permitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• services-Permitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	

**NOTICE:** Providing Offered event also on the calling side is configurable.

## 5.5 Answering Call Scenarios

### 5.5.1 Successful answer call

This clause illustrates how calls are answered by CSTA services.



**Figure 20: Answer call successful**

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before service" state.

**Table 233: Answer Callservice**

Activity	Monitored Device D1		Monitored Device D2		Comments
1) Answer call service is invoked on D2.			Answer Call Request		
			• call to answer call ID	D2C1	
1) Acknowledgement			Answer Call Response		
1) D2 answers the call.	Established		Established		
	• establishedConnection	D2C1	• establishedConnection	D2C1	

Activity	Monitored Device D1		Monitored Device D2		Comments
	• answeringDevice	D2	• answeringDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	normal	• cause	normal	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	

**Remark:**  
The manual case is similar to the described event flow.

5.6 Connection Termination Scenarios

5.6.1 Device disconnects from a call by on-hook

5.6.1.1 Non-SIP Device disconnects from a call by on-hook

In this scenario device D1 is manually put on-hook to release itself from the call. The remaining device goes blocked, until the device goes on-hook.

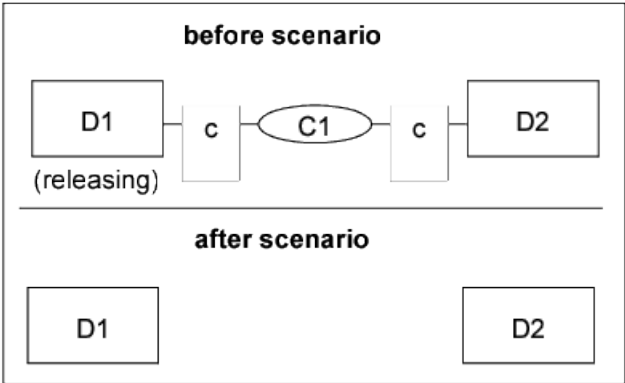


Figure 21: Connection termination scenarios

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before scenario" state.

**Table 234: Non-SIP Device disconnects from a call by on-hook**

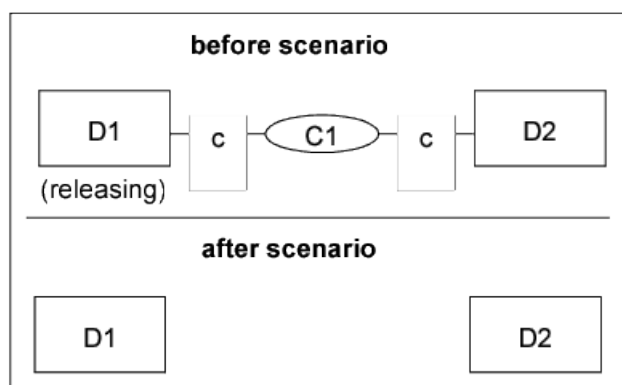
Activity	Monitored Device D1		Monitored Device D2		Comments
1) D1 goes on-hook.	Connection Cleared		Connection Cleared		
	• droppedConnection	D1C1	• droppedConnection	D1C1	
	• releasingDevice	D1	• releasingDevice	D1	
	• localConnectionInfo	null	• localConnectionInfo	connected	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	none	• servicesPermitted	ClearConn	
1) As a result of the "far end disconnect", the remaining connection D2C1 goes blocked.			Failed		
			• failedConnection	D2C1	
			• failingDevice	D2	
			• callingDevice	D1	
			• calledDevice	D2	
			• lastRedirectionDevice	NS	
			• localConnectionInfo	fail	
			• cause	blocked	
			• servicesPermitted	ClearConn	
1) The remaining device goes onhook.			Connection Cleared		
			• droppedConnection	D2C1	
			• releasingDevice	D2	
			• localConnectionInfo	null	
			• cause	normalClr	
			• servicesPermitted	none	

**Remark:**

None

**5.6.1.2 SIP Device disconnects from a call by on-hook**

In this scenario device D1 (SIP) is manually put on-hook to release itself from the call. Both devices go blocked, until the devices go on-hook.



**Figure 22: SIP device disconnects from a call by on-hook**

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before scenario" state.

**Table 235: SIP Device disconnects from a call by on-hook**

Activity	Monitored Device D1		Monitored Device D2		Comments
1) D1 goes on-hook.	Failed		Connection Cleared		
	• failedConnection	D1C1	• droppedConnection	D1C1	
	• failingDevice	D1	• releasingDevice	D1	
	• callingDevice	D1	• localConnectionInfo	connected	
	• calledDevice	D2	• cause	normalClr	
	• lastRedirectionDevice	NS			
	• localConnectionInfo	fail			
	• cause	blocked			
1) As a result of the "far end disconnect", the remaining connection D2C1 goes blocked.	Connection Cleared		Failed		
	• droppedConnection	D1C1	• failedConnection	D2C1	
	• releasingDevice	D1	• failingDevice	D2	
	• localConnectionInfo	null	• callingDevice	D1	
	• cause	normalClr	• calledDevice	D2	
	• servicesPermitted	none	• lastRedirectionDevice	NS	
			• localConnectionInfo	fail	
			• cause	blocked	
			• servicesPermitted	ClearConn	

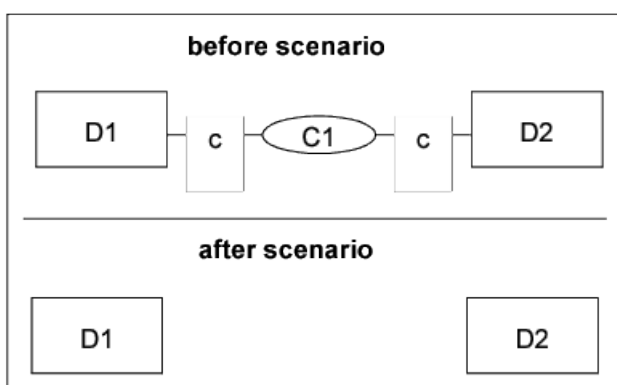
Activity	Monitored Device D1	Monitored Device D2	Comments
1) The remaining device goes onhook.		Connection Cleared	
		• droppedConnection D2C1	
		• releasingDevice D2	
		• localConnectionInfo null	
		• cause normalClr	
		• servicesPermitted none	

**Remark:**

None

### 5.6.2 Device disconnects from a call using the Clear Connection service (remaining device goes blocked)

The Clear Connection service is used to disconnect device D1 from the call. After the service is invoked both devices go into blocked state.



**Figure 23: Device disconnects by using the clear connection service**

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before service" state.

**Table 236: Device disconnects by using the Clear Connection service**

Activity	Monitored Device D1	Monitored Device D2	Comments
1) A Clear Connection service is invoked.	ClearConnectionRequest		
	• connectionToBeCleared D1C1		
1) Acknowledgement.	ClearConnectionResult Response		

## Call Scenarios

### External outgoing calls

Activity	Monitored Device D1		Monitored Device D2		Comments
1) D1C1 goes blocked.	Failed		Failed		This illustrates connection failures that report the Failed event for all devices involved with the call and that will provide a complete connectionID for the failed connection.
	• failedConnection	D1C1	• failedConnection	D1C1	
	• failingDevice	D1	• failingDevice	D1	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	fail	• localConnectionInfo	fail	
	• cause	blocked	• cause	blocked	
	• servicesPermitted	ClearConn	• servicesPermitted	ClearConn	
Scenario proceeds as shown in <a href="#">Section 5.6.1, "Device disconnects from a call by on-hook"</a>					

#### Remark:

The connection, on which the Connection Cleared service was initiated, first goes to failed state. This behaviour is different from the related scenario of ECMA TR/82.

## 5.7 External outgoing calls

Devices outside the CSTA sub-domain can not be directly monitored, network interface devices (NID) (e.g., trunk interface), act as proxies for those devices. Depending upon the type of signalling supported by the network, there may be a reduced level of event reporting after a Network Reached event and possibly no additional device feedback except connection clearing for trunks without Answer Supervision.

For external outgoing calls the associatedCalledDevice is a mandatory parameter. It specifies the Network Interface Device associated with the called device.

#### Remark:

If the IP Direct Access ( IPDA ) feature is used with broken IP connection, as the Survivability Path consists of normal network interface devices, the call between the Access Point and the central OpenScape switch will be reported as an external call.

### 5.7.1 Manual call to a device outside the CSTA subdomain

This scenario illustrates a manual external outgoing call.

Since device D2 is located outside the CSTA sub-domain, it can not be directly monitored through this CSTA interface and therefore no events will be seen for that device. However, device N2, which is a network interface device (NID) (e.g., trunk interface), acts as a proxy for device D2.

This scenario describes the behaviour of a network interface device with network information .

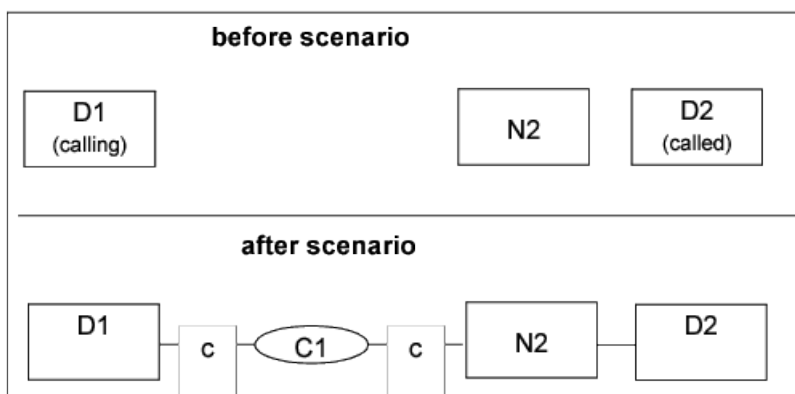


Figure 24: External outgoing call

Table 237: External outgoing call

Activity	Monitored Device D1		Monitored Device N2	Comments
1) D1 goes offhook.	Service Initiated			
	• initiatedConnection	D1C1		
	• initiatingDevice	D1		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn, DialDgt		
1) D1 completes dialling D2's number	Digits Dialed			D2's number is 1234
	• diallingConnection	D1C1		
	• diallingDevice	D1		
	• diallingSequence	"1234"		
	• localConnectionInfo	initiated		
	• cause	normal		
1) D1 is connected to the call.	Originated			
	• originatedConnection	D1C1		
	• callingDevice	D1		
	• calledDevice	D2		
	• lastRedirectionDevice	NS		
	• localConnectionInfo	connected		

## Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Comments
	• cause	normal			
	• servicesPermitted	ClearConn			
1) The call leaves the CSTA subdomain.	Network Reached		Network Reached		
	• outbound connection	N2C1	• outbound connection	N2C1	
	• NID device	N2	• NID device	N2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	normal	• cause	normal	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	Deflect, ClearConn, SendUserInfo	
1) Device D2 is alerted.	Delivered		Delivered		The cause of NetworkSignal indicates that the event is due to activity at the device located outside of the CSTA switching sub-domain (D2), not the NID (N2).
	• connection	N2C1	• connection	N2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	networkSignal	• cause	networkSignal	
	• assocCalled	N2	• assocCalled	N2	
1) Device D2 answers the call.	Established		Established		Network information is received from the network (this depends upon the type of signalling supported by the network).
	• establishedConnection	N2C1	• establishedConnection	N2C1	
	• answeringDevice	D2	• answeringDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	

Activity	Monitored Device D1		Monitored Device N2		Comments
	• cause	networkSignal	• cause	networkSignal	
	• assocCalled	N2	• assocCalled	N2	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• servicesPermitted	ClearConn, SendUserInfo	

**Remark:**

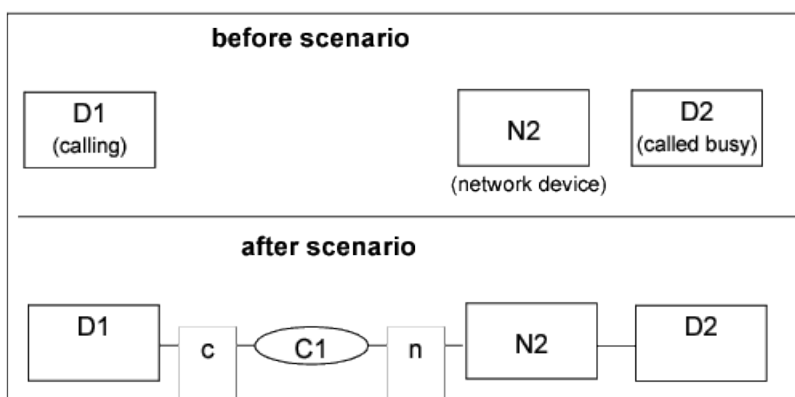
- The switching function provides the same event flow in the service initiated and in the manual case as well. It was modelled after the external outgoing Make Call service of ECMA TR/82.
- When Device D1 is not monitored and Device D2 has a call forward activated, then CA4000 will not be able to provide Device D2 as the originally called device. The originally called device will be the destination of the call forwarding.

## 5.7.2 Manual call to a busy device outside the CSTA subdomain

This scenario illustrates a manual external outgoing call to a busy device.

Since device D2 is located outside the CSTA sub-domain, it can not be directly monitored through this CSTA interface and therefore no events will be seen for that device. However, device N2, which is a network interface device (NID) (e.g., trunk interface), acts as a proxy for device D2.

This scenario describes the behaviour of a network interface device with network information.



**Figure 25: External outgoing call to a busy device**

**Table 238: External outgoing call to a busy device**

Activity	Monitored Device D1	Monitored Device N2	Comments
Steps 1-3 are shown in <a href="#">Section 5.7.1, "Manual call to a device outside the CSTA subdomain", on page 374.</a>			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Comments
4. D2 is busy. The call can not be completed. D1 hears busy tone.	Failed		Failed		
	• failedConnection	N2C1	• failedConnection	N2C1	
	• failingDevice	D2	• failingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	fail	
	• cause	busy	• cause	busy	
	• assocCalled	N2	• assocCalled	N2	
5. The busy connection is cleared immediately.	Connection Cleared		Connection Cleared		
	• droppedConnection	N2C1	• droppedConnection	N2C1	
	• releasingDevice	N2	• releasingDevice	N2	
	• localConnectionInfo	connected	• localConnectionInfo	null	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	ClearConn	• servicesPermitted	none	

### Remark:

The protocol converter of the switching function will immediately send a Connection Cleared event after a connection goes into the busy failed state. This does not necessarily mean, that the connection physically goes to idle.

### Possible event causes:

**Table 239: Event causes**

Event Cause	Description	Associated Features
Busy	The call failed after it encountered a busy or unavailable device.	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
Invalid Number Format	The call failed because the dialled number is incorrect.	Connection Failure

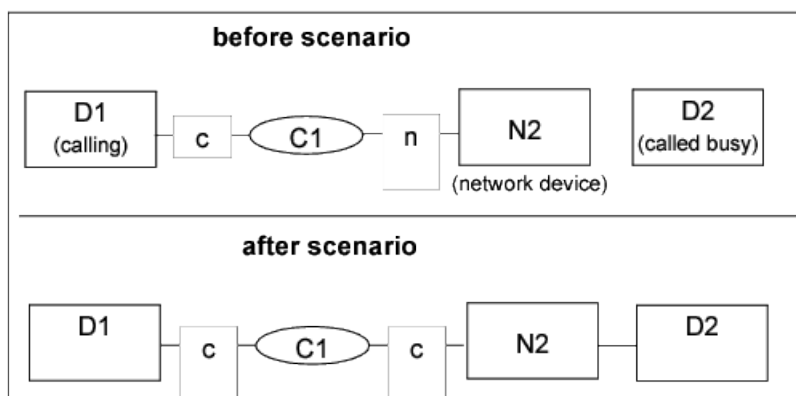
Event Cause	Description	Associated Features
Network Congestion	The call failed because it encountered a congested network.  In some circumstances, this event cause indicates that the user is listening to a special signal tone from a network. The tone may be accompanied by a voiced statement similar to "All circuits are busy..."	Connection Failure
Network Signal	The call failed because it encountered a problem after it left the switching sub-domain.	External Calls
Number Unallocated	The call failed because the called number is not allocated to a subscriber.	Connection Failure

### 5.7.3 External outgoing camp-on

This scenario illustrates an automatic camp-on to a busy device.

Since device D2 is located outside the CSTA sub-domain, it can not be directly monitored through this CSTA interface and therefore no events will be seen for that device. However, device N2, which is a network interface device (NID) (e.g., trunk interface), acts as a proxy for device D2.

This scenario describes the behaviour of a network interface device with network information.



**Figure 26: External outgoing camp-on**

See [Section 5.7.2, "Manual call to a busy device outside the CSTA subdomain"](#) for the event flow to get into the "before scenario" state.

## Call Scenarios

### External incoming calls

**Table 240: External outgoing camp-on**

Activity	Monitored Device D1		Monitored Device N2		Comments
1) The call leaves the CSTA subdomain.	Network Reached		Network Reached		
	• outbound connection	N2C1	• outbound connection	N2C1	
	• NID device	N2	• NID device	N2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	normal	• cause	normal	
1) Device D1 hears ringback, and the call queues to D2 at the other end.	Delivered		Delivered		The switching function provides Delivered event. It means that an application at the outgoing side will not be able to differentiate the camp on call from a basic call.
	• connection	N2C1	• connection	N2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	networkSignal	• cause	networkSignal	
	• assocCalled	N2	• assocCalled	N2	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	Deflect, ClearConn, SendUserInfo	

#### Remark:

None

## 5.8 External incoming calls

Devices outside the CSTA sub-domain can not be directly monitored, network interface devices (NID) (e.g., trunk interface), act as proxies for those devices. Depending upon the type of signalling supported by the network, the following information can be present :

- **networkCallingDevice**: It specifies the ANI number if it is provided by the network. Otherwise it is not present.
- **callingDevice**: It specifies the ANI number if it is provided by the network. Otherwise it is not known ( NK ).

- **networkCalledDevice**: It specifies the DNIS number if it is provided by the network. Otherwise it is not present. This information element will be only present in case of a DNIS trunk.
- **calledDevice**: It specifies an internal format of the DNIS number if it is provided by the network. Otherwise it is not known ( NK ) .
- **assocCalledDevice**: It specifies an internal format of the DNIS number, that is different from the calledDevice, if it is provided by the network. Otherwise it is not present.

For external incoming calls the associatedCallingDevice is a mandatory parameter. It specifies the Network Interface Device associated with the calling device if the call is incoming.

**Remark:**

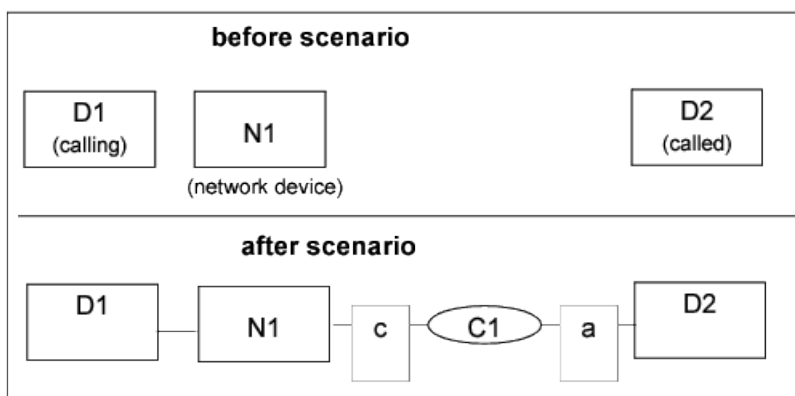
If the IP Direct Access ( IPDA ) feature is used with broken IP connection, as the Survivability Path consists of normal network interface devices, the call between the Access Point and the central OpenScape switch will be reported as an external call.

### 5.8.1 External incoming call

This scenario illustrates a manual external incoming call.

Since device D1 is located outside this CSTA sub-domain, it can not be directly monitored through this CSTA interface and therefore no events will be seen for that device. However, device N1, which is a network interface device (NID) (e.g., trunk interface), acts as a proxy for device D1.

This scenario describes the behaviour of a typical non-DNIS network interface device with ANI network information.



**Figure 27: Incoming external call**

## Call Scenarios

**Table 241: External incoming call**

Activity	Monitored Device N1		Monitored Device D2		Comments
1) Indicates an incoming call from N1.	Service Initiated				
	• initiatedConnection	N1C1			
	• initiatingDevice	N1			
	• localConnectionInfo	initiated			
	• cause	normal			
	• assocCalling	N1			
	• servicesPermitted	ClearConn			
1) The NID has connected to the call.	Originated				
	• originatedConnection	N1C1			
	• callingDevice	D1			
	• calledDevice	D2			
	• lastRedirectionDevice	NS			
	• localConnectionInfo	connected			
	• cause	normal			
	• assocCalling	N1			
	• networkCalling	D1			
	• servicesPermitted	ClearConn			
1) Device D2 is available and starts ringing.	Delivered		Delivered		
	• connection	D2C1	• connection	D2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• origNID connID	N1C1	• origNID connID	N1C1	
	• localConnectionInfo	connected	• localConnectionInfo	alert	
	• cause	normal	• cause	normal	
	• assocCalling	N1	• assocCalling	N1	
	• networkCalling	D1	• networkCalling	D1	

Activity	Monitored Device N1		Monitored Device D2		Comments
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn, SendUserInfo	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	Answer, ClearConn, Deflect, SendUserInfo	

**Remark:**

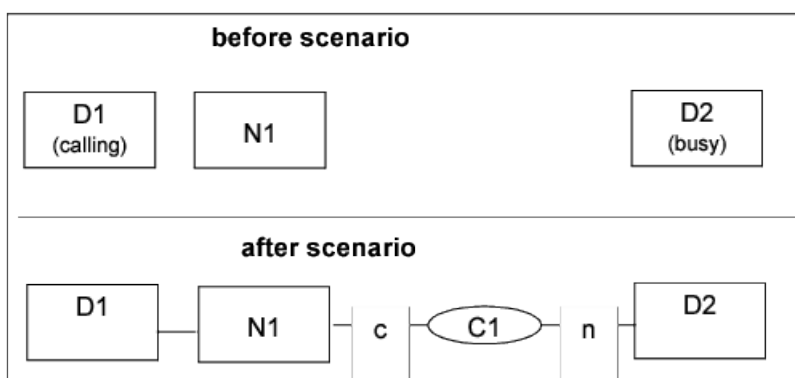
None

## 5.8.2 Incoming external call to a busy device

This scenario illustrates a manual external incoming call to a busy device.

Since device D1 is located outside this CSTA sub-domain, it can not be directly monitored through this CSTA interface and therefore no events will be seen for that device. However, device N1, which is a network interface device (NID) (e.g., trunk interface), acts as a proxy for device D1.

This scenario describes the behaviour of a typical non-DNIS network interface device with ANI network information.



**Figure 28: Incoming external call to a busy device**

**Table 242: External incoming call**

Activity	Monitored Device N1		Monitored Device D2		Comments
Steps 1-2 are shown in <a href="#">Section 5.8.1, "External incoming call"</a> .					
3. D2 is busy. The call can not be completed. D1 hears busy tone.	Failed		Failed		
	<ul style="list-style-type: none"><li>failedConnection</li></ul>	D2C1	<ul style="list-style-type: none"><li>failedConnection</li></ul>	D2C1	
	<ul style="list-style-type: none"><li>failingDevice</li></ul>	D2	<ul style="list-style-type: none"><li>failingDevice</li></ul>	D2	
	<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1	<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1	
	<ul style="list-style-type: none"><li>calledDevice</li></ul>	D2	<ul style="list-style-type: none"><li>calledDevice</li></ul>	D2	
	<ul style="list-style-type: none"><li>lastRedirectionDevice</li></ul>	NS	<ul style="list-style-type: none"><li>lastRedirectionDevice</li></ul>	NS	
	<ul style="list-style-type: none"><li>origNID connID</li></ul>	N1C1	<ul style="list-style-type: none"><li>origNID connID</li></ul>	N1C1	

## Call Scenarios

Activity	Monitored Device N1		Monitored Device D2		Comments
	• localConnectionInfo	connected	• localConnectionInfo	fail	
	• cause	busy	• cause	busy	
	• assocCalling	N1	• assocCalling	N1	
	• networkCalling	D1	• networkCalling	D1	
	• servicesPermitted	ClearConn	• servicesPermitted	none	
4. The busy connection is cleared immediately.	Connection Cleared		Connection Cleared		
	• droppedConnection	D2C1	• droppedConnection	D2C1	
	• releasingDevice	D2	• releasingDevice	D2	
	• localConnectionInfo	connected	• localConnectionInfo	null	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	ClearConn	• servicesPermitted	none	

### Remark:

- The protocol converter of the switching function will immediately send a Connection Cleared event after a connection goes into the busy failed state. This does not necessarily mean, that the connection physically goes to idle.
- The Network Interface Device behaviour differs from the normal extensions in the clearing of the busy connection, since the Network Interface Device will not be blocked after this situation. It means that no Failed event will be provided.

## 5.8.3 External incoming camp-on

This scenario illustrates an incoming call camp on to a busy device. The calling device D1 is located outside the CSTA sub-domain. This feature queues the call for the busy device D2 until that device becomes available.

Since device D1 is located outside this CSTA sub-domain, it can not be directly monitored through this CSTA interface and therefore no events will be seen for that device. However, device N1, which is a network interface device (NID) (e.g., trunk interface), acts as a proxy for device D1.

This scenario describes the behaviour of a typical non-DNIS network interface device with ANI network information.

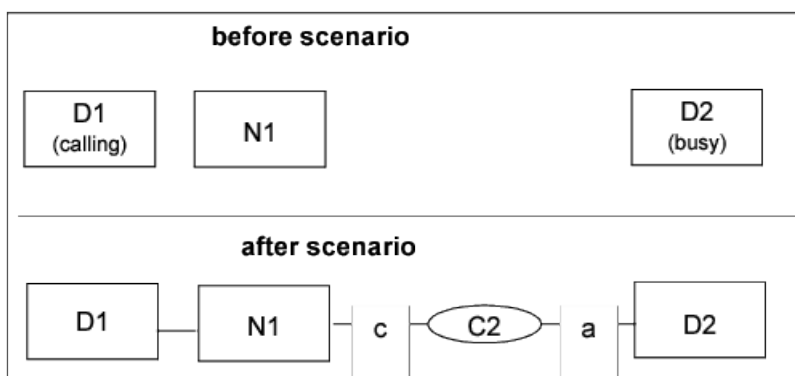


Figure 29: Incoming external call to a busy device

Table 243: External incoming call

Activity	Monitored Device N1		Monitored Device D2	Comments
1) Indicates an incoming call from N1.	Service Initiated			
	• initiatedConnection	N1C2		
	• initiatingDevice	N1		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn		
1) The NID is connected to the call.	Originated			
	• originatedConnection	N1C2		
	• callingDevice	D1		
	• calledDevice	D2		
	• lastRedirectionDevice	NS		
	• localConnectionInfo	connected		
	• cause	normal		
	• assocCalling	N1		
	• networkCalling	D1		
	• servicesPermitted	ClearConn		

## Call Scenarios

Activity	Monitored Device N1		Monitored Device D2		Comments
1) D2 is busy. The call is queued at D2.	<b>Queued</b>		<b>Queued</b>		At the incoming side the computing function gets the necessary information to identify the camp on, not like at the outgoing side.
	• queuedConnection	D2C2	• queuedConnection	D2C2	
	• queue	D2	• queue	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	queue	
	• cause	campOn	• cause	campOn	
	• assocCalling	N1	• assocCalling	N1	
	• networkCalling	D1	• networkCalling	D1	
	• servicesPermitted	ClearConn, SendUserInfo	• servicesPermitted	SendUserInfo	
1) Device D2 sometime later clears from its active call.			Connection Cleared		C1 was the previous active call of D2 and the reason why it was busy.
			• droppedConnection	D2C1	
			• releasingDevice	D2	
			• localConnectionInfo	null	
			• cause	normalClr	
			• servicesPermitted	none	
1) Since D2 is available, the call alerts D2.	<b>Delivered</b>		<b>Delivered</b>		
	• connection	D2C2	• connection	D2C2	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• origNID connID	N1C2	• origNID connID	N1C2	
	• localConnectionInfo	connected	• localConnectionInfo	alert	
	• cause	recall	• cause	recall	
	• assocCalling	N1	• assocCalling	N1	
	• networkCalling	D1	• networkCalling	D1	

Activity	Monitored Device N1		Monitored Device D2		Comments
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn, SendUserInfo	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	Answer, ClearConn, Deflect, SendUserInfo	

**Remark:**

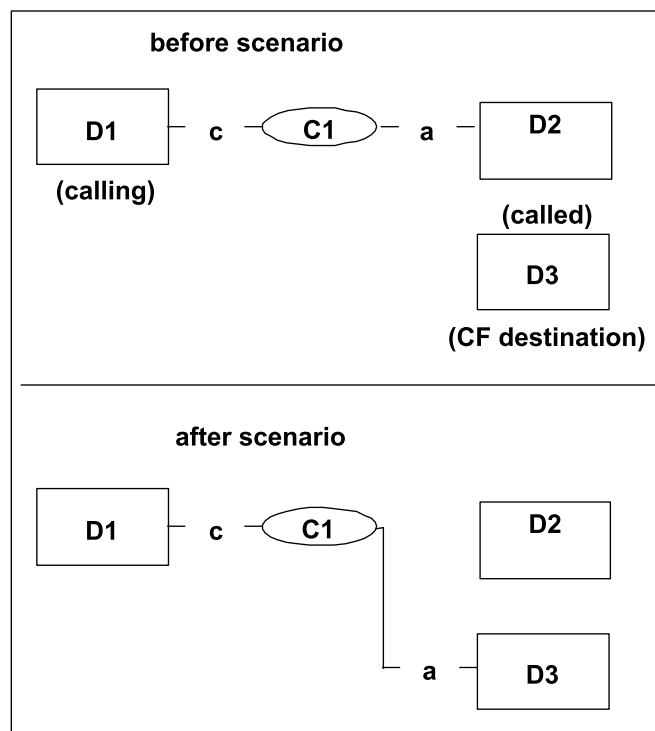
None

## 5.9 Forwarding Call Scenarios

The basic model of the switching function is to provide the Diverted event only to the device which leaves the call, there is an optional possibility to configure it to provide Diverted event also for the calling side in all scenarios where diversion involved. The local connection info remains the calling device's actual local connection info (connected). Services permitted are not provided in Diverted.

### 5.9.1 Call Forward No Answer

This scenario illustrates the event flow of a basic call forward no answer. A call comes to a device which is set to forward calls to a predefined device after a specified number of rings / time.



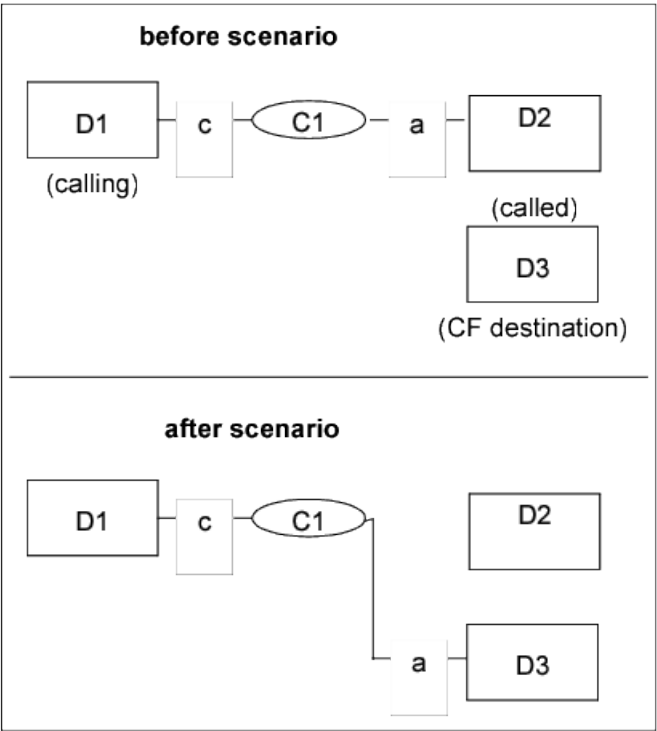


Figure 30: Call forward no answer

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before scenario" state.

Table 244: Call Forward - No Answer

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) D2 is alerted for a specified number of rings and then forwards the call to device D3.		Diverted		Device D3 is the device predefined by device D2 to forward its call.  The switching function sends the Diverted event only to the divertingDevice.
		• connection D2C1		
		• divertingDevice D2		
		• newDestination D3		
		• Calling D1		
		• calledDevice D2		
		• lastRedirectionDevice NS		
		• localConnectionInfo null		
		• cause forwardNoAnswer		

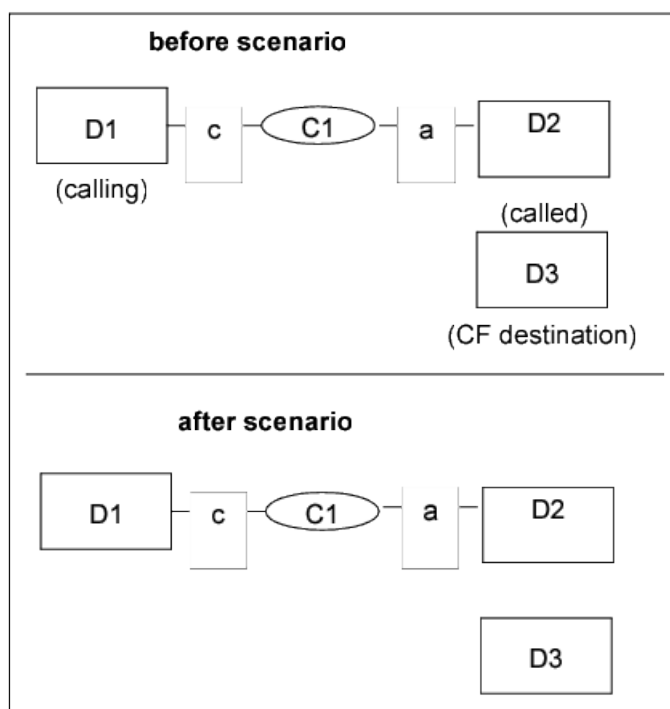
Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
			• servicesPermitted	none			
1) D3 starts ringing.	Delivered				Delivered		
	• connection	D3C1			• connection	D3C1	
	• alertingDevice	D3			• alertingDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	
	• lastRedirectionDevice	D2			• lastRedirectionDevice	D2	
	• localConnectionInfo	connected			• localConnectionInfo	alert	
	• cause	forwardNoAnswer			• cause	forwardNoAnswer	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo			• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	

**Remark:**

None

### 5.9.2 Call Forward no Answer: Forwarding Device and Destination (Busy) have Offered Mode on

This scenario illustrates the event flow of a basic call forward no answer. A call comes to a device which is set to forward calls to a predefined device after a specified number of rings / time. Normally this monitoring type requires Diverged and Offered events also for calling side. It is configurable.



**Figure 31: Call forwarding no answer to destination with offered mode**

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before scenario" state.

**Table 245: Call Forward - No Answer to Destination with Offered Mode**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
1) D2 is alerted for a specified number of rings and then forwards the call to device D3.	Diverted (optional)		Diverted			Device D3 is the device predefined by device D2 to forward its call.  The switching function sends the Diverted event only to the divertingDevice.
	• connection	D2C1	• connection	D2C1		
	• divertingDevice	D2	• divertingDevice	D2		
	• newDestination	D3	• newDestination	D3		
	• Calling	D1	• Calling	D1		
	• calledDevice	D2	• calledDevice	D2		
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS		
	• localConnectionInfo	connected	• localConnectionInfo	null		
	• cause	forwardNoAnswer	• cause	forwardNoAnswer		

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
	• servicesPermitted	none	• servicesPermitted	none			
1) Call is offered to D3	Offered (optional)				Offered		
	• offeredConnection	D3C1			• offeredConnection	D3C1	
	• offeredDevice	D3			• offeredDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	
	• lastRedirectionDevice	D2			• lastRedirectionDevice	D2	
	• localConnectionInfo	connected			• localConnectionInfo	alerting	
	• cause	forwardNoAnswer			• cause	forwardNoAnswer	
	• servicesPermitted	ClearConn			• servicesPermitted	AcceptCall, ClearConn, Deflect	
1) Application accepts the call					Accept Call IRequest		
					• callToBeAccepted	D3C1	
1) Acknowledged					Accept Call Response		
1) Call fails on D3	Diverted (optional)				Diverted		
	• divertedConnection	D3C1			• divertedConnection	D3C1	
	• divertingDevice	D3			• divertingDevice	D3	
	• newDest	D2			• newDest	D2	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	

## Call Scenarios

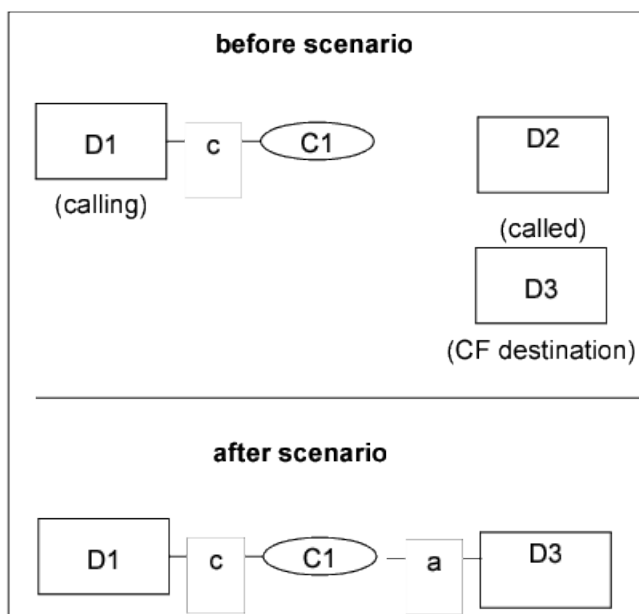
Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
	• localCon- nectionInfo	connected			• localCon- nectionInfo	null	
	• cause	redirected			• cause	redirected	
	• servicesPer- mitted	none			• servicesPer- mitted	none	
1) D2 starts to alert again	Delivered		Delivered				
	• delivered- Connection	D2C1	• delivered- Connection	D2C1			
	• alertingDe- vice	D2	• alertingDe- vice	D2			
	• callingDe- vice	D1	• callingDe- vice	D1			
	• calledDe- vice	D2	• calledDe- vice	D2			
	• lastRedirec- tionDevice	D3	• lastRedirec- tionDevice	NS			
	• localCon- nectionInfo	connected	• localCon- nectionInfo	alerting			
	• cause	redirected	• cause	redirected			
	• servicesPer- mitted	callBack, clearConn	• servicesPer- mitted	answer, clear- Conn, deflect			

### Remark:

None

## 5.9.3 Call Forward Immediate

This scenario illustrates the flow for a basic call forward immediate. A call comes to a device which is set to forward calls immediately to a predefined device.



**Figure 32: Call forward immediate**

See [Section 7.4.1, "Manually dialed call"](#) for the event flow to get into the "before scenario" state.

**Table 246: Call Forward - Immediate**

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
		Diverted		
		<ul style="list-style-type: none"> <li>diverting-Connection D2C1</li> </ul>		
		<ul style="list-style-type: none"> <li>divertingDevice D2</li> </ul>		
		<ul style="list-style-type: none"> <li>newDestination D3</li> </ul>		
		<ul style="list-style-type: none"> <li>callingDevice D1</li> </ul>		
		<ul style="list-style-type: none"> <li>calledDevice D2</li> </ul>		
		<ul style="list-style-type: none"> <li>lastRedirectionDev NS</li> </ul>		
		<ul style="list-style-type: none"> <li>localConnectionInfo null</li> </ul>		
		<ul style="list-style-type: none"> <li>cause forwardImm</li> </ul>		
		<ul style="list-style-type: none"> <li>servicesPermitted none</li> </ul>		

## Call Scenarios

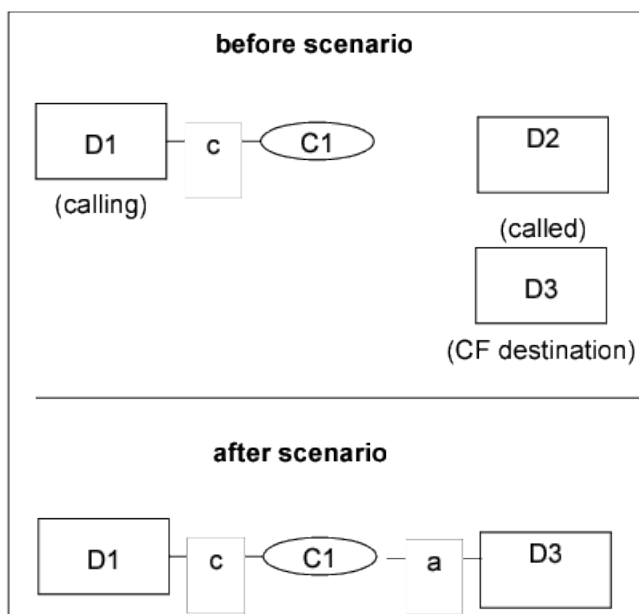
Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3		Comments
1) D3 starts ringing.	Delivered			Delivered		
	• connection	D3C1		• connection	D3C1	
	• alertingDe-vice	D3		• alertingDe-vice	D3	
	• callingDe-vice	D1		• callingDe-vice	D1	
	• calledDe-vice	D2		• calledDe-vice	D2	
	• lastRedirec-tionDevice	D2		• lastRedirec-tionDevice	D2	
	• localCon-nectionInfo	connected		• localCon-nectionInfo	alert	
	• cause	forwardImme-diate		• cause	forwardImme-diate	
	• servicesPer-mitted	CallBack, ClearConn, SendUserInfo		• servicesPer-mitted	Answer, ClearConn, Deflect, SendUserInfo	

**Remark:**

None

### 5.9.4 Call Forward Immediate: Called Party is OOS (Out Of Service)

This scenario illustrates the flow for a basic call forward immediate. A call comes to a device which is out of service and set to forward calls immediately to a predefined device.



**Figure 33: Call forward immediate - Called party out of service**

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before scenario" state.

**Table 247: Call Forward Immediate Called Party OutOfService (OOS)**

Activity	Monitored Device D1	Monitored Device D2 is OOS	Monitored Device D3	Comments
1) Called party D2 is Out of Service and D2 has Cf-All to D3		Diverted		
		• diverting-Connection	D2C1	
		• divertingDevice	D2	
		• newDestination	D3	
		• callingDevice	D1	
		• calledDevice	D2	
		• lastRedirectionDev	NS	
		• localConnectionInfo	null	
		• cause	forwardImm	
		• servicesPermitted	none	

## Call Scenarios

### Multiple Forwarding Scenarios

Activity	Monitored Device D1		Monitored Device D2 is OOS	Monitored Device D3		Comments
1) D3 starts ringing.	Delivered			Delivered		
	• connection	D3C1		• connection	D3C1	
	• alertingDevice	D3		• alertingDevice	D3	
	• callingDevice	D1		• callingDevice	D1	
	• caledDevice	D2		• caledDevice	D2	
	• lastRedirectionDev	D2		• lastRedirectionDev	D2	
	• localConnectionInfo	connected		• localConnectionInfo	alert	
	• cause	forwardImm		• cause	forwardImm	
	• servicesPermitted	CallBack, Clear Conn, SendUI		• servicesPermitted	Answer, Clear Conn, Deflect, SendUI	

**Remark:**

None

## 5.9.5 Call Forward Busy

In case of a Call forward busy similar event flow will be generated to the Call forward immediate case.

See [Section 5.9.3, "Call Forward Immediate"](#).

The only difference is the CSTA event cause which will be forwardbusy in this case.

**Remark:**

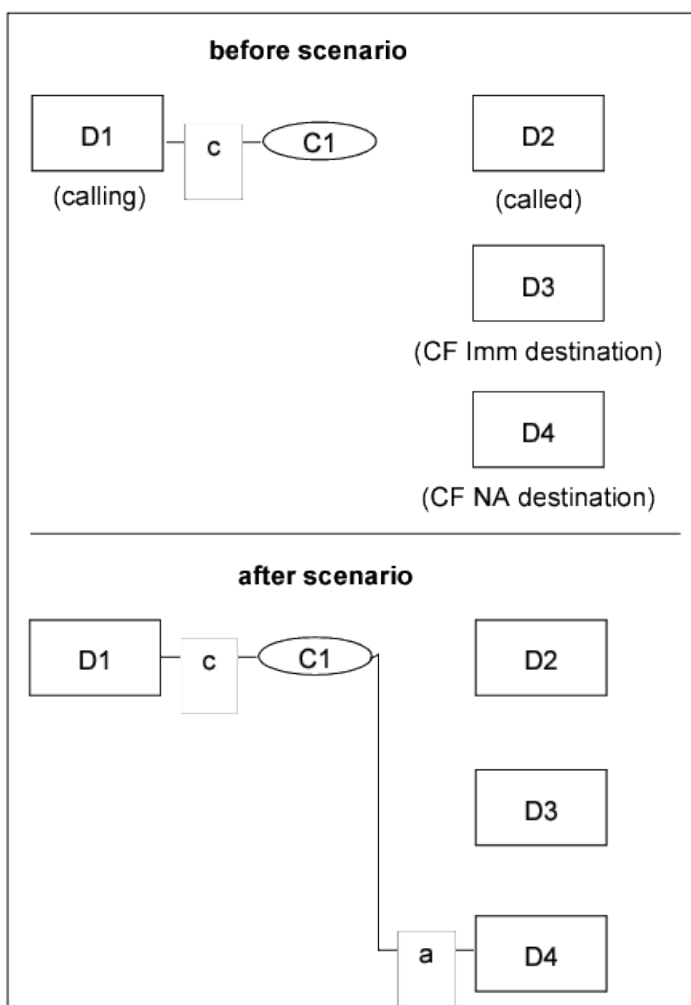
None

## 5.10 Multiple Forwarding Scenarios

### 5.10.1 Call Forward Immediate followed by Call Forward No Answer

This scenario illustrates a multiple forwarding call scenario.

A call comes to a device which is set to forward every incoming call to a predefined device immediately. The destination device is set to forward calls to another predefined device after a specified number of rings.



**Figure 34: Call forward immediate followed by call forward no answer**

See [Section 7.4.1, "Manually dialed call"](#) for the event flow to get into the "before scenario" state.

**Table 248: Call Forward Immediate followed by Call Forward No Answer**

Activity	Monitored Device D1	Monitored Device D2		Monitored Device D3
1) Device D2 has CF All to D3.		Diverted		
		• connection	D2C1	
		• divertingDevice	D2	
		• newDestination	D3	
		• Calling	D1	
		• calledDevice	D2	
		• lastRedirectionDevice	NS	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	
			• localConnectionInfo	null		
			• cause	forwardImm		
			• servicesPermitted	none		
1) D3 starts ringing.	Delivered				Delivered	
	• connection	D3C1			• connection	D3C1
	• alertingDevice	D3			• alertingDevice	D3
	• callingDevice	D1			• callingDevice	D1
	• calledDevice	D2			• calledDevice	D2
	• lastRedirectionDevice	NK			• lastRedirectionDevice	NK
	• localConnectionInfo	connected			• localConnectionInfo	alert
	• cause	forwardImmediate			• cause	forward
1) D3 is alerted for a specified number of rings then forwards the call to device D4.					Diverted	
					• connection	D3C1
					• divertingDevice	D3
					• newDestination	D4
					• Calling	D1
					• calledDevice	D2
					• lastRedirectionDevice	NS
					• localConnectionInfo	null
1) D4 starts ringing.	Delivered					
	• connection	D4C1				
	• alertingDevice	D4				
	• callingDevice	D1				
	• calledDevice	D2				
	• lastRedirectionDevice	D3				
	• localConnectionInfo	connected				
	• cause	forwardNoAnswer				

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	CallBack, ClearConn, SendUserInfo		

**Remark:**

None

## 5.10.2 Call Forward No Answer followed by Call Forward Immediate

### 5.10.2.1 Destination is available

This scenario illustrates a multiple forwarding call scenario.

A call comes to a device which is set to forward every incoming call to a predefined device after a specified number of rings. The destination device is set to forward calls to another predefined device immediately. D4 is available.

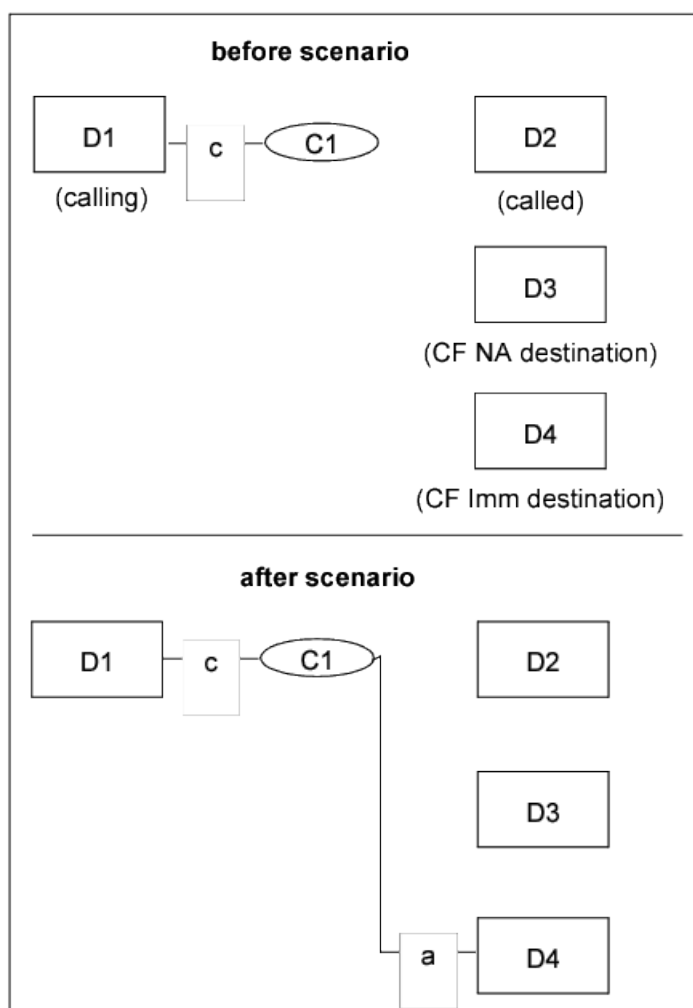


Figure 35: Call forward no answer followed by call forward immediate

Section 7.4.1, "Manually dialled call" for the event flow to get into the "before scenario" state.

**Table 249: Call Forward No Answer followed by Call Forward Immediate**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	
1) D2 starts ringing.	Delivered		Delivered			
	• connection	D2C1	• connection	D2C1		
	• alertingDevice	D2	• alertingDevice	D2		
	• callingDevice	D1	• callingDevice	D1		
	• calledDevice	D2	• calledDevice	D2		
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS		
	• localConnection Info	connected	• localConnection Info	alert		
	• cause	normal	• cause	normal		
1) D2 is alerted for a specified number of rings then tries to reach device D3. D3 has a call forwarding activated to D4.					Diverted	
					• connection	D3C1
					• divertingDevice	D3
					• newDestination	D4
					• Calling	D1
					• calledDevice	D2
					• lastRedirectionDevice	NS
					• localConnectionInfo	null
1) If D4 is available then the call will be forwarded from D2 to D4.			Diverted		• cause	forward
					• servicesPermitted	none
			• connection	D2C1		
			• divertingDevice	D2		
			• newDestination	D4		
			• Calling	D1		
			• calledDevice	D2		
			• lastRedirectionDevice	NS		

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3
			• localConnectionInfo	null	
			• cause	forwardNo Answer	
			• servicesPermitted	none	
1) D4 starts ringing.	Delivered				
	• connection	D4C1			
	• alertingDevice	D4			
	• callingDevice	D1			
	• calledDevice	D2			
	• lastRedirectionDevice	D4			
	• localConnectionInfo	connected			
	• cause	forwardNo Answer			
	• servicesPermitted	CallBack, ClearConn, SendUserInfo			

**Remark:**

Please note that the Call Forward No Answer happens physically after the Call Forward Immediate!

**5.10.2.2 Destination is not available**

A call comes to a device which is set to forward every incoming call to a predefined device after a specified number of rings. The destination device is set to forward calls to another predefined device immediately. D4 is not available

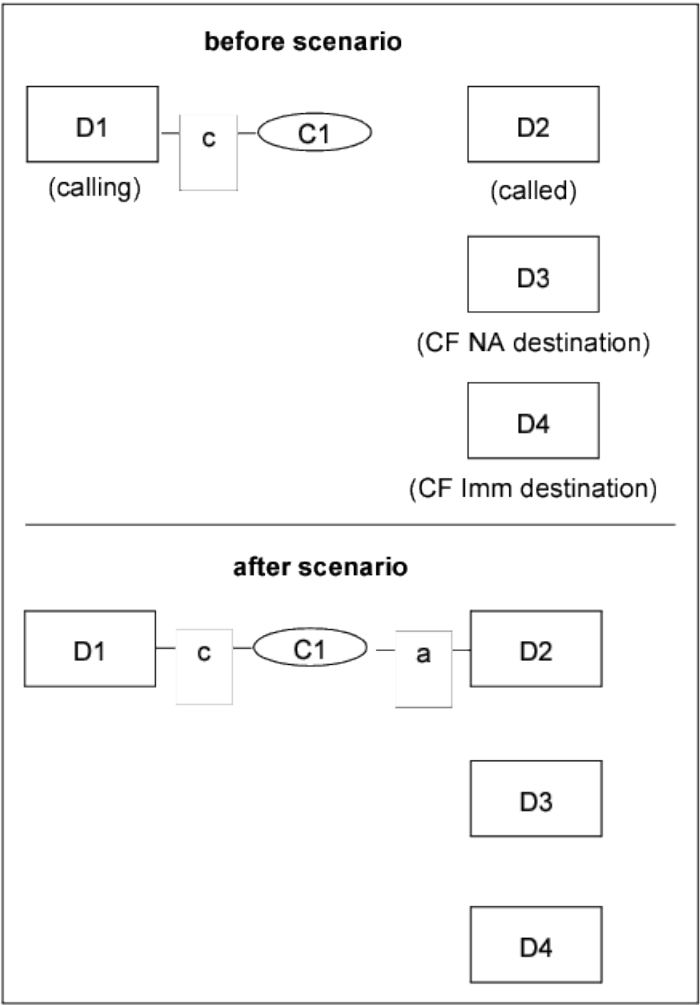


Figure 36: Call forward no answer followed by call forward immediate - Destination not available

Section 7.4.1, "Manually dialed call" for the event flow to get into the "before scenario" state.

Table 250: Call Forward No Answer followed by Call Forward Immediate

Activity	Monitored Device D1		Monitored Device D2		Monitored Devicr D3
1) D2 starts ringing.	Delivered		Delivered		
	• connection	D2C1	• connection	D2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	alert	

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	
	• cause	normal	• cause	normal		
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	An- swer,ClearConn, Deflect, Sen- dUserInfo		
1) D3 for- wards the call to D4.					Diverted	
					• connection	D30
					• divertingDevice	D3
					• newDestination	D4
					• Calling	D1
					• calledDevice	D2
					• lastRedirection Device	NS
					• localConnectionInfo	null
					• cause	forw
					• servicesPermitted	non
1) D4 is busy in another call.						

**Remark:**

In this case Device D2 will continue ringing, so the Call Forward No Answer will not be executed. After the next timeout the switch tries to reach D3 and D4 again (periodically) until D4 becomes available or D2 answers the call.

### 5.10.3 Call Forward Immediate followed by Call Forward Busy

This scenario illustrates a multiple forwarding call scenario.

A call comes to a device which is set to forward every incoming call to a predefined device immediately. The destination device is set to forward calls to another predefined device in case of it is busy in another call.

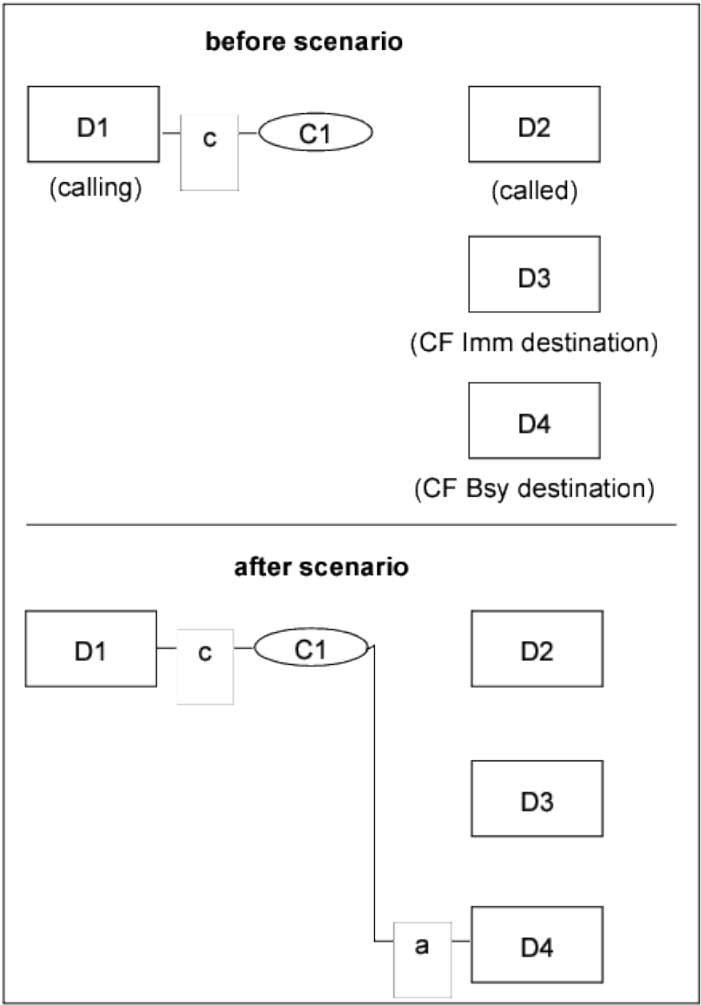


Figure 37: Call forward immediate followed by call forward busy

Section 7.4.1, "Manually dialed call" for the event flow to get into the "before scenario" state.

Table 251: Call Forward Immediate followed by Call Forward Busy

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Monitored Device D4	Comments
1) D2 forwards the call to D3.		Diverted			
		• connection D2C1			
		• diverting- D2 Device			
		• newDesti-D3 nation			
		• Calling D1			
		• calledDe- D2 vice			

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Monitored Device D4		Comments
			<ul style="list-style-type: none"><li>lastRedirection Device</li></ul>	NS					
			<ul style="list-style-type: none"><li>localConnection-Info</li></ul>	null					
			<ul style="list-style-type: none"><li>cause</li></ul>	forwardImm					
			<ul style="list-style-type: none"><li>services-Permitted</li></ul>	none					
1) D3 forwards the call to D4.					Diverted				D3 is busy in another call.
					<ul style="list-style-type: none"><li>connection</li></ul>	D3C1			
					<ul style="list-style-type: none"><li>diverting-Device</li></ul>	D3			
					<ul style="list-style-type: none"><li>newDestination</li></ul>	D4			
					<ul style="list-style-type: none"><li>Calling</li></ul>	D1			
					<ul style="list-style-type: none"><li>calledDevice</li></ul>	D2			
					<ul style="list-style-type: none"><li>lastRedirection-Device</li></ul>	D2			
					<ul style="list-style-type: none"><li>localConnection-Info</li></ul>	null			
					<ul style="list-style-type: none"><li>cause</li></ul>	forward-Busy			
					<ul style="list-style-type: none"><li>services-Permitted</li></ul>	none			
1) D4 starts ringing.	Delivered						Delivered		
	<ul style="list-style-type: none"><li>connection</li></ul>	D4C1					<ul style="list-style-type: none"><li>connection</li></ul>	D4C1	
	<ul style="list-style-type: none"><li>alerting-Device</li></ul>	D4					<ul style="list-style-type: none"><li>alerting-Device</li></ul>	D4	
	<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1					<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1	
	<ul style="list-style-type: none"><li>calledDevice</li></ul>	D2					<ul style="list-style-type: none"><li>calledDevice</li></ul>	D2	

## Call Scenarios

### Call Movement Scenarios

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3	Monitored Device D4		Comments
	• lastRedirection-Device	NK			• lastRedirection-Device	NK	
	• localConnection-Info	connected			• localConnection-Info	alert	
	• cause	forwardBusy			• cause	forward-Busy	
	• services-Permitted	CallBack, ClearConn, SendUserInfo			• services-Permitted	Answer, ClearConn, Deflect, SendUserInfo	

**Remark:**

none

## 5.11 Call Movement Scenarios

This section includes examples of moving calls from one device to another, initiated manually or by CSTA services.

### 5.11.1 Deflect Call service

This scenario illustrates how an alerting call is diverted to another destination.

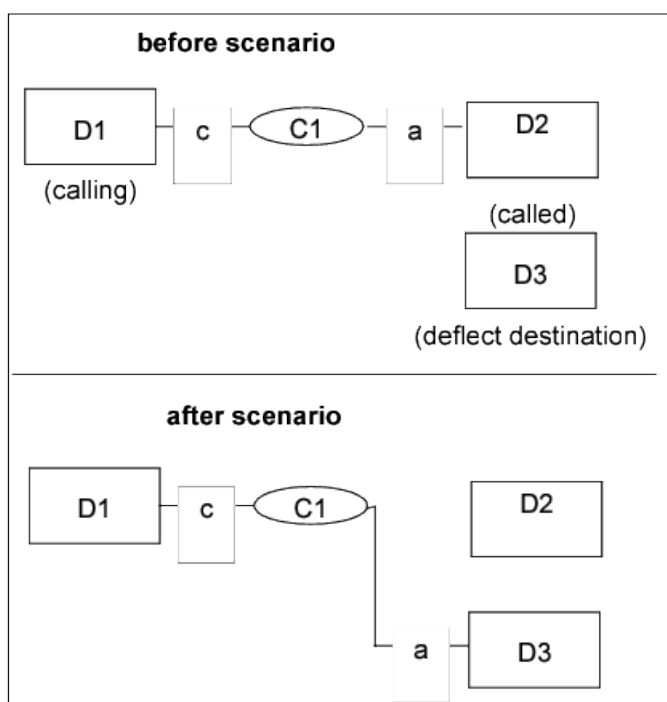


Figure 38: Defelct call service

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before service" state.

Table 252: Deflect Call service

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) Deflect service is invoked on behalf of D2.		Deflect Call Request		
		<ul style="list-style-type: none"> <li>connection to be deflected: D2C1</li> <li>new destination D3</li> </ul>		
1) Acknowledgement.		Deflect Call Response		
1) The event indicates that the call has been diverted from D1.		Diverted		The switching function sends the Diverted event only to the divertingDevice.
		<ul style="list-style-type: none"> <li>connection D2C1</li> </ul>		
		<ul style="list-style-type: none"> <li>divertingDevice D2</li> </ul>		
		<ul style="list-style-type: none"> <li>newDestination D3</li> </ul>		
		<ul style="list-style-type: none"> <li>callingDevice D1</li> </ul>		

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
			<ul style="list-style-type: none"> <li>calledDe-vice</li> </ul>	D2			
			<ul style="list-style-type: none"> <li>lastRedirec-tionDevice</li> </ul>	NS			
			<ul style="list-style-type: none"> <li>localConnec-tionInfo</li> </ul>	null			
			<ul style="list-style-type: none"> <li>cause</li> </ul>	redirected			
			<ul style="list-style-type: none"> <li>servicesPer-mitted</li> </ul>	none			
1) D3 starts ringing.	Delivered				Delivered		
	• connection	D3C1			• connection	D3C1	
	• alertingDe-vice	D3			• alertingDe-vice	D3	
	• callingDe-vice	D1			• callingDe-vice	D1	
	• calledDe-vice	D2			• calledDe-vice	D2	
	• lastRedirec-tionDevice	D2			• lastRedirec-tionDevice	D2	
	• localConnec-tionInfo	connected			• localConnec-tionInfo	alert	
	• cause	redirected			• cause	redirected	
	• servicesPer-mitted	CallBack, ClearConn, SendUserInfo			• servicesPer-mitted	Answer, ClearConn, Deflect, SendUserInfo	

### Remark:

None

## 5.11.2 Deflect call with ReRouting enabled

This scenario illustrates a successful Deflect Service when ReRouting is enabled.

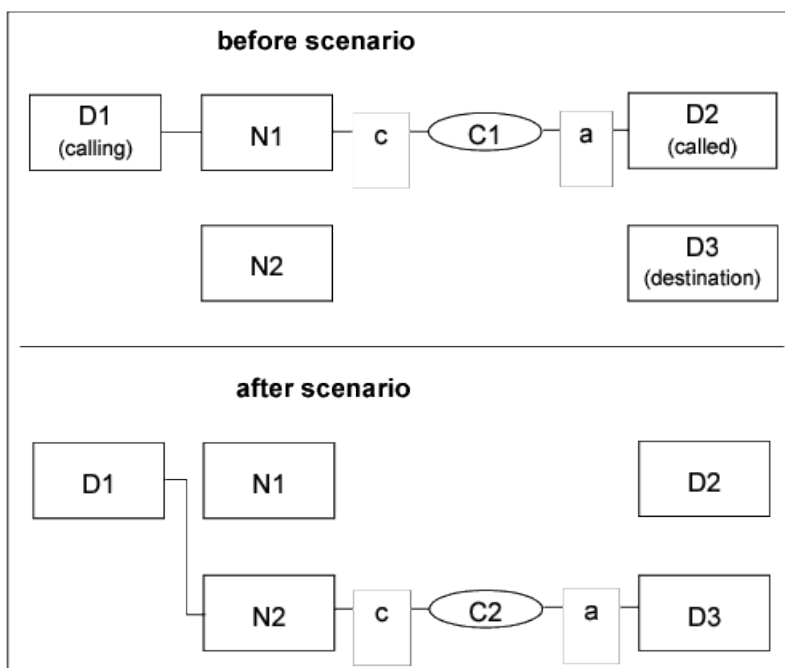


Figure 39: Deflect call with rerouting enabled

Table 253: Deflect call with ReRouting enabled

Activity	Monitored Device N1	Monitored Device D2	Monitored Device N2	Monitored Device D3	Comments
1) Deflect Call service is invoked on behalf of device D1.	Deflect Call Request				
	<ul style="list-style-type: none"> <li>conn-ToBe-Deflect</li> <li>newDesti-nation-Device</li> </ul>	D2C1  D3			
1) Acknowledgment.	Deflect Call Response				
1) New network device is seized incoming.			Service Initiated		
			<ul style="list-style-type: none"> <li>initiated-Connection</li> <li>initiating-Device</li> <li>localCon-nection-Info</li> <li>cause</li> </ul>	N2C2  N1  initiated  normal	

## Call Scenarios

Activity	Monitored Device N1	Monitored Device D2	Monitored Device N2		Monitored Device D3		Comments
			• services-Permitted	ClearConn			
1) N2 is connected to the call.			Originated				
			• originated-N2C2	Connection			
			• callingDe-	D1 vice			
			• calledDe-	D3 vice			
			• localCon-	connected			
			nection-	Info			
			• cause	normal			
			• services-Permitted	ClearConn			
1) The call alerts at D3.			• network-	D1			
			Calling-	Device			
			• assocCall-	N2			
			ingDe-	vice			
			Delivered		Delivered		
			• connec-	D3C2	• connec-	D3C2	
			tion		tion		
			• alerting-	D3	• alerting-	D3	
			Device		Device		
			• callingDe-	D1	• callingDe-	D1	
			vice		vice		
			• calledDe-	D3	• calledDe-	D3	
			vice		vice		
			• lastRedi-	NS	• lastRedi-	NS	
			rection-		rection-		
			Device		Device		
			• origNID	N2C2	• origNID	N2C2	
			• localCon-	connected	• localCon-	alerting	
			nection-		nec-		
			Info		tionInfo		

Activity	Monitored Device N1	Monitored Device D2	Monitored Device N2	Monitored Device D3	Comments
			<ul style="list-style-type: none"> <li>cause normal</li> <li>services-Permitted ClearConn, SendUserInfo</li> <li>network-Calling-Device D1</li> <li>assocCall-N2 ingDe-vice</li> </ul>	<ul style="list-style-type: none"> <li>cause normal</li> <li>services-Permitted Answer, ClearConn, Deflect, SendUserInfo</li> <li>network-Calling-Device D1</li> <li>assoc-Calling-Device N2</li> </ul>	
1) The network device N1 clears from the call.	Connection Cleared <ul style="list-style-type: none"> <li>dropped- N1C1 Connection</li> <li>releasing-N1 Device</li> <li>localCon-null nec-tionInfo</li> <li>cause normalClr</li> <li>services- none Permitted</li> </ul>	Connection Cleared <ul style="list-style-type: none"> <li>dropped- N1C1 Connection</li> <li>releasing-N1 Device</li> <li>localCon-alerting nec-tionInfo</li> <li>cause normalClr</li> <li>services- none Permitted</li> </ul>			
1) Device D2 clears also.		Connection Cleared <ul style="list-style-type: none"> <li>dropped- D2C1 Connection</li> <li>releasing-D2 Device</li> <li>localCon-null nec-tionInfo</li> <li>cause normalClr</li> <li>services- none Permitted</li> </ul>			

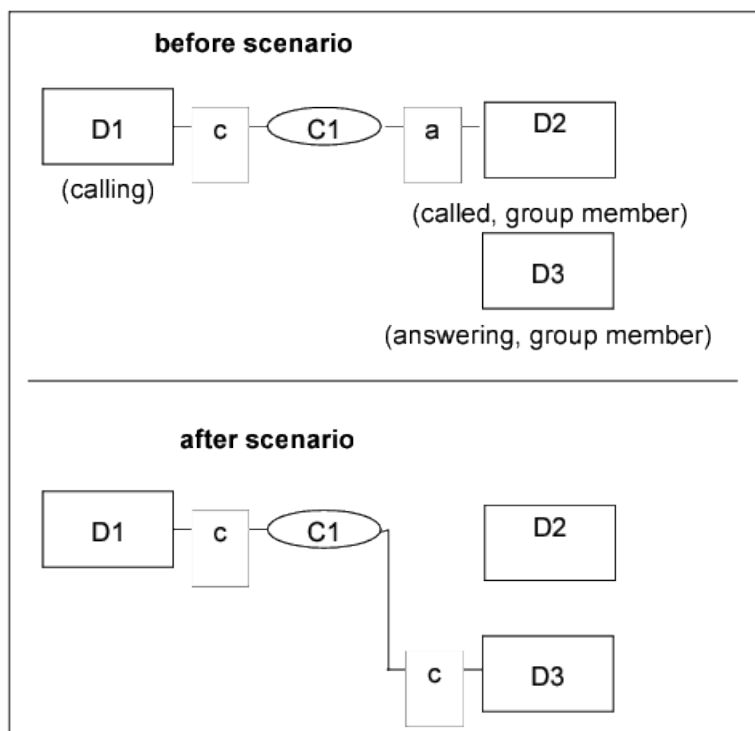
## Remark:

- When ReRouting is disabled, the call flow will be similar to [Section 5.11.1, "Deflect Call service", on page 413](#).
- In order to disable RE-ROUTING you need to remove the **FNAN** (RWSN in german) parameter from the AMO COT of the incoming trunk. This parameter enables the re-routing for Call Forward No Answer and Deflect scenarios.
- English AMO version:  
CHANGE-COT:COTNO=<number>, COTTYPE=COTDEL, PAR=FNAN;
- German AMO version:  
AENDERN-COT:COTNU=<number>, COTART=COTWEG, PAR=RWSN;

## 5.11.3 Manual group pickup

This scenario illustrates a pickup of a call that is alerting at a device as a member of a pickup group.

The call is moved and connected at the new specified destination.



**Figure 40: Manual group pickup**

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before scenario" state.

Table 254: Manual group pickup

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) D3 hits the pickup key.			Diverted				The switching function sends the Diverted event only to the divertingDevice.
			• connection	D2C1			
			• divertingDevice	D2			
			• newDestination	D3			
			• callingDevice	D1			
			• calledDevice	D2			
			• lastRedirectionDevice	NS			
			• localConnectionInfo	null			
			• cause	callPickup			
1) D3 is in connection with D1.	Established				Established		The switching function provides lastRedirectionDevice NS instead of the proper value.
	• established-Connection	D3C1			• established-Connection	D3C1	
	• answering-Device	D3			• answering-Device	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	
	• localConnectionInfo	connected			• localConnectionInfo	connected	
	• cause	CallPickup			• cause	CallPickup	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo			• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	

Remark:  
None

5.11.4 Manual directed park call

This scenario illustrates how a connected call is parked at another device.

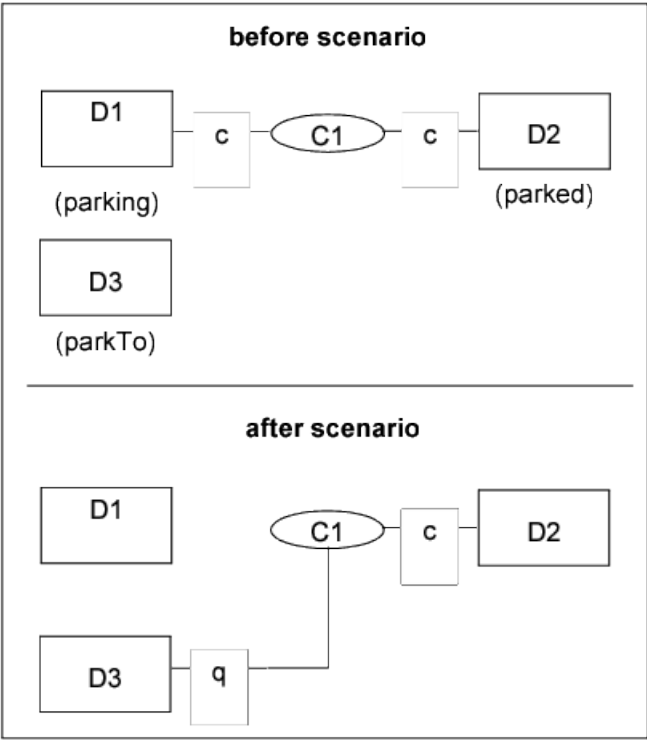


Figure 41: Manual direct park call

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before scenario" state.

Table 255: Manual directed park

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
1) D1 presses Park key. Connection is placed on hold.	Held		Held			
	• heldConnec-	D1C1	• heldConnec-	D1C1		
	tion		tion			
	• holdingDe-	D1	• holdingDe-	D1		
	vice		vice			
	• localConnec-	held	• localConnec-	connected		
	tionInfo		tionInfo			
	• cause	consultation	• cause	consultation		
	• servicesPer-	SendUserInfo,	• servicesPer-	ClearConn,		
	mitted	Reconnect	mitted	SendUserInfo		

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) D1 dials park destination.	Service Initiated			
	• initiatedConnection D1C2			
	• initiatingDevice D1			
	• localConnectionInfo initiated			
	• cause consultation			
1) D1 completes dialling destination's number ("1234")	• servicesPermitted ClearConn, DialDgt, Reconnect			
	Digits Dialed			
	• diallingConnection D1C1			
	• diallingDevice D1			
	• diallingSequence "1234"			
1) The call has been diverted from D1.	• localConnectionInfo initiated			
	• cause normal			
	Diverted			
	• connection D1C1			
	• divertingDevice D1			
	• newDestination D3			
	• callingDevice D1			
	• calledDevice D2			
	• lastRedirectionDevice NS			
	• localConnectionInfo null			
	• cause park			
				The switching function sends the Diverted event only to the divertingDevice.

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
	• servicesPermitted	none					
1) The switching function clears D1C2.	Connection Cleared						
	• droppedConnection	D1C2					
	• releasingDevice	D1					
	• localConnectionInfo	null					
	• cause	normalClr					
	• servicesPermitted	none					
1) The call is parked at device D3.			Queued		Queued		
			• queuedConnection	D3C1	• queuedConnection	D3C1	
			• queue	D3	• queue	D3	
			• callingDevice	D1	• callingDevice	D1	
			• calledDevice	D2	• calledDevice	D2	
			• lastRedirectionDevice	D1	• lastRedirectionDevice	D1	
			• localConnectionInfo	connected	• localConnectionInfo	queued	
			• cause	park	• cause	park	
			• servicesPermitted	ClearConn, Hold, SendUserInfo	• servicesPermitted	SendUserInfo	
1) Device D1 goes blocked.	Failed						
	• failedConnection	D1C3					
	• failingDevice	D1					
	• callingDevice	NK					
	• calledDevice	NK					

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
	<ul style="list-style-type: none"> <li>lastRedirectionDevice</li> </ul>	NS		
	<ul style="list-style-type: none"> <li>localConnectionInfo</li> </ul>	fail		
	<ul style="list-style-type: none"> <li>cause</li> </ul>	blocked		
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn		
1) D1 goes on-hook	Connection Cleared			
	<ul style="list-style-type: none"> <li>droppedConnection</li> </ul>	D1C3		
	<ul style="list-style-type: none"> <li>releasingDevice</li> </ul>	D1		
	<ul style="list-style-type: none"> <li>localConnectionInfo</li> </ul>	null		
	<ul style="list-style-type: none"> <li>cause</li> </ul>	normalClr		
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	none		

**Remark:**

- The switching function does not change the calling, called parameters in the event flow.
- ECMA TR/82 reports changing calling, called devices in the related scenario.

## 5.11.5 Manual system park

This scenario illustrates placing an established call on system park.

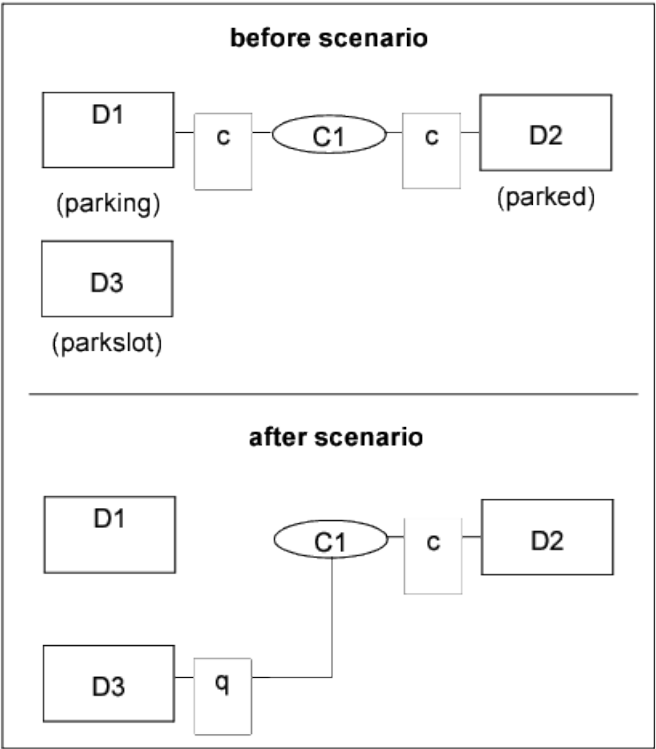


Figure 42: Manual system park

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before scenario" state.

Table 256: Manual system park

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3	Comments
1) D1 presses system park. The call has been diverted from D1, and placed on system hold.	Diverted			none	The switching function sends the Diverted event only to the divertingDevice.  Proper newDestination parameter cannot be provided due to switching function limitation.
	• connection	D1C1			
	• divertingDevice	D1			
	• newDestination	NK			
	• callingDevice	NK			
	• calledDevice	D2			
	• lastRedirectionDevice	NS			
	• localConnectionInfo	null			
	• cause	park			

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3	Comments
	<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	none			
1) D2 is parked to a parkslot.			<b>Queued</b>		A parkslot device cannot be monitored. That is why no event is reported for D3.
			<ul style="list-style-type: none"><li>queuedConnection</li></ul>	D3C1	
			<ul style="list-style-type: none"><li>queue</li></ul>	D3	
			<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1	
			<ul style="list-style-type: none"><li>calledDevice</li></ul>	D2	
			<ul style="list-style-type: none"><li>lastRedirectionDevice</li></ul>	NS	
			<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	connected	
			<ul style="list-style-type: none"><li>cause</li></ul>	park	
			<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	ClearConn, Hold, SendUserInfo	

**Remark:**

None

5.11.6 Manual system unpark

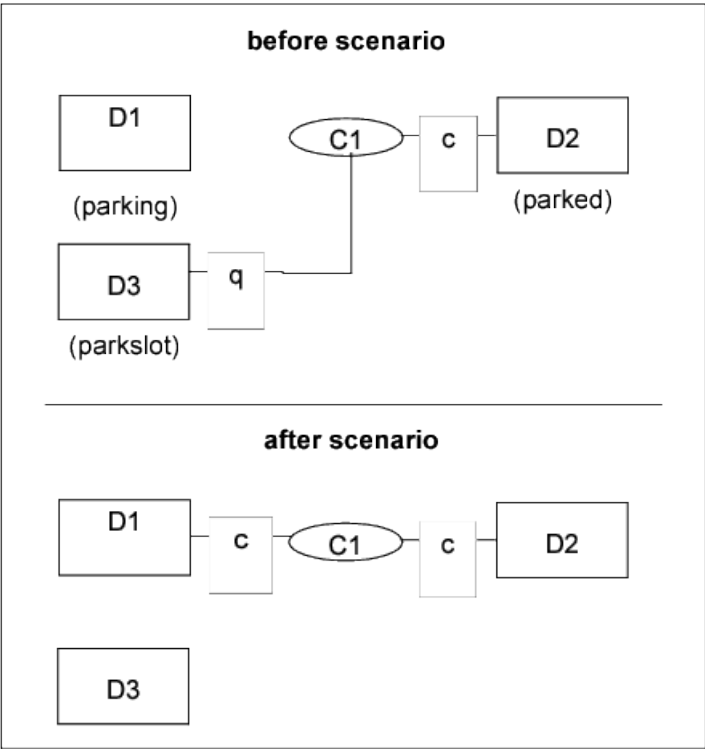


Figure 43: Manual system unpark

See [Section 5.11.5, "Manual system park", on page 424](#) for the event flow to get into the "before scenario" state.

Table 257: Manual system unpark

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
1) D1 hits the park key again: it invokes unpark.	Established		Established			
	• established-Connection	D1C1	• established-Connection	D1C1		
	• answering-Device	D1	• answering-Device	D1		
	• callingDevice	D1	• callingDevice	D1		
	• calledDevice	D2	• calledDevice	D2		
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS		
	• localConnectionInfo	connected	• localConnectionInfo	connected		
	• cause	park	• cause	park		

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo		

### 5.11.7 Call movement, Forwarding or Pick Up on Remote OpenScape 4000

Since Version 7 OpenScape 4000 is able to provide CSTA events about call movements, forwarding and pick up happening on the partner OpenScape 4000. Basically the event flow is the same as in the local case, with the restriction that there is no special handling of the call IDs and there is no way to connect the call IDs on the two systems. Any linkage must be based on the call linkage data as before. To have this functionality, switch on the mapping of the remote features, see Service Manual for details. Without the configuration or in case of any missing information, the system falls back to the original behaviour, which is to provide a CallInformationEvent showing the change in the CallLinkageData due to the change of the remote partner.

## 5.12 Hold/Retrieving Scenarios

This section includes examples of successful Hold and Retrieve Call scenarios.

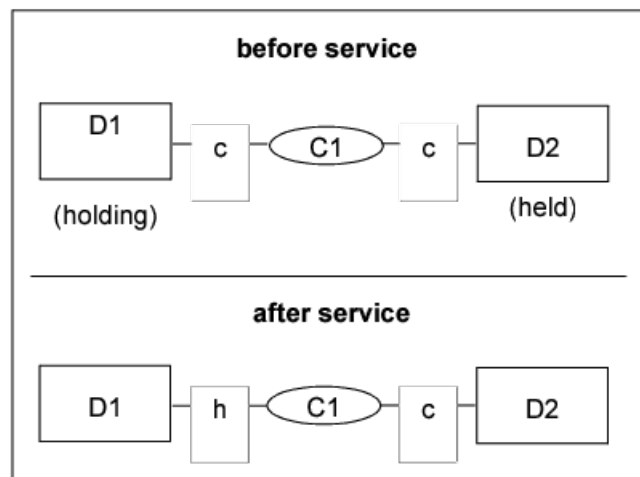
### 5.12.1 Hold/Retrieve Call

These scenarios illustrate the successful use of Hold Call and Retrieve Call services.

Hold Call service places an existing connection on hold.

Retrieve Call services retrieves a call from the hold state.

D1 is a DIGITE or CLIENT/DESK device.



**Figure 44: Hold call (D1 is holding)**

See [Section 7.5.1 "Successful answer call"](#) for the event flow to get into the "before service" state.

#### 5.12.1.1 Manual hold/Retrieve feature for DIGITE (except for KEYSETS)

The user of a DIGITE device can put on hold an active call and retrieve a held call using the HOLD Key or via CSTA Hold/Retrieve Call requests.

- During the holding state, the user can only retrieve the held call or go on hook.
- During the holding state, the partner can clear the call on hold.
- During the holding state, if the user goes on hook, then an immediate recall is initiated.
- The user of a DIGITE device cannot put on hold a consultation call.

## 1) D1 and D2 are connected

Table 258: D1 and D2 are connected

Activity	Monitored Device D1	Monitored Device D2
User D1 presses <b>HOLD KEY</b> ( <i>does not apply for D1 as UFIP</i> ) OR Application invokes <b>HoldCallRequest</b>	<b>Manual HOLD KEY</b> or <b>CSTA HoldCall</b> <i>callToBeHeld</i> D1 C1	
	<b>HeldEvent</b> <i>heldConnection</i> : D1 C1 <i>holdingDevice</i> : D1 <i>localConnectionInfo</i> : hold <i>cause</i> : normal <i>servicePermitted</i> : retrieveCall	<b>HeldEvent</b> <i>heldConnection</i> : D1C1 <i>holdingDevice</i> : D1 <i>localConnectionInfo</i> : connected <i>cause</i> : normal <i>servicePermitted</i> : clearConnection

## 2) D1 is holding D2

Table 259: D1 is holding D2

Activity	Monitored Device D1	Monitored Device D2
User D1 presses <b>HOLD KEY</b> ( <i>does not apply for D1 as UFIP</i> ) OR Application invokes <b>RetrieveCallRequest</b>	<b>Manual HOLD KEY</b> or <b>CSTA RetrieveCall</b> <i>callToBeRetrieved</i> D1 C1	
	<b>RetrievedEvent</b> <i>retrievedConnection</i> : D1 C1 <i>retrievingDevice</i> : D1 <i>localConnectionInfo</i> : connected <i>cause</i> : normal <i>servicePermitted</i> : <ul style="list-style-type: none"> <li>clearConnection,</li> <li>consuntationCall,</li> <li>holdCall,</li> <li>singleStepTransfer</li> </ul>	<b>CSTA RetrievedEvent</b> <i>retrievedConnection</i> : D1C1 <i>retrievingDevice</i> : D1 <i>localConnectionInfo</i> : connected <i>cause</i> : normal <i>servicePermitted</i> : <ul style="list-style-type: none"> <li>clearConnection,</li> <li>consuntationCall,</li> <li>holdCall,</li> <li>singleStepTransfer</li> </ul>

## 3) D1 is holding D2

Table 260: D1 is holding D2

Activity	Monitored Device D1	Monitored Device D2
User D1 goes on hook	<b>DeliveredEvent</b> <i>connection:</i> D1 C1 <i>alertingDevice:</i> D1 <i>calling Device:</i> D1 <i>calledDevice:</i> D2 <i>localConnectionInfo:</i> alerting <i>cause:</i> recall <i>servicePermitted:</i> <ul style="list-style-type: none"> <li>answerCall,</li> <li>clearConnection</li> </ul>	<b>DeliveredEvent</b> <i>connection:</i> D1 C1 <i>alertingDevice:</i> D1 <i>calling Device:</i> D1 <i>calledDevice:</i> D2 <i>localConnectionInfo:</i> connected <i>cause:</i> recall <i>servicePermitted:</i> <ul style="list-style-type: none"> <li>callBack,</li> <li>clearConnection</li> </ul>

**NOTICE:** For SIP-UFIP devices, the feature is a pure CSTA feature and cannot be invoked from Phone device in this way. This however is comparable to CSTA Consultation Call Request and no specific display info is sent to the phone. A user using a SIP-UFIP device can put on hold an active call and retrieve a held call using the CSTA Hold Call Request and CSTA Retrieve Call Request.

- During the holding state, the user can only retrieve the held call or go on hook.
- During the holding state, the partner can clear the call on hold.
- If the user goes on hook, then an immediate recall is initiated.

There are two exceptions:

- If the user has a SIP-UFIP station and puts the call on hold by pressing the HOLD key, then goes on-hook, no recall is initiated, but the call is ended.
- If the user has a SIP-UFIP station with uaCSTA capabilities and puts the call on hold via CSTA Hold Call request, then goes on-hook, no recall is initiated, but the call is ended.

**NOTICE:** For SIP-UFIP devices, the local hold phone feature is activated by pressing the hard HOLD key on the phone.

### 5.12.1.2 CSTA Hold/Retrieve feature for Client/Desk

In case of a Client/Desk configuration, the event flow for HOLD and RETRIEVE is similar as for SIP-UFIP devices. The difference is that **no recall is initiated if the user goes on hook**.

- 1) D1 (Client/Desk) and D2 are connected

**Table 261: D1 and D2 are connected**

Activity	Monitored Device D1	Monitored Device D2
Application invokes <b>HoldCallRequest</b>	<b>CSTA HoldCall</b> <i>callToBeHeld</i> D1 C1	
	<b>HeldEvent</b> <i>heldConnection:</i> D1 C1 <i>holdingDevice:</i> D1 <i>localConnectionInfo:</i> hold <i>cause:</i> normal <i>servicePermitted:</i> retrieveCall	<b>HeldEvent</b> <i>heldConnection:</i> D1 C1 <i>holdingDevice:</i> D1 <i>localConnectionInfo:</i> connected <i>cause:</i> normal <i>servicePermitted:</i> clearConnection

- 2) D1 (Client/Desktop) is holding D2

**Table 262: D1 and D2 are connected**

Activity	Monitored Device D1	Monitored Device D2
Application invokes <b>RetrieveCallRequest</b>	<b>CSTA RetrieveCall</b> <i>callToBeRetrieved</i> D1 C1	
	<b>RetrievedEvent</b> <i>retrievedConnection:</i> D1 C1 <i>retrievingDevice:</i> D1 <i>localConnectionInfo:</i> connected <i>cause:</i> normal <i>servicePermitted:</i> <ul style="list-style-type: none"> <li>clearConnection,</li> <li>consultationCall,</li> <li>holdCall,</li> <li>singleStepTransfer</li> </ul>	<b>RetrievedEvent</b> <i>retrievedConnection:</i> D1 C1 <i>retrievingDevice:</i> D1 <i>localConnectionInfo:</i> connected <i>cause:</i> normal <i>servicePermitted:</i> <ul style="list-style-type: none"> <li>clearConnection,</li> <li>consultationCall,</li> <li>holdCall,</li> <li>singleStepTransfer</li> </ul>

3) D1 (Client/Desk) is holding D2

**Table 263: D1 and D2 are connected**

Activity	Monitored Device D1	Monitored Device D2
User D1 goes on hook	<b>FailedEvent</b> <i>connection:</i> D1 C1 <i>callingDevice:</i> D1 <i>calledDevice:</i> D2 <i>localConnectionInfo:</i> fail <i>cause:</i> blocked <i>servicePermitted:</i> clearConnection	<b>FailedEvent</b> <i>connection:</i> D1 C1 <i>callingDevice:</i> D1 <i>calledDevice:</i> D2 <i>localConnectionInfo:</i> fail <i>cause:</i> blocked <i>servicePermitted:</i> clearConnection

## 5.12.2 Hold / Retrieve on Remote OpenScape 4000

Since Version 7 OpenScape 4000 is able to provide CSTA events about a hold or retrieve happening on the partner OpenScape 4000. The provided event flow is the same as in the local case. To have this functionality, switch on the mapping of the remote features, see Service Manual for details. Without the configuration or in case of any missing information, the system falls back to the original behaviour, which is to provide a CallInformationEvent showing the change in the CallLinkageData due to the state change of the remote partner.

## 5.13 Consultation Call Scenarios

This section illustrates examples of successful Consultation Call, Reconnect Call and Alternate Call initiated by CSTA services.

### 5.13.1 Successful Consultation Call

#### 5.13.1.1 Consulting party is a Non-SIP device

This service places an existing active call at a device on hold and initiates a new call from the same device.

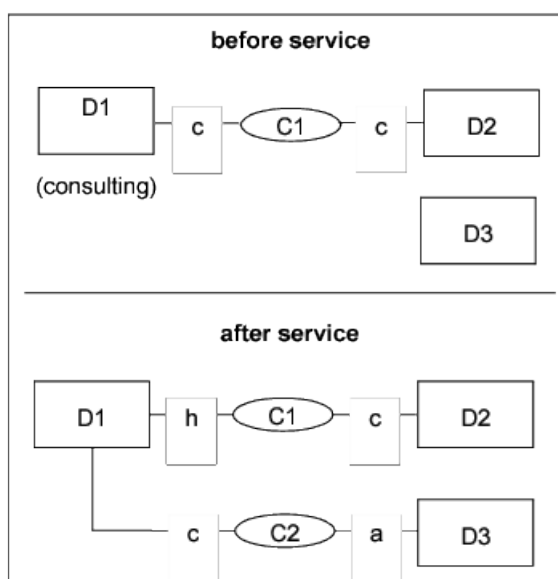


Figure 45: Consultation call

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before service" state.

Table 264: Consultation Call

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
1) Consulta- tion Call service is invoked on behalf of D1.	Consultation Call Request					
	• connToBe- Held	D1C1				
	• consulted- Device:	D3				
1) Acknowl- edgement.	Consultation Call Response					
	• consulted- Connection	D1C2				
1) Connection placed on hold.	Held		Held			
	• heldConnec- tion	D1C1	• heldConnec- tion	D1C1		
	• holdingDe- vice	D1	• holdingDe- vice	D1		
	• localConnec- tionInfo	held	• localConnec- tionInfo	connected		
	• cause	consultation	• cause	consultation		
	• servicesPer- mitted	SendUserInfot	• servicesPer- mitted	ClearConn, SendUserInfo		

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3		Comments
1) D1 begins to dial.	Service Initiated					
	• initiatedCon-	D1C2				
	nection					
	• initiatingDe-	D1				
	vice					
	• localConnec-	initiated				
	tionInfo					
	• cause	consultation				
	• servicesPer-	ClearConn,				
	mitted	Reconn,				
1) D1 completes dialling D3's number ("1234")	Digits Dialed					
	• diallingCon-	D1C1				
	nection					
	• diallingDe-	D1				
	vice					
	• diallingSe-	"1234"				
	quence					
	• localConnec-	initiated				
	tionInfo					
	• cause	consultation				
	Originated					
	• originated-	D1C2				
	Connection					
	• callingDe-	D1				
	vice					
	• calledDe-	D3				
	vice					
	• lastRedirec-	NS				
	tionDevice					
	• localConnec-	connected				
	tionInfo					
	• cause	consultation				
	• servicesPer-	ClearConn				
	mitted					
1) D3 starts ringing.	Delivered			Delivered		
	• connection	D3C2		• connection	D3C2	

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3		Comments
	• alertingDe- vice	D3		• alertingDe- vice	D3	
	• callingDe- vice	D1		• callingDe- vice	D1	
	• calledDe- vice	D3		• calledDe- vice	D3	
	• lastRedirec- tionDevice	NS		• lastRedirec- tionDevice	NS	
	• localConnec- tionInfo	connected		• localConnec- tionInfo	alert	
	• cause	normal		• cause	normal	
	• servicesPer- mitted	CallBack, ClearConn, SendUserInfo, Transfer, Reconnect		• servicesPer- mitted	Answer, ClearConn, Deflect, SendUserInfo	

**Remark:**

The manual case is similar to the described event flow.

### 5.13.1.2 Consulting party is a SIP device

This service places an existing active call at a device on hold and initiates a new call from the same device.

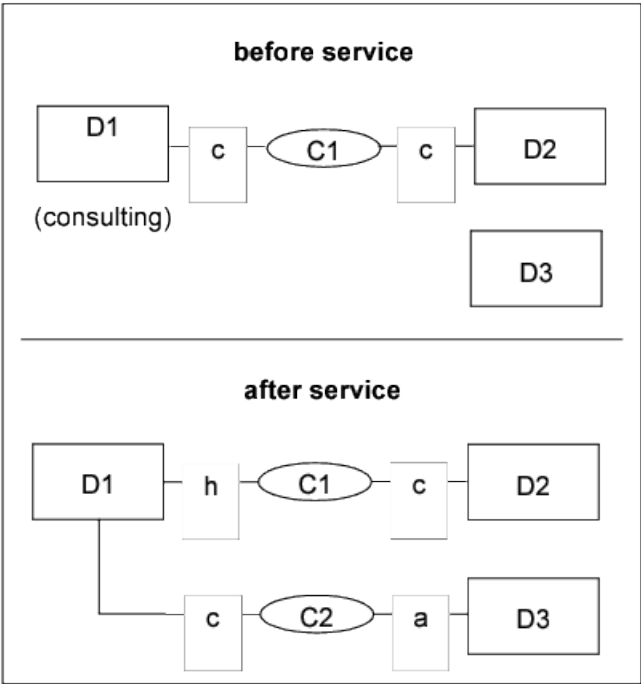


Figure 46: Consultation call (Cosultation party is a SIP device)

See [Section 5.5.1, "Succesful answer call"](#) for the event flow to get into the "before service" state.

Table 265: Consultation Call - Consultation party is a SIP device

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Consulta- tion button is pressed on D1.	• connToBe- Held	D1C1					
	• consulted- Device:	D3					
1) Acknowl- edgement.	Consultation Call Response						
	• consulted- Connection	D1C2					
1) Connection placed on hold.	Held		Held				
	• heldConnec- tion	D1C1	• heldConnec- tion	D1C1			
	• holdingDe- vice	D1	• holdingDe- vice	D1			
	• localCon- nectionInfo	held	• localCon- nectionInfo	connected			
	• cause	normal	• cause	normal			

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
	<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	none	<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	ClearConn, Hold, SendUserInfo		
1) D1 begins to dial.	Service Initiated					
	<ul style="list-style-type: none"><li>initiatedConnection</li></ul>	D1C2				
	<ul style="list-style-type: none"><li>initiatingDevice</li></ul>	D1				
	<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	initiated				
	<ul style="list-style-type: none"><li>cause</li></ul>	normal				
	<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	ClearConn, (from HP4k V6 only!)				
1) D1 completes dialling D3's number ("1234")	Digits Dialed					
	<ul style="list-style-type: none"><li>diallingConnection</li></ul>	D1C1				
	<ul style="list-style-type: none"><li>diallingDevice</li></ul>	D1				
	<ul style="list-style-type: none"><li>diallingSequence</li></ul>	"1234"				
	<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	initiated				
	<ul style="list-style-type: none"><li>cause</li></ul>	consultation				
	Originated					
	<ul style="list-style-type: none"><li>originated-Connection</li></ul>	D1C2				
	<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1				
	<ul style="list-style-type: none"><li>calledDevice</li></ul>	D3				
	<ul style="list-style-type: none"><li>lastRedirectionDevice</li></ul>	NS				
	<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	connected				
	<ul style="list-style-type: none"><li>cause</li></ul>	normal				

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3		Comments
	• servicesPermitted	ClearConn (from HP4k V6 only!)				
1) D3 starts ringing.	Delivered			Delivered		
	• connection	D3C2		• connection	D3C2	
	• alertingDevice	D3		• alertingDevice	D3	
	• callingDevice	D1		• callingDevice	D1	
	• calledDevice	D3		• calledDevice	D3	
	• lastRedirectionDevice	NS		• lastRedirectionDevice	NS	
	• localConnectionInfo	connected		• localConnectionInfo	alert	
	• cause	normal		• cause	normal	
	• servicesPermitted	ClearConn. (from HP4k V6 only!)		• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	

### Remark:

The manual case is similar to the described event flow.

## 5.13.2 Consulting out of a conference

This scenario illustrates the case when a conference member holds the conference. D1 is a Non-SIP device.

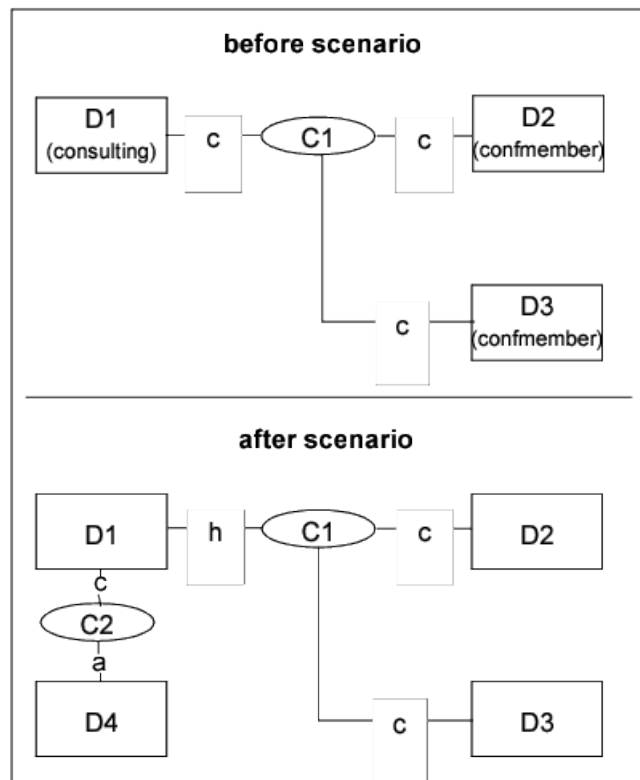


Figure 47: Conference

Table 266: Conference

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) Device D1 hits the consultation key.	Held			Note that there is no Held event for devices D2 and D3.  This is a switching function limitation.
	• heldConnection	D1C1		
	• holdingDevice	D1		
	• localConnectionInfo	hold		
	• cause	consultation		
1) Since device D1 still off-hook it can dial.	Service Initiated			
	• initiatedConnection	D1C2		
	• initiatingDevice	D1		

## Call Scenarios

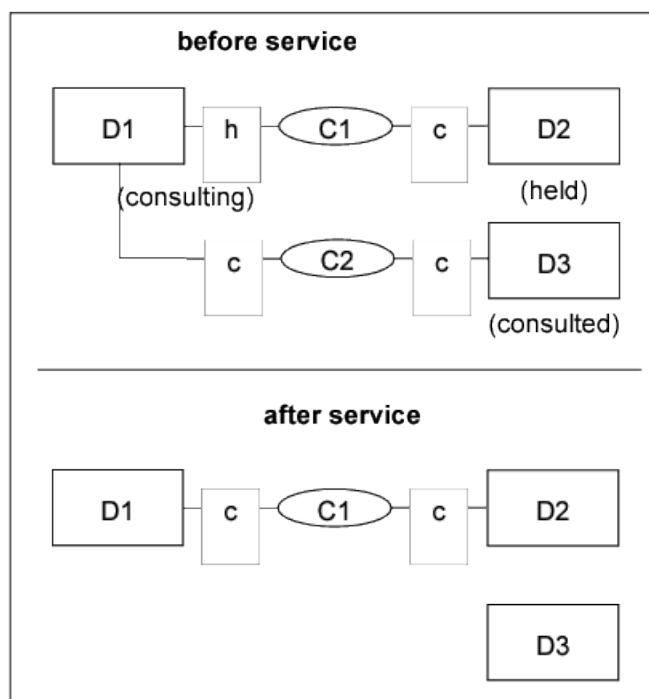
Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
	<ul style="list-style-type: none"> <li>localConnec- tionInfo</li> </ul>			
	<ul style="list-style-type: none"> <li>cause</li> </ul>	consultation		
	<ul style="list-style-type: none"> <li>servicesPer- mitted</li> </ul>	ClearConn, DialDgt, Re- conn		

### Remark:

Events for D4 are not shown above, because they are not relevant in this context.

## 5.13.3 Reconnect Call

This service clears an existing connection and then retrieves a previously held connection at the same device.



**Figure 48: Reconnect call**

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

Table 267: Reconnect Call

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Retrieve Call service is invoked on behalf of D1.	Reconnect Call Request						
	• heldConne- ction	D1C1					
	• activeCon- nection	D1C2					
1) Acknowledgement.	Reconnect Call Response						
1) Device D1 is cleared from the active call.	Connection Cleared				Connection Cleared		
	• droppedCon- nection	D1C2			• droppedCon- nection	D1C2	
	• releasingDe- vice	D1			• releasingDe- vice	D1	
	• localConnec- tionInfo	null			• localConnec- tionInfo	connected	
	• cause	normalClr			• cause	normalClr	
	• servicesPer- mitted	none			• servicesPer- mitted	ClearConn	
1) Device D1 is connect- ed back to the previously held call.	Retrieved		Retrieved				
	• retrieved- Connection	D1C1	• retrieved- Connection	D1C1			
	• Retrieving	D1	• Retrieving	D1			
	• localConnec- tionInfo	connected	• localConnec- tionInfo	connected			
	• cause	normal	• cause	normal			
	• servicesPer- mitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• servicesPer- mitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo			
1) As a result of D1C2 clearing, remining device D3 goes blocked.					Failed		
					• failedCon- nection	D3C2	
					• failingDe- vice	D3	
					• callingDe- vice	D1	

## Call Scenarios

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3		Comments
			<ul style="list-style-type: none"> <li>calledDe-vice</li> </ul>	D3	
			<ul style="list-style-type: none"> <li>lastRedirec-tionDevice</li> </ul>	NS	
			<ul style="list-style-type: none"> <li>localConnec-tionInfo</li> </ul>	-fail	
			<ul style="list-style-type: none"> <li>cause</li> </ul>	blocked	
			<ul style="list-style-type: none"> <li>servicesPer-mitted</li> </ul>	ClearConn	
1) As a result of D1C2 clearing, D3C2 is also cleared.			Connection Cleared		
			<ul style="list-style-type: none"> <li>droppedCon-nection</li> </ul>	D3C2	
			<ul style="list-style-type: none"> <li>releasingDe-vice</li> </ul>	D3	
			<ul style="list-style-type: none"> <li>localConnec-tionInfo</li> </ul>	-null	
			<ul style="list-style-type: none"> <li>cause</li> </ul>	normalClr	
			<ul style="list-style-type: none"> <li>servicesPer-mitted</li> </ul>	none	

### Remark:

The manual case is similar to the described event flow.

## 5.13.4 Held Party Releases

### 5.13.4.1 Consulting device is a Non-SIP device

This scenario illustrates the situation when the held party in a consultation releases the call.

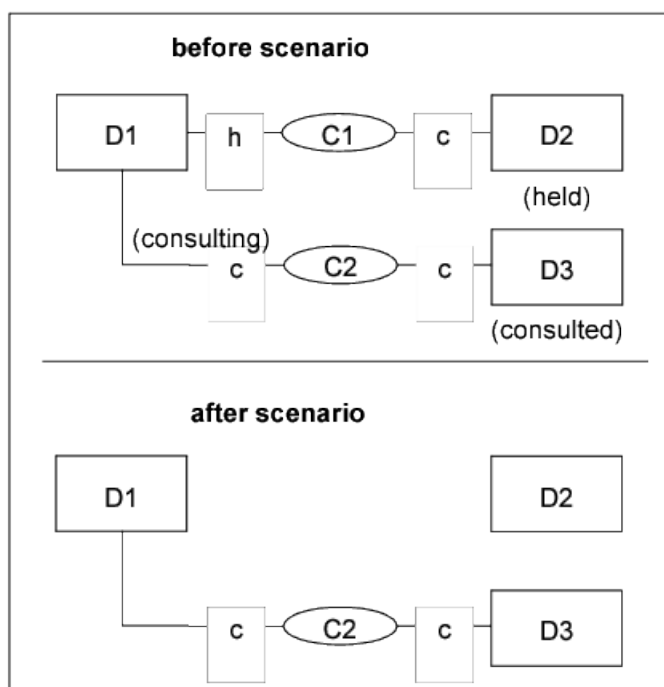


Figure 49: Held party release

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before scenario" state.

Table 268: Held Party Releases

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) Device D2 is cleared from the held call.	Connection Cleared	Connection Cleared	none	
	• droppedCon-D2C1 nection	• droppedCon-D2C1 nection		
	• releasingDe- D2 vice	• releasingDe- D2 vice		
	• localConnec-held tionInfo	• localConnec-null tionInfo		
	• cause normalClr	• cause normalClr		
1) Device D1 is cleared from the held call.	• servicesPer- none mitted	• servicesPer- none mitted		
	Connection Cleared			
	• droppedCon-D1C1 nection			
	• releasingDe- D1 vice			
	• localConnec-null tionInfo			

Call Scenarios

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3	Comments
1) Call information is provided for device D1	• cause	normalClr			CallInformation event is generated to provide the changed permitted services.
	• servicesPermitted	none			
	• ConnectionId	D1C2			
	• Subject deviceId:	D1			
	• servicesPermitted	ClearConn, Consultation-Call, Hold, SST, GenDgt, GenTelTones, SendUserInfo			

Remark:

None

5.13.4.2 Consulting device is a SIP device

This scenario illustrates the situation when the held party in a consultation releases the call.

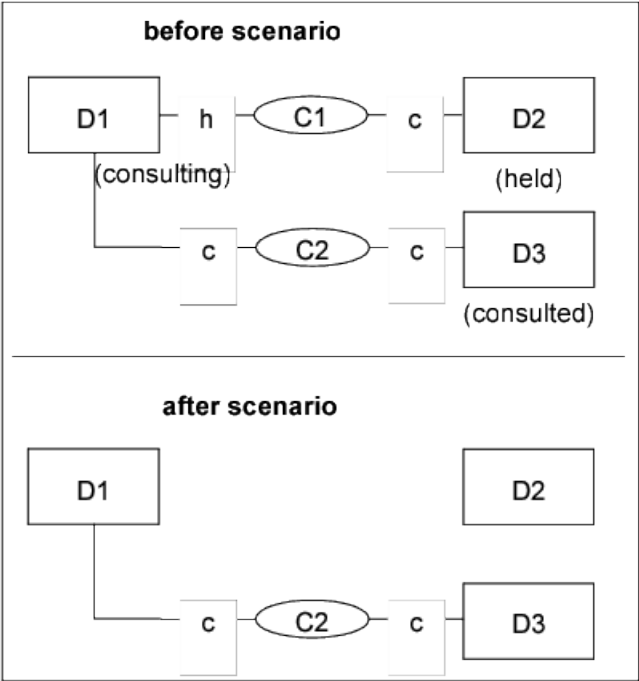


Figure 50: Held party release

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before scenario" state.

**Table 269: Held Party Releases**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
1) Device D2 is cleared from the held call.	Failed		Connection Cleared		none	
	• failedCon- nection	D1C1	• droppedCon-D2C1 nection			
	• failingDe- vice	D1	• releasingDe- D2 vice			
	• callingDe- vice	D1	• localConnec-null tionInfo			
	• calledDe- vice	D2	• cause	normalClr		
	• lastRedirec- tionDevice	NS	• servicesPer- mitted	none		
	• localConnec-fail tionInfo					
	• cause	blocked				
1) Device D1 is cleared from the held call.	• servicesPer- mitted	ClearConn (from HP4k V6 only!)				
	Connection Cleared					
	• droppedCon-D1C1 nection					
	• releasingDe- D1 vice					
	• localConnec-null tionInfo					
	• cause	normalClr				
	• servicesPer- mitted	none				

**Remark:**

None

## 5.13.5 Alternate Call

5.13.5.1 Consulting party is a Non-SIP device

This service places an existing active call on hold and then retrieves a previously held call at the same device. The effect of this service is to swap the device's active and held calls.

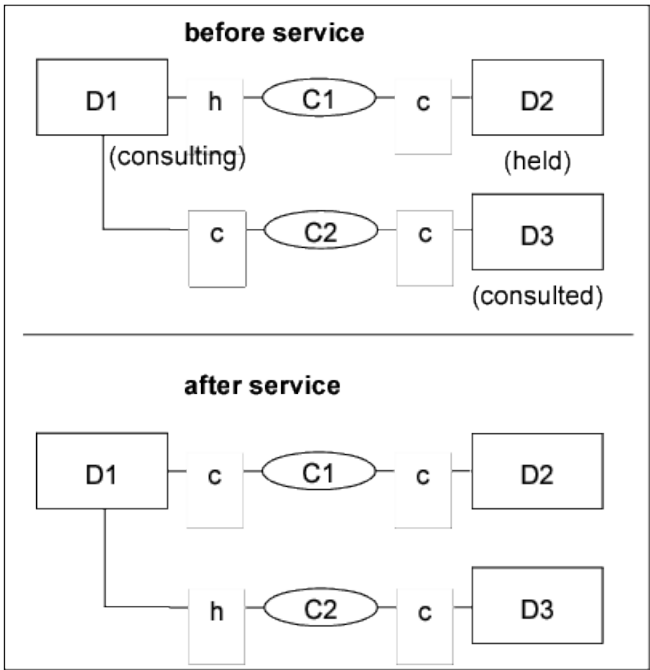


Figure 51: Alternate call

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

Table 270: Alternate Call

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) Alternate Call service is invoked on behalf of D1.	Alternate Call Request			
	<ul style="list-style-type: none"><li>heldConnection D1C1</li><li>activeConnection D1C2</li></ul>			
1) Acknowledgement.	Alternate Call Response			
1) Connection D1C2 is placed on hold in the active call.	Held		Held	
	<ul style="list-style-type: none"><li>heldConnection D1C2</li><li>holdingDevice D1</li></ul>		<ul style="list-style-type: none"><li>heldConnection D1C2</li><li>holdingDevice D1</li></ul>	

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
	<ul style="list-style-type: none"><li>localConnec- tionInfo</li></ul>				<ul style="list-style-type: none"><li>localConnec- connected tionInfo</li></ul>		
	<ul style="list-style-type: none"><li>cause</li></ul>	alternate			<ul style="list-style-type: none"><li>cause</li></ul>	alternate	
	<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	SendUserInfo			<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	SendUserIn- fo, Hold, ClearConn	
1) Device D1 is connect- ed back to the previously held call.	Retrieved		Retrieved				
	<ul style="list-style-type: none"><li>retrieved- Connection</li></ul>	D1C1	<ul style="list-style-type: none"><li>retrieved- Connection</li></ul>	D1C1			
	<ul style="list-style-type: none"><li>Retrieving</li></ul>	D1	<ul style="list-style-type: none"><li>Retrieving</li></ul>	D1			
	<ul style="list-style-type: none"><li>localConnec- connected tionInfo</li></ul>		<ul style="list-style-type: none"><li>localConnec- connected tionInfo</li></ul>				
	<ul style="list-style-type: none"><li>cause</li></ul>	alternate	<ul style="list-style-type: none"><li>cause</li></ul>	alternate			
	<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	ClearConn, AlternateCall,  Conference- Call, GenTel- Tones, Sen- dUserInfo	<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	ClearConn, Consult, Hold, GenDgt, GenTelTones, SendUserInfo			

**Remark:**

The manual case is similar to the described event flow.

### 5.13.5.2 Consulting party is a SIP device

Consulting SIP party places an existing active call on hold and then retrieves a previously held call at the same device. The effect of this feature is to swap the device's active and held calls.

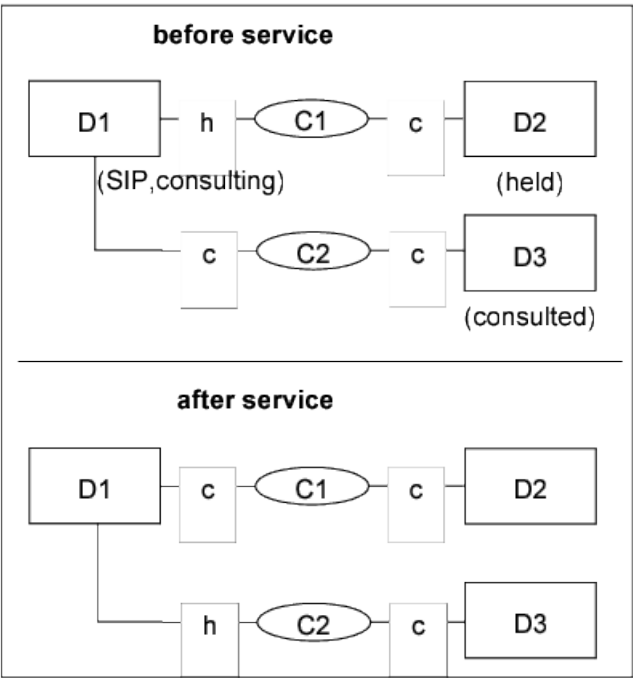


Figure 52: Alternate call on SIP device

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

Table 271: Alternate Call on SIP device

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) SIP presses alternate: Connection D1C2 is placed on hold in the active call.	Held				Held		
	• heldConnec- tion	D1C2			• heldConnec- tion	D1C2	
	• holdingDe- vice	D1			• holdingDe- vice	D1	
	• localConnec- tionInfo	hold			• localConnec- tionInfo	connected	
	• cause	normal			• cause	normal	
1) Device D1 is connect- ed back to the previously held call.	Retrieved		Retrieved				
	• retrieved- Connection	D1C1	• retrieved- Connection	D1C1			
	• Retrieving	D1	• Retrieving	D1			
	• localConnec- tionInfo	connected	• localConnec- tionInfo	connected			

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
	• cause	normal	• cause	normal		
	• servicesPermitted	ClearConn, Single-StepTransfer (from HP4k V6 only!)	• servicesPermitted	ClearConn, Consult, Hold, GenDgt, GenTelTones, SendUserInfo		

Remark:

## 5.14 Transfer Call Scenarios

### 5.14.1 Screened Transfer (with local view in Transferred event)

#### 5.14.1.1 Transferring party is a Non-SIP device

This service transfers a held party to a consulted party.

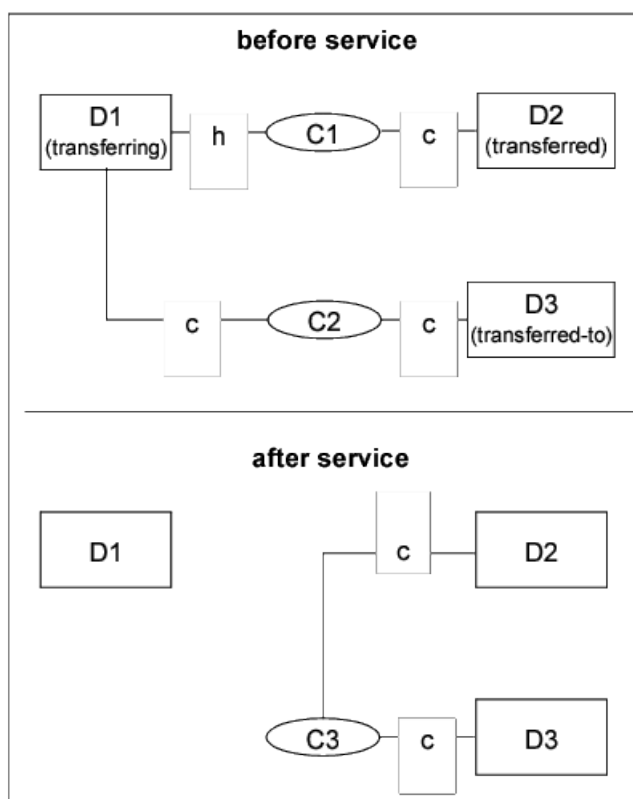


Figure 53: Screened transfer

## Call Scenarios

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

**Table 272: Screened Transfer**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Transfer Call service is invoked on behalf of device D1.	Transfer Call Request						
	• heldConnec- tion	D1C1					
	• activeCon- nection	D1C2					
1) Acknowl- edgement.	Transfer Call Response						
	• transferred- Connection	D3C3					
1) Calls be- tween D1, D2 and D1, D3 are re- leased. The connections between D2, D1 and D3, D1 are replaced with a single connection between D2 and D3.	Transferred		Transferred		Transferred		The CSTA Transferred event Local View model- ing option is provided by the switching function. This means that the primary old call parameters in the Trans- ferred events represent a device oriented view.
	• primaryOld- Call	D1C1	• primaryOld- Call	D2C1	• primaryOld- Call	D3C2	
	• second- aryOldCall	D1C2					
	• transfer- ringDevice	D1	• transfer- ringDevice	D1	• transfer- ringDevice	D1	
	• transferred- ToDevice	D3	• transferred- ToDevice	D3	• transferred- ToDevice	D3	
	• transferred- Conne- ctions 1. new / old 2. new / old	(D2C3) / (D2C1) (D3C3) / (D3C2)	• transferred- Conne- ctions 1. new / old 2. new	(D2C3) / (D2C1) (D3C3)	• transferred- Conne- ctions 1. new / old 2. new	(D3C3) / (D3C2) (D2C3)	
	• localConnec- tionInfo	null	• localConnec- tionInfo	connected	• localConnec- tionInfo	connected	
	• cause	Transfer	• cause	Transfer	• cause	Transfer	
	• servicesPer- mitted	none	• servicesPer- mitted	ClearConn, Consult, Hold, SST, GenDg, GenTelTone, SendUserInfo	• servicesPer- mitted	ClearConn, Consult, Hold, SST, GenDg, GenTelTone, SendUserInfo	

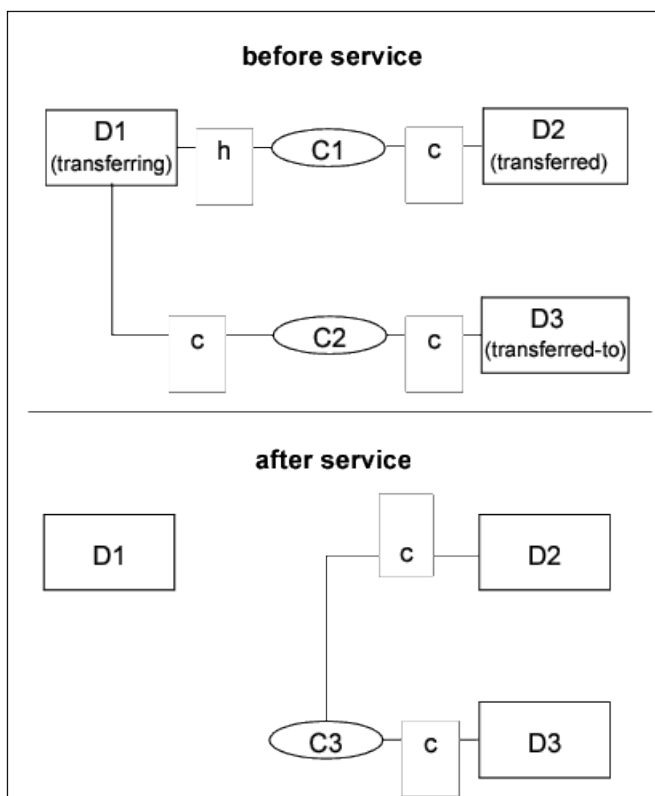
### Remark:

- The manual case is similar to the described event flow.
- In case of a transfer a new call Id (C3) will be generated that means the called device (D3) of this new call Id will not be the same as it was (D2) in the old call (C2).

- For more information about Transferred local/global view, see ECMA TR/82 Scenarios for Computer Supported Telecommunication Applications (CSTA) Phase III, (December 2000) document on page 65.

#### 5.14.1.2 Transferring party is a SIP device

This service transfers a held party to a consulted party.



**Figure 54: Creened transfer (transferring party is a SIP device)**

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

**Table 273: Screened Transfer, transferring party is SIP**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) SIP presses "Join". Calls between D1, D2 and D1, D3 are released. The connections between D2, D1 and D3, D1 are replaced with a single connection between D2 and D3.	Transferred		Transferred		Transferred		The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred events represent a device oriented view.
	• primaryOld-Call	D1C1	• primaryOld-Call	D2C1	• primaryOld-Call	D3C2	
	• secondaryOldCall	D1C2					
	• transferringDevice	D1	• transferringDevice	D1	• transferringDevice	D1	
	• transferred-ToDevice	D3	• transferred-ToDevice	D3	• transferred-ToDevice	D3	
	• transferred-Connections 1. new / old 2. new / old	(D2C3) / (D2C1) (D3C3) / (D3C2)	• transferred-Connections 1. new / old 2. new	(D2C3) / (D2C1) (D3C3)	• transferred-Connections 1. new / old 2. new	(D3C3) / (D3C2) (D2C3)	
	• localConnectionInfo	null	• localConnectionInfo	connected	• localConnectionInfo	connected	
• cause	Transfer	• cause	Transfer	• cause	Transfer		
	• servicesPermitted	none	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDg, GenTelTone, SendUserInfo	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDg, GenTelTone, SendUserInfo	
1) 4. D1 goes in blocked state	Failed						
	• failedConnection	D1C1					
	• failingDevice	D1					
	• callingDevice	D1					
	• calledDevice	D2					
	• lastRedirectionDevice	NS					
	• localConnectionInfo	fail					
• cause	blocked						

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	ClearConn		
	Connection Cleared			
	<ul style="list-style-type: none"> <li>droppedConnection</li> </ul>			
	<ul style="list-style-type: none"> <li>releasingDevice</li> </ul>			
	<ul style="list-style-type: none"> <li>localConnectionInfo</li> </ul>			
	<ul style="list-style-type: none"> <li>cause</li> </ul>	normalClr		
	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	none		

**Remark:**

- In case of a transfer a new call Id (C3) will be generated that means the called device (D3) of this new call Id will not be the same as it was (D2) in the old call (C2).
- For more information about Transferred local/global view, see ECMA TR/82 Scenarios for Computer Supported Telecommunication Applications (CSTA) Phase III, (December 2000) document on page 65.

## 5.14.2 Blind Transfer (with local view in Transferred event)

### 5.14.2.1 Transferring device is a Non-SIP device

This service transfers a held party to a consulted party. The transfer service request is issued before the consulted device connects into the new call.

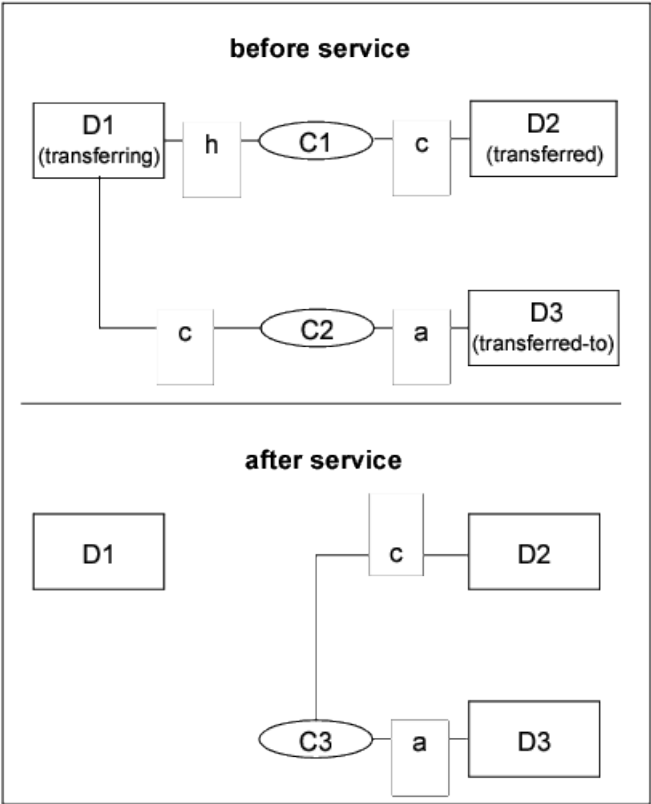


Figure 55: Blind transfer

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

Table 274: Blind Transfer

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3	Comments
1) Transfer Call service is invoked on behalf of device D1.	Transfer Call Request				
	• heldConnec- tion	D1C1			
	• activeCon- nection	D1C2			
1) Acknowl- edgement.	Transfer Call Response				
	• transferred- Connection	D3C3			

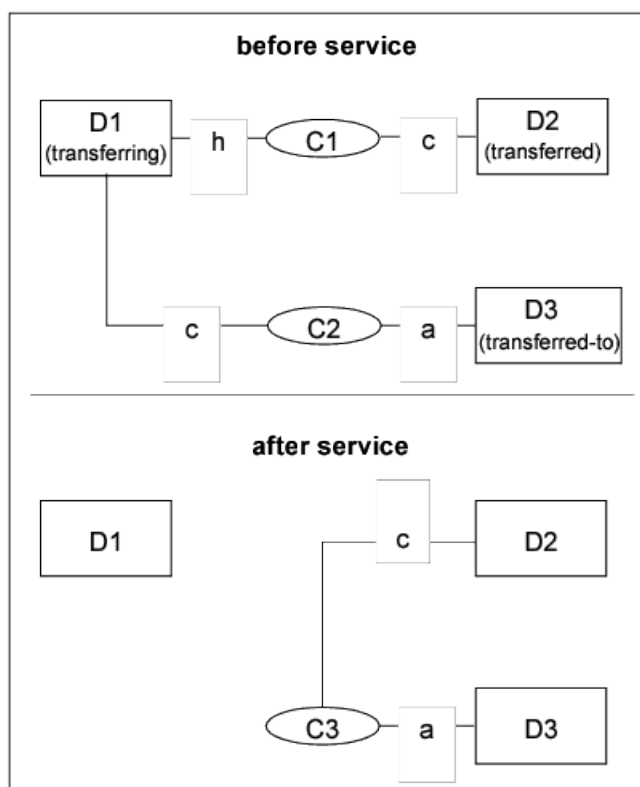
Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Calls between D1, D2 and D1, D3 are released. The connections between D2, D1 and D3, D1 are replaced with a single connection between D2 and D3.	Transferred		Transferred		Transferred		The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.
	• primaryOld-Call	D1C1	• primaryOld-Call	D2C1	• primaryOld-Call	D3C2	
	• secondaryOldCall	D1C2					
	• transferringDevice	D1	• transferringDevice	D1	• transferringDevice	D1	
	• transferred-ToDevice	D3	• transferred-ToDevice	D3	• transferred-ToDevice	D3	
	• transferred-Connections 1. new / old 2. new / old	(D2C3) / (D2C1) (D3C3) / (D3C2)	• transferred-Connections 1. new / old 2. new	(D2C3) / (D2C1) (D3C3)	• transferred-Connections 1. new / old 2. new	(D3C3) / (D3C2) (D2C3)	
	• localConnectionInfo	null	• localConnectionInfo	connected	• localConnectionInfo	alerting	
	• cause	Transfer	• cause	Transfer	• cause	Transfer	
	• servicesPermitted	none	• servicesPermitted	ClearConn, SendUserInfo	• servicesPermitted	Answer, ClearConn, SendUserInfo	

**Remark:**

- The manual case is similar to the described event flow.
- In case of a transfer a new call Id (C3) will be generated that means the called device (D3) of this new call Id will not be the same as it was (D2) in the old call (C2).
- For more information about Transferred local/global view, see ECMA TR/82 Scenarios for Computer Supported Telecommunication Applications (CSTA) Phase III, (December 2000) document on page 65.

**5.14.2.2 Transferring device is a SIP device**

The SIP device transfers a held party to a consulted party. The transfer is issued before the consulted device connects into the new call.



**Figure 56: Blind transfer on SIP device**

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

**Table 275: Blind Transfer on SIP device**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) SIP device presses "Join". Calls between D1, D2 and D1, D3 are released. The connections between D2, D1 and D3, D1 are replaced with a single connection between D2 and D3.	Transferred		Transferred		Transferred		The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.
	• primaryOld-Call	D1C1	• primaryOld-Call	D2C1	• primaryOld-Call	D3C2	
	• secondaryOldCall	D1C2					
	• transferringDevice	D1	• transferringDevice	D1	• transferringDevice	D1	
	• transferred-ToDevice	D3	• transferred-ToDevice	D3	• transferred-ToDevice	D3	
	• transferred-Connections 1. new / old 2. new / old	(D2C3) / (D2C1) (D3C3) / (D3C2)	• transferred-Connections 1. new / old 2. new	(D2C3) / (D2C1) (D3C3)	• transferred-Connections 1. new / old 2. new	(D3C3) / (D3C2) (D2C3)	

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
	<ul style="list-style-type: none"><li>localConnec- tionInfo</li></ul>		<ul style="list-style-type: none"><li>localConnec- tionInfo</li></ul>	connected	<ul style="list-style-type: none"><li>localConnec- tionInfo</li></ul>	alerting	
	<ul style="list-style-type: none"><li>cause</li></ul>	Transfer	<ul style="list-style-type: none"><li>cause</li></ul>	Transfer	<ul style="list-style-type: none"><li>cause</li></ul>	Transfer	
	<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	none	<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	ClearConn, SendUserInfo	<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	Answer, ClearConn, SendUserInfo	
	Failed						
	<ul style="list-style-type: none"><li>failedCon- nection</li></ul>	D1C1					
	<ul style="list-style-type: none"><li>failingDe- vice</li></ul>	D1					
	<ul style="list-style-type: none"><li>callingDe- vice</li></ul>	D1					
	<ul style="list-style-type: none"><li>calledDe- vice</li></ul>	D2					
	<ul style="list-style-type: none"><li>lastRedirec- tionDevice</li></ul>	NS					
	<ul style="list-style-type: none"><li>localConnec- fail tionInfo</li></ul>						
	<ul style="list-style-type: none"><li>cause</li></ul>	blocked					
	<ul style="list-style-type: none"><li>servicesPer- mitted</li></ul>	ClearConn					
	Connection Cleared						
	<ul style="list-style-type: none"><li>droppedCon- D1C1 nection</li></ul>						
	<ul style="list-style-type: none"><li>releasingDe- D1 vice</li></ul>						
	<ul style="list-style-type: none"><li>localConnec- null tionInfo</li></ul>						
	<ul style="list-style-type: none"><li>cause</li></ul>	normalClr					
	<ul style="list-style-type: none"><li>servicesPer- none mitted</li></ul>						

**Remark:**

- In case of a transfer a new call Id (C3) will be generated that means the called device (D3) of this new call Id will not be the same as it was (D2) in the old call (C2).
- For more information about Transferred local/global view, see ECMA TR/82 Scenarios for Computer Supported Telecommunication Applications (CSTA) Phase III, (December 2000) document on page 65.

5.14.3 Transfer to a busy station (with local view in Transferred event)

This service transfers a held party to a busy party. The transferred party camps-on to the transferred-to party.

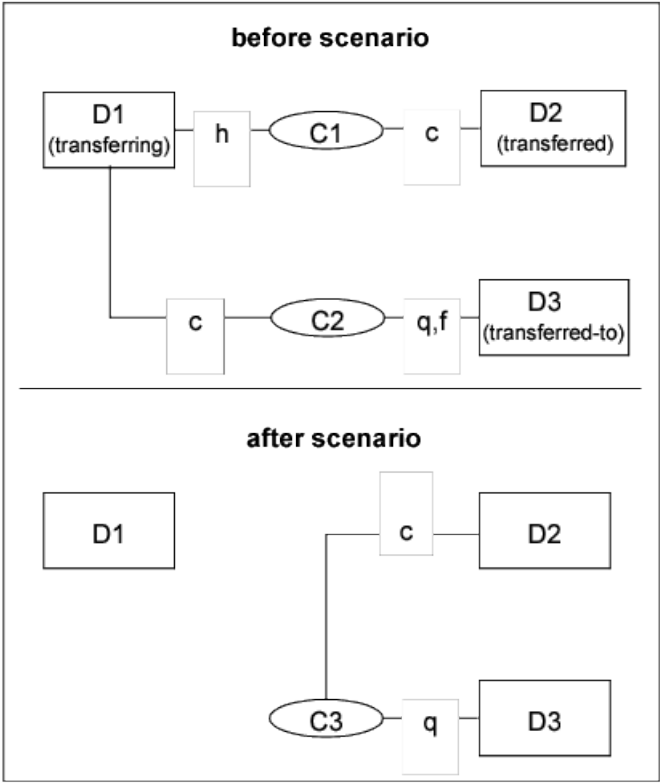


Figure 57: Transfer to a busy station

Table 276: Transfer to a busy station

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3		Comments
1) The switch automatically clears the failed connection.	Connection Cleared			Connection Cleared		
	•	droppedCon-D3C2 nection		•	droppedCon-D3C2 nection	
	•	releasingDe- D3 vice		•	releasingDe- D3 vice	
	•	localConnec-connected tionInfo		•	localConnec-null tionInfo	
	•	cause normalClr		•	cause normalClr	
	•	servicesPer- ClearConn mitted		•	servicesPer- none mitted	

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Transfer Call service is invoked on behalf of device D1.	Transfer Call Request						
	• heldConnec- tion	D1C1					
	• activeCon- nection	D1C2					
1) Acknowl- edgement.	Transfer Call Response						
	• transferred- Connection	D3C3					
1) Calls be- tween D1, D2 and D1, D3 are re- leased. The connections between D2, D1 and D3, D1 are replaced with a single connection between D2 and D3.	Transferred		Transferred		Transferred		The CSTA Transferred event Local View model- ing option is provided by the switching function. This means that the primary old call parameters in the Trans- ferred event represent a device oriented view.
	• primaryOld- Call	D1C1	• primaryOld- Call	D2C1	• primaryOld- Call	D3C2	
	• second- aryOldCall	D2C2					
	• transfer- ringDevice	D1	• transfer- ringDevice	D1	• transfer- ringDevice	D1	
	• transferred- ToDevice	D3	• transferred- ToDevice	D3	• transferred- ToDevice	D3	
	• transferred- Connec- tions 1. new / old 2. new / old	(D2C3) / (D2C1) (D3C3) / (D3C2)	• transferred- Connec- tions 1. new / old 2. new	(D2C3) / (D2C1) (D3C3)	• transferred- Connec- tions 1. new / old 2. new	(D3C3) / (D3C2) (D2C3)	
	• localConnec- tionInfo	null	• localConnec- tionInfo	connected	• localConnec- tionInfo	fail	
	• cause	campOn	• cause	normal	• cause	normal	
	• servicesPer- mitted	none	• servicesPer- mitted	CallBack, ClearConn, SendUserInfo	• servicesPer- mitted	SendUserInfo	
1) D2 camps on D3.			Queued		Queued		
			• queuedCon- nection	D3C3	• queuedCon- nection	D3C3	
			• queue	D3	• queue	D3	
			• callingDe- vice	D2	• callingDe- vice	D2	
			• calledDe- vice	D3	• calledDe- vice	D3	
			• lastRedirec- tionDevice	NS	• lastRedirec- tionDevice	NS	

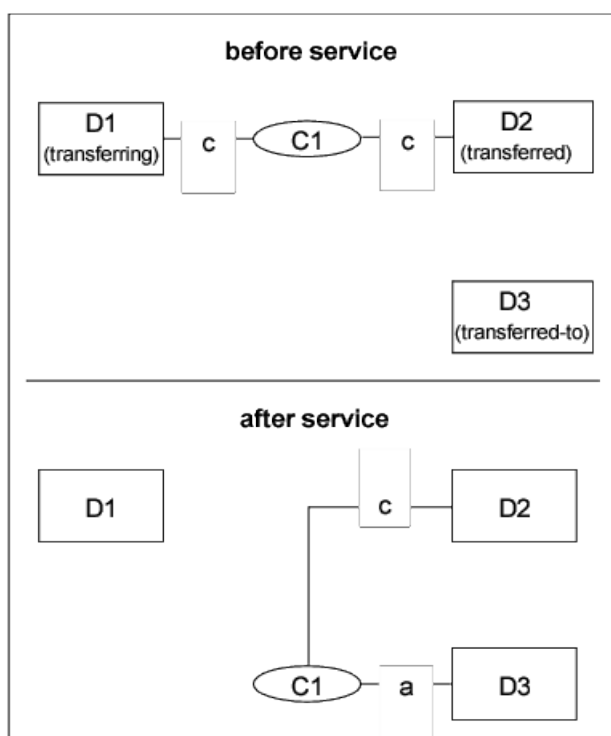
Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
		<ul style="list-style-type: none"> <li>localConnec- tionInfo</li> </ul>	<ul style="list-style-type: none"> <li>localConnec- queued tionInfo</li> </ul>	
		<ul style="list-style-type: none"> <li>cause</li> </ul>	<ul style="list-style-type: none"> <li>campOn</li> </ul>	<ul style="list-style-type: none"> <li>cause</li> </ul>
		<ul style="list-style-type: none"> <li>servicesPer- mitted</li> </ul>	<ul style="list-style-type: none"> <li>CallBack, ClearConn, SendUserInfo</li> </ul>	<ul style="list-style-type: none"> <li>SendUserInfo</li> </ul>

## Remark:

- The manual case is similar to the described event flow.
- In case of a transfer a new call Id (C3) will be generated that means the called device (D3) of this new call Id will not be the same as it was (D2) in the old call (C2).
- For more information about Transferred local/global view, see ECMA TR/82 Scenarios for Computer Supported Telecommunication Applications (CSTA) Phase III, (December 2000) document on page 65.

## 5.14.4 Single Step Transfer (with local view in Transferred event)

This service transfers a device in one step.



**Figure 58: Single step transfer (devices)**

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before service" state.

Table 277: Single Step Transfer (Devices)

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Single Step Transfer Call service is invoked on behalf of device D1.	Single Step Transfer Call Request						
	• activeCon- nection	D1C1					
	• transferred- To	D3					
1) Acknowl- edgement.	Single Step Transfer Call Re- sponse						
	• transferred- Connection	D3C1					
1) The call between D1 and D2 is replaced with an alerting call between D2 and D3.	Transferred		Transferred				The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.  Note that the switching function will not provide a new call id.
	• primaryOld- Call	D1C1	• primaryOld- Call	D2C1			
	• transfer- ringDevice	D1	• transfer- ringDevice	D1			
	• transferred- ToDevice	D3	• transferred- ToDevice	D3			
	• transferred- Connec- tions 1. new 2. new	D2C1 D3C1	• transferred- Connec- tions 1. new: 2. new	D2C1 D3C1			
	• localConnec- tionInfo	null	• localConnec- tionInfo	connected			
	• cause	SST	• cause	SST			
	• servicesPer- mitted	none	• servicesPer- mitted	ClearConn, SendUserInfo			
1) The call alerts de- vice D3.			Delivered		Delivered		This event re- flects the con- nection state change at D3C1.
			• connection	D3C1	• connection	D3C1	
			• alertingDe- vice	D3	• alertingDe- vice	D3	
			• callingDe- vice	D2	• callingDe- vice	D2	
			• calledDe- vice	D3	• calledDe- vice	D3	
			• lastRedirec- tionDevice	NS	• lastRedirec- tionDevice	NS	

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
		• localConnec- tionInfo	• localConnec- tionInfo	
		• cause	• cause	
		• servicesPer- mitted	• servicesPer- mitted	
		connected	alerting	
		SST	SST	
		ClearConn, SendUserInfo	Answer, ClearConn, Deflect, Sen- dUserInfo	

**Remark:**

The swithing function does not allocate a new callID in case of a Single Step Transfer.

5.14.5 Single Step Transfer between network interface devices (with local view in Transferred event)

This scenario illustrates a successful Single Step Transfer Service. The transferred and transferred-to devices are network interface devices (NIDs) .

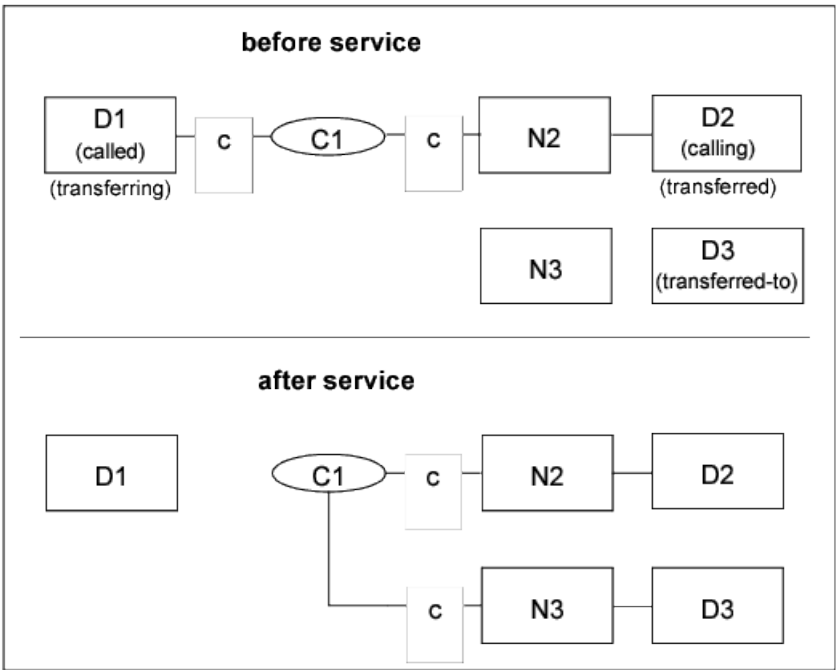


Figure 59: Single step transfer (trunk to trunk)

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before service" state.

Table 278: Single Step Transfer (Trunk to Trunk)

Activity	Monitored Device D1		Monitored Device N2		Monitored Device N3		Comments
1) Single Step Transfer Call service is invoked on behalf of device D1.	Single Step Transfer Call Request						
	• activeCon- nection	D1C1					
	• transferred- To	D3					
1) Ac- knowledge- ment.	Transfer Call Response						
	• transferred- Conn	D3C1					
1) The net- work is reached again.	Network Reached		Network Reached		Network Reached		
	• outbound- Connection	N3C1	• outbound- Connection	N3C1	• outbound- Connection	N3C1	
	• networkInter- faceUsed	N3	• networkInter- faceUsed	N3	• networkInter- faceUsed	N3	
	• callingDe- vice	D2	• callingDe- vice	D2	• callingDe- vice	D2	
	• calledDe- vice	D3	• calledDe- vice	D3	• calledDe- vice	D3	
	• lastRedirec- tionDevice	NS	• lastRedirec- tionDevice	NS	• lastRedirec- tionDevice	NS	
	• localConnec- tionInfo	connected	• localConnec- tionInfo	connected	• localConnec- tionInfo	connected	
	• cause	normal	• cause	normal	• cause	normal	
	• servicesPer- mitted	none	• servicesPer- mitted	ClearConn, SendUserInfo	• servicesPer- mitted	ClearConn, Deflect, Sen- dUserInfo	
			• networkCall- ingDevice	D2	• networkCall- ingDevice	D2	
		• assocCall- ingDevice	N2	• assocCall- ingDevice	N2		

## Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Monitored Device N3		Comments
1) Device D2 transfers.	Transferred		Transferred				The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.  Note that the switching function will not provide a new call id.
	• primaryOld-Call	D1C1	• primaryOld-Call	N2C1			
	• transferringDevice	D1	• transferringDevice	D1			
	• transferred-ToDevice	D3	• transferred-ToDevice	D3			
	• transferred-Connections 1. new:as-sNID:endp 2. new:as-sNID:endp	(N2C1):N2:D2 (N3C1):N3:D3	• transferred-Connections 1. new/old:as-sNID:endp 2. new:as-sNID:endp	(N2C1)/ (N2C1):N2:D2 (N3C1):N3:D3			
	• localConnectionInfo	null	• localConnectionInfo	connected			
	• cause	SST	• cause	SST			
	• servicesPermitted	none	• servicesPermitted	ClearConn, SendUserInfo			
The call alerts device D3.			Delivered		Delivered		This event reflects the connection state change at N3C1.
			• connection	N3C1	• connection	N3C1	
			• alertingDevice	D3	• alertingDevice	D3	
			• callingDevice	D2	• callingDevice	D2	
			• calledDevice	D3	• calledDevice	D3	
			• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
			• origNID	N2C1	• origNID	N2C1	
			• localConnectionInfo	connected	• localConnectionInfo	alert	
			• cause	SST	• cause	SST	
			• servicesPermitted	ClearConn, SendUserInfo	• servicesPermitted	ClearConn, Deflect, SendUserInfo	
				• networkCallingDevice	D2	• networkCallingDevice	

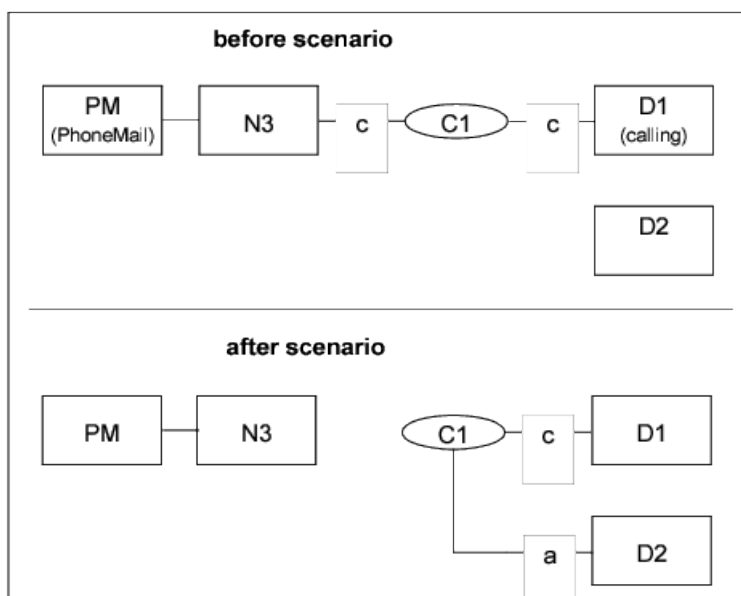
Activity	Monitored Device D1	Monitored Device N2	Monitored Device N3	Comments
		<ul style="list-style-type: none"> <li>• assocCallingDevice N2</li> </ul>	<ul style="list-style-type: none"> <li>• assocCallingDevice N2</li> </ul>	
		<ul style="list-style-type: none"> <li>• assocCalled-N3 Device</li> </ul>	<ul style="list-style-type: none"> <li>• assocCalled-N3 Device</li> </ul>	

**Remark:**

The swithing function does not allocate a new callID in case of a Single Step Transfer.

### 5.14.6 Single Step Call Transfer, Phone Mail transfers

The scenario describes an event flow when a Phone Mail device transfers in one step.



**Figure 60: Single step transfer from Phone Mail to agent**

See [Section 5.17.1.3, "Internal ACD call completed to Phone Mail agent"](#) for the event flow to get into the "before scenario" state.

**Table 279: Single Step Transfer from Phone Mail to Agent**

Activity	Monitored Device D1		Monitored Device N3		Monitored Device D2		Comments
1) The call between D1 and N3 is replaced with an alerting call between D1 and D2.	Transferred		Transferred				The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.
	• primaryOld-Call	D1C1	• primaryOld-Call	N3C1			
	• transferringDevice	PM	• transferringDevice	PM			
	• transferred-ToDevice	D2	• transferred-ToDevice	D2			
	• transferred-Connections 1.new 2.new	(D1C1) (D2C1)	• transferred-Connections 1.new 2.new	(D1C1) (D2C1)			
	• localConnectionInfo	connected	• localConnectionInfo	null			
	• cause	SST	• cause	SST			
1) The call alerts D2.	Delivered				Delivered		These events reflect the connection state change at D2C1.
	• connection	D2C1			• connection	D2C1	
	• alertingDevice	D2			• alertingDevice	D2	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	
	• localConnectionInfo	connected			• localConnectionInfo	alerting	
• cause	SST	• cause	SST				
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	none			

Activity	Monitored Device D1	Monitored Device N3	Monitored Device D2	Comments
1) Phone Mail goes into blocked state.		Failed		
		• failedCon- nection	N3C2	
		• failingDe- vice	N3	
		• callingDe- vice	NK	
		• calledDe- vice	NK	
		• lastRedirec- tionDevice	NS	
		• localConnec- tionInfo	fail	
		• cause	blocked	
1) N3C3 is dropped.		Connection Cleared		
		• droppedCon- nection	N3C2	
		• releasingDe- vice	N3	
		• localConnec- tionInfo	null	
		• cause	normalClr	
		• servicesPer- mitted	none	

**Remark:**

None

### 5.14.7 Single Step Transfer to destination with call forward immediate handled in the switching subdomain (D3 is internal analogue or digital device)

This service transfers and forwards a connection in one step.

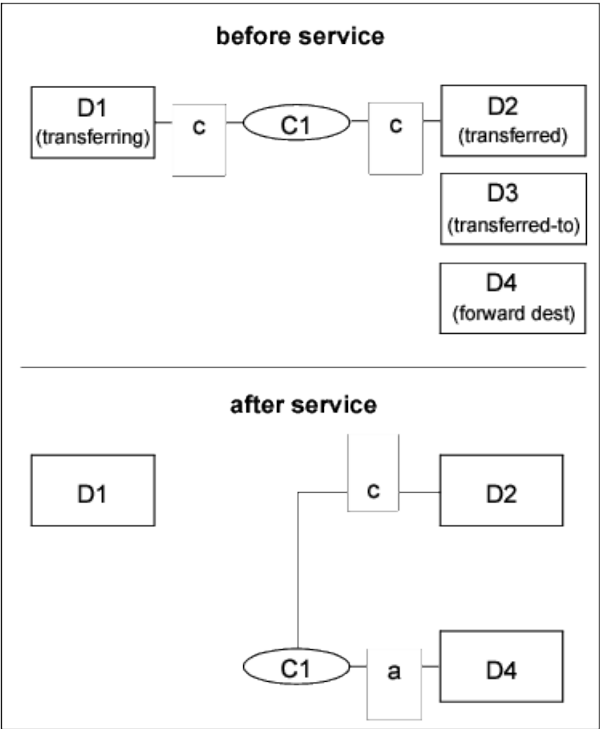


Figure 61: Single step transfer with call forward (devices)

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before service" state.

Table 280: Single Step Transfer with call forward (Devices)

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D4	Comments
1) Single Step Transfer Call service is invoked on behalf of device D1.	Single Step Transfer Call Request			
	<ul style="list-style-type: none"><li>activeCon- nection</li></ul> D1C1			
	<ul style="list-style-type: none"><li>transferred- To</li></ul> D3			
1) Acknowl- edgement.	Single Step Transfer Call Re- sponse			
	<ul style="list-style-type: none"><li>transferred- Connection</li></ul> D4C1			

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D4		Comments
1) The call between D1 and D2 is replaced with an alerting call between D2 and D3.	Transferred		Transferred				The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.  Note that the switching function will not provide a new call id.
	• primaryOld-Call	D1C1	• primaryOld-Call	D2C1			
	• transferringDevice	D1	• transferringDevice	D1			
	• transferred-ToDevice	D4	• transferred-ToDevice	D4			
	• transferred-Connections 1. new 2. new	D2C1 D4C1	• transferred-Connections 1. new: 2. new	D2C1 D4C1			
	• localConnectionInfo	null	• localConnectionInfo	connected			
	• cause	SST	• cause	SST			
	• servicesPermitted	none	• servicesPermitted	ClearConn, SendUserInfo			
1) The call alerts device D3.			Delivered		Delivered		This event reflects the connection state change at D4C1.
			• connection	D4C1	• connection	D4C1	
			• alertingDevice	D4	• alertingDevice	D4	
			• callingDevice	D2	• callingDevice	D2	
			• calledDevice	D4	• calledDevice	D4	
			• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
			• localConnectionInfo	connected	• localConnectionInfo	alerting	
			• cause	SST	• cause	SST	
			• servicesPermitted	ClearConn, SendUserInfo	• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	

**NOTICE:** Please note that the forward can only be followed by the change in the destination.

5.14.8 Single Step Transfer attempt to busy destination with offered mode activated

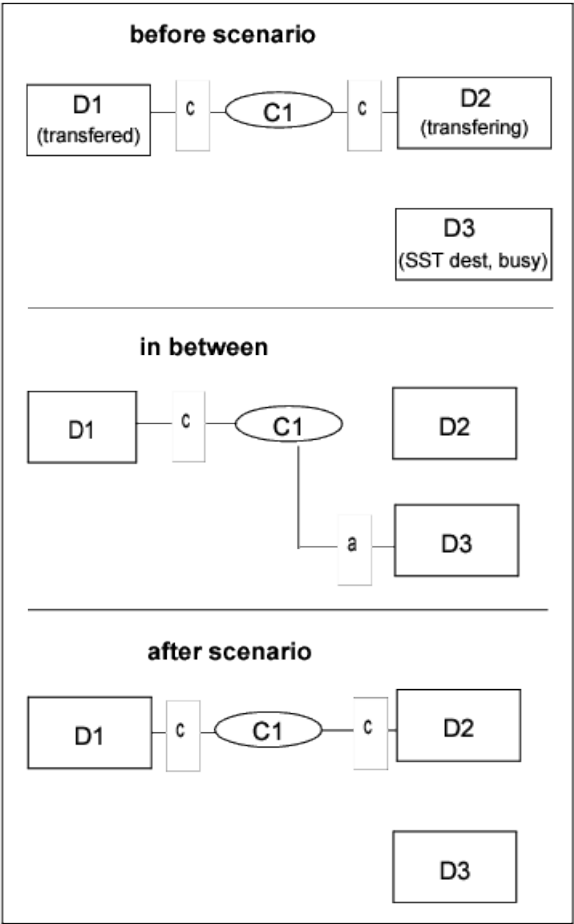


Figure 62: Single step transfer attempt to busy destination without offered mode activated

See [Section 5.5.1, "Successful answer call"](#) for the event flow to get into the "before service" state.

Table 281: Single Step Transfer attempt to busy destination with offered mode activated

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) Single Step Transfer Call service is invoked on behalf of device D1.	Single Step Transfer Call Request			
	<ul style="list-style-type: none"><li>activeCon- nection</li><li>transferred- To</li></ul>	D2C1  D3		
1) Acknowl- edgement.	Single Step Transfer Call Re- sponse			

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
	<ul style="list-style-type: none"><li>transferred-Connection</li></ul>	D3C1					
1) The call between D1 and D2 is replaced with an alerting call between D2 and D3.	Transferred		Transferred				The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.  Note that the switching function will not provide a new call id.
	<ul style="list-style-type: none"><li>primaryOld-Call</li></ul>	D1C1	<ul style="list-style-type: none"><li>primaryOld-Call</li></ul>	D2C1			
	<ul style="list-style-type: none"><li>transfer-ringDevice</li></ul>	D2	<ul style="list-style-type: none"><li>transfer-ringDevice</li></ul>	D2			
	<ul style="list-style-type: none"><li>transferred-ToDevice</li></ul>	D3	<ul style="list-style-type: none"><li>transferred-ToDevice</li></ul>	D3			
	<ul style="list-style-type: none"><li>transferred-Connections 1. new 2. new</li></ul>	D1C1 D3C1	<ul style="list-style-type: none"><li>transferred-Connections 1. new: 2. new</li></ul>	D1C1 D3C1			
	<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	connected	<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	null			
	<ul style="list-style-type: none"><li>cause</li></ul>	SST	<ul style="list-style-type: none"><li>cause</li></ul>	SST			
1) The call is offered to D3.	Offered (optional)				Offered		This event reflects the connection state change at D4C1.
	<ul style="list-style-type: none"><li>connection</li></ul>	D3C1			<ul style="list-style-type: none"><li>connection</li></ul>	D3C1	
	<ul style="list-style-type: none"><li>offeredDevice</li></ul>	D3			<ul style="list-style-type: none"><li>offeredDevice</li></ul>	D3	
	<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1			<ul style="list-style-type: none"><li>callingDevice</li></ul>	D1	
	<ul style="list-style-type: none"><li>calledDevice</li></ul>	D3			<ul style="list-style-type: none"><li>calledDevice</li></ul>	D3	
	<ul style="list-style-type: none"><li>lastRedirectionDevice</li></ul>	D2			<ul style="list-style-type: none"><li>lastRedirectionDevice</li></ul>	D2	
	<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	connected			<ul style="list-style-type: none"><li>localConnectionInfo</li></ul>	alerting	
<ul style="list-style-type: none"><li>cause</li></ul>	SST	<ul style="list-style-type: none"><li>cause</li></ul>	SST	<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	ClearConn, SendUserInfo	<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	Answer, ClearConn, Deflect, SendUserInfo

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2	Monitored Device D3		Comments
1) Application accepts the call				Accept Call IRequest • callToBeAccepted	D3C1	
1) Acknowledged				Accept Call Response		
1) Destination is busy	Failed			Failed		
	• failedConnection	D3C1		• failedConnection	D3C1	
	• failingDevice	D3		• failingDevice	D3	
	• callingDevice	D1		• callingDevice	D1	
	• calledDevice	D3		• calledDevice	D3	
	• lastRedirectionDevice	NS		• lastRedirectionDevice	NS	
	• localConnectionInfo	connected		• localConnectionInfo	null	
	• cause	busy		• cause	busy	
	• servicesPermitted	ClearConn		• servicesPermitted	none	
1) Recall D2	Diverted (optional)			Diverted		
	• connection	D3C1		• connection	D3C1	
	• divertingDevice	D3		• divertingDevice	D3	
	• newDestination	D2		• newDestination	D2	
	• Calling	D1		• Calling	D1	
	• calledDevice	D3		• calledDevice	D3	
	• lastRedirectionDevice	NS		• lastRedirectionDevice	NS	
	• localConnectionInfo	connected		• localConnectionInfo	null	
	• cause	recallBusy		• cause	recallBusy	
	• servicesPermitted	none		• servicesPermitted	none	

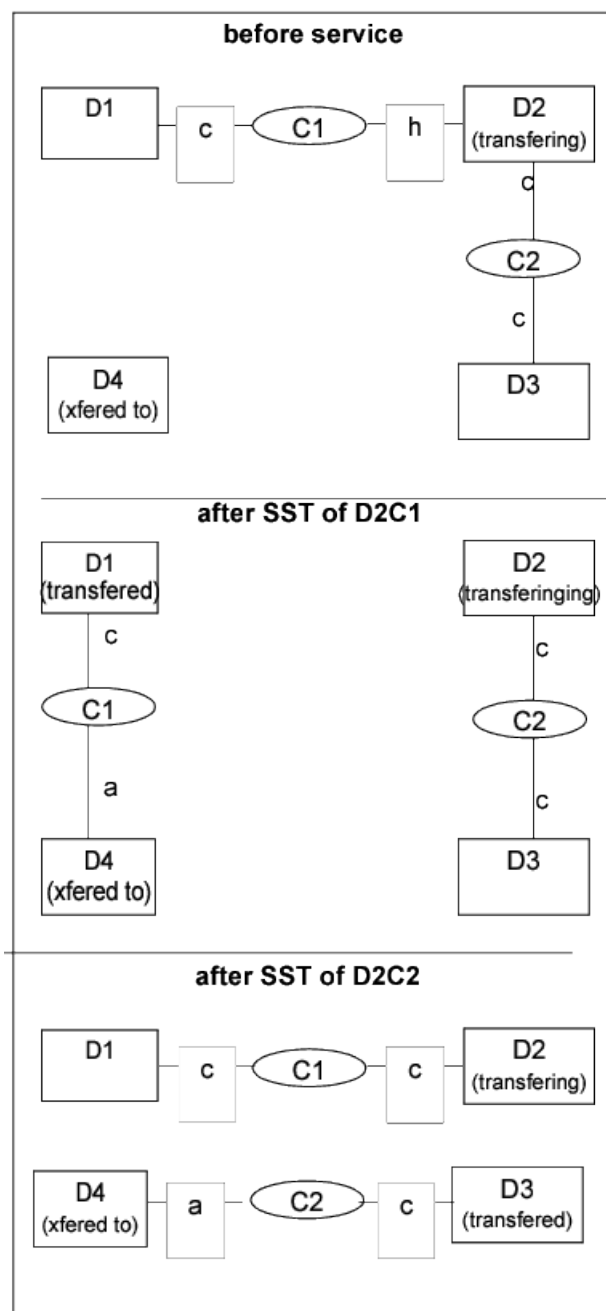
Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	Comments
1) Call re-connected to	Established		Established			
	• established-Connection	D2C1	• established-Connection	D2C1		
	• answerind-Device	D2	• answerind-Device	D2		
	• callingDe-vice	D1	• callingDe-vice	D1		
	• calledDe-vice	D2	• calledDe-vice	D2		
	• lastRedirec-tionDevice	D3	• lastRedirec-tionDevice	D3		
	• localConnec-tionInfo	connected	• localConnec-tionInfo	connected		
	• cause	recallBusy	• cause	recallBusy		
	• servicesPer-mitted	ClearConn, Consult, Hold, SST, GenDg, GenTelTone, SendUserInfo	• servicesPer-mitted	ClearConn, Consult, Hold, SST, GenDg, GenTelTone, SendUserInfo		

---

**NOTICE:** This event flow can occur in cases where the Offered mode is activated on the destination and the ONS monitoring is activated. See Service Manual for details.

---

### 5.14.9 Single Step Transfer for Consulting Party



Please note that only one of the mentioned SST requests can be executed: either SST of D2C1 or SST of D2C2. The Transferred and Delivered events are provided on the same way as for the "classical" single step transfer, see 5.14.4

### 5.14.10 Transfer on Remote OpenScape 4000

Since Version 7 OpenScape 4000 is able to provide CSTA events about a transfer happening on the partner OpenScape 4000. Basically the event flow is the same as in case of the local transfer, with the restriction that there is no change in the Call ID and there is no way to recognize Single Step Transfer.

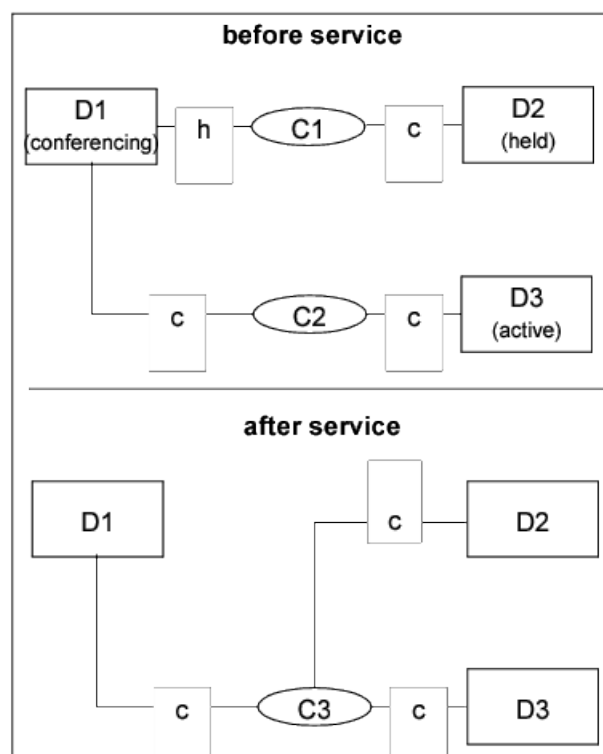
To have this functionality, switch on the mapping of the remote features, see Service Manual for details. Without the configuration or in case of any missing information, the system falls back to the original behaviour, which is to provide a CallInformationEvent showing the change in the CallLinkageData due to the change of the remote partner.

## 5.15 Conference Call Scenarios

### 5.15.1 Conference (with local view in Conferenced event)

#### 5.15.1.1 Conference master is a Non-SIP device

This 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 at the conferencing device.



**Figure 63: Conference call - Master: non-SIP device**

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

**Table 282: Conference**

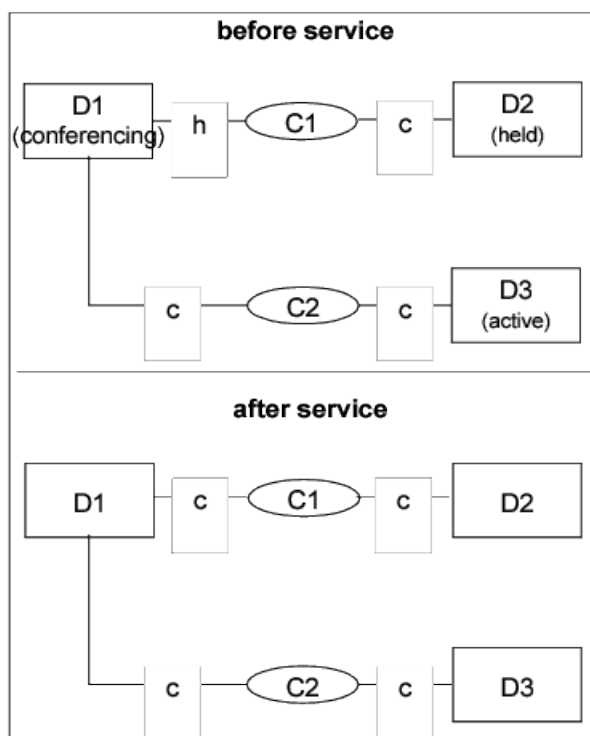
Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Conference Call service is requested on behalf of device D1.	Conference Request						
	• heldConnection	D1C1					
	• activeConnection	D1C2					
1) Acknowledgement.	Conference Response						
	• conferencedConnection	D1C3					
1) Conferencing established.	Conferenced		Conferenced		Conferenced		The added-Party specifies the device ID of the device, that belongs to the active (not held) call of the conference.  Note that the primaryOld-Call and the secondaryOldCall parameters follows the "local view" modeling option.
	• primaryOld-Call	D1C1	• primaryOld-Call	D2C1	• primaryOldCall	D3C2	
	• secondaryOld-Call	D1C2					
	• conferencing-Device	D1	• conferencing-Device	D1	• conferencingDevice	D1	
	• Added	D3	• Added	D3	• Added	D3	
	• conferenceConnections 1. new/old 2. new/old 3. new/old 4. new	(D1C3)/ (D1C1) (D1C3)/ (D1C2) (D2C3) (D3C3)	• conferenceConnections 1. new/old 2. new/old 3. new	(D2C3)/ (D2C1) (D1C3)/ (D1C1) (D3C3)	• conferenceConnections 1. new/old 2. new/old 3. new	(D1C3)/ (D1C2) (D3C3)/ (D3C2) (D2C3)	
	• localConnectionInfo	connected	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	normal	• cause	normal	• cause	normal	
	• servicesPermitted	Clear-Conn, Consult, Hold, SendUserInfo	• servicesPermitted	Clear-Conn, Consult, Hold, SendUserInfo	• servicesPermitted	ClearConn, Consult, Hold, SendUserInfo	

**Remark:**

The manual case is similar to the described event flow.

## 5.15.1.2 Conference master is a SIP device

This service provides a conference of his existing 2 active calls at a conferencing device. The held call will be retrieved.



**Figure 64: Conference call - Master: SIP device**

See [Section 5.13.1, "Successful Consultation Call"](#), on page 434 for the event flow to get into the "before service" state.

**Table 283: Conference**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Conference button is pushed on device D1.	• heldConnection	D1C1					
	• activeConnection	D1C2					
1) Conference established.	Retrieved		Retrieved				
	• retrievedConnection	D1C1	• retrievedConnection	D1C1			
	• Retrieving	D1	• Retrieving	D1			
	• localConnectionInfo	connecting	• localConnectionInfo	connecting			
	• cause	normal	• cause	normal			
	• servicesPermitted	ClearConn, SingleStepTransfer (from HP4k V6 only!)	• servicesPermitted	ClearConn, Consult, Hold, SingleStepTransfer, GenerateDigits, SendUserInfo			

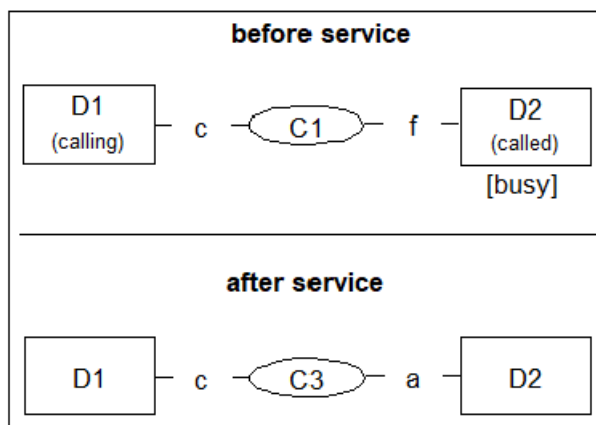
**Remark:**

SIP device can't add more conference member.

## 5.16 Call Completion Scenarios

### 5.16.1 Call Back Call Related

This scenario illustrates the use of the Call Back Call Related service where the called device is busy.



**Figure 65: Call completion - Call back call related**

See [Section 5.4.2, "Manually dialled call - Called party is busy"](#) for the event flow to get into the "before service" state.

**Table 284: Call Back Call Related**

Activity	Monitored Device D1		Monitored Device D2		Comments
1) The busy connection is cleared immediately.	Connection Cleared		Connection Cleared		
	• droppedConnection	D2C1	• droppedConnection	D2C1	
	• releasingDevice	D2	• releasingDevice	D2	
	• localConnectionInfo	connected	• localConnectionInfo	null	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	ClearConn	• servicesPermitted	none	
1) The Call Back Call Related service is invoked on behalf of device D1.	CallBack Request				
	• connection	D1C1			
1) Acknowledgement.	CallBack Response				
	• targetDevice	D2			
1) Device D1 is blocked.	Failed				

Activity	Monitored Device D1		Monitored Device D2		Comments
	• failedConnection	D1C1			
	• failingDevice	D1			
	• callingDevice	D1			
	• calledDevice	D2			
	• lastRedirectionDevice	NS			
	• localConnectionInfo	fail			
	• cause	blocked			
	• servicesPermitted	ClearConn			
1) Device D1 clears its failed connection.	Connection Cleared				
	• droppedConnection	D1C1			
	• releasingDevice	D1			
	• localConnectionInfo	null			
	• cause	normalClr			
	• servicesPermitted	none			
1) Device D2 sometime later clears from its active call.			Connection Cleared		C2 is the active call of D2.
			• droppedConnection	D2C2	
			• releasingDevice	D2	
			• localConnectionInfo	null	
			• cause	normalClr	
			• servicesPermitted	none	
1) Since device D2 is now available, the CallBack is initiated from device D1. D1 is being prompted to go off-hook.	Service Initiated				The cause code of Call-Back indicates that the device is being prompted to go off-hook.
	• initiatedConnection	D1C3			
	• initiatingDevice	D1			
	• localConnectionInfo	initiated			
	• cause	CallBack			
	• servicesPermitted	Answer, Clear-Conn, Deflect, SendUserInfo			
1) The switching function reserves the CallBack destination (D2).			Failed		
			• failedConnection	D2C3	
			• failingDevice	D2	

## Call Scenarios

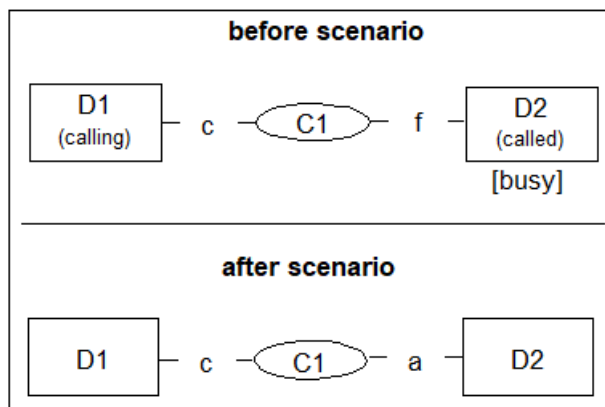
Activity	Monitored Device D1		Monitored Device D2		Comments
			• callingDevice	D1	
			• calledDevice	D2	
			• lastRedirectionDevice	NS	
			• localConnectionInfo	fail	
			• cause	blocked	
			• servicesPermitted	ClearConn	
1) Device D1 goes off hook and is connected on the call.	Originated				
	• originatedConnection	D1C3			
	• callingDevice	D1			
	• calledDevice	D2			
	• localConnectionInfo	connected			
	• cause	CallBack			
1) The failed connection of D2 is cleared.			Connection Cleared		
			• droppedConnection	D2C3	
			• releasingDevice	D2	
			• localConnectionInfo	null	
			• cause	normalClr	
			• servicesPermitted	none	
1) Device D2 is alerted.	Delivered		Delivered		
	• deliveredConnection	D2C3	• deliveredConnection	D2C3	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	alerting	
	• cause	CallBack	• cause	CallBack	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	ClearConn, Answer, Deflect, SendUser-Info	

**Remark:**

None

## 5.16.2 Manual Camp On Call

The calling party queues a call for a busy called device by pressing the camp-on key until that device becomes available.

**Figure 66: Call completion - Manual camp on call**

See [Section 5.4.2, "Manually dialled call - Called party is busy"](#) for the event flow to get into the "before scenario" state.

**Table 285: Manual Camp On Call**

Activity	Monitored Device D1		Monitored Device D2		Comments
1) The busy connection is cleared immediately.	Connection Cleared		Connection Cleared		
	• droppedConnection	D2C1	• droppedConnection	D2C1	
	• releasingDevice	D2	• releasingDevice	D2	
	• localConnectionInfo	connected	• localConnectionInfo	null	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	ClearConn	• servicesPermitted	none	
1) D1 presses the camp on key.	Queued		Queued		
	• queuedConnection	D2C1	• queuedConnection	D2C1	
	• queue	D2	• queue	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	queued	

## Call Scenarios

### Distribution Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Comments
	• cause	campOn	• cause	campOn	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	SendUserInfo	
1) Some time later device D2 clears from its active call.			Connection Cleared		
			• droppedConnection	D2C2	
			• releasingDevice	D2	
			• localConnectionInfo	null	
			• cause	normalClr	
			• servicesPermitted	none	
1) Since device D2 is available the call alerts D2.	Delivered		Delivered		
	• deliveredConnection	D2C1	• deliveredConnection	D2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	alerting	
	• cause	recall	• cause	recall	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	ClearConn, Answer, De- flect, SendU- serInfo	

**Remark:**

None

## 5.17 Distribution Call Scenarios

### 5.17.1 Automatic Call Distribution Scenarios

### 5.17.1.1 Automatic Call Distribution Overview

#### ACD Call Processing

ACD Call Processing is based on a call analysis and routing scheme that processes calls according to customer requirements. The routing scheme is configured by defining ACD numbers, route control groups (RCGs), and ACD routing tables (ARTs) to process the call.

The sequence of call processing begins with **"host-based" source- and destination-based routing**, continues with **calendar routing**, and is finally processed by the sequence of commands in the routing tables (**route processing**). Further routing may be necessary at the end of work shifts with **end of shift routing**. For an illustration of this call process, see [Figure 69 on page 491](#).

- "Host-based" Routing
- When a call arrives at the OpenScape 4000 destined for a call center, an external application may be used to route the call using the caller's identity, location, or the reason for the call. The application can take advantage of previous calls and place this call to an agent who has handled this customer previously. The application may route a caller to a different call center based on call volumes. The application could reside on a host or external server.
- Source- and Destination-based Routing
- Without special handling, when a call is placed to an ACD number, routing analysis begins when the ACD software determines the source and destination of the call. The source of the call is the number of the calling party received by the OpenScape 4000. It may be a 10-digit telephone number, the main number of the private branch exchange (PBX), or a private network number. Automatic Number Identification (ANI) is an example of a trunk service that provides a source number.

The destination is the called party. It may be a translated Dialed Number Identification Service (DNIS), Direct Inward Dialling (DID), or in some cases, something other than the original number dialed by the calling party, converted by public or private network number translation. ACD uses this source and destination data to associate a route control group (RCG) with the call. The RCG defines routing to an ACD routing table, based on the time of day and the day of the week the call is processed.

- Calendar Routing
- Calendar routing uses the day of the week and the time of day to reference the RCG and to select the ACD routing table (ART) for processing the call. Calendar Routing also can be used to setup holidays in advance.
- Route Processing
- The ACD routing table (ART) is used for final processing of the call. ACD routing tables contain a series of steps for processing the call. Each step in an ART is performed sequentially, except when a conditional step or a "GOTO" step directs the call to a specific step number. If this occurs, processing continues sequentially for the step defined in the GOTO statement. ARTs can be configured with two types of steps, fixed and conditional. Fixed steps route the call in a specific manner (for example, route to server). Conditional steps have dependencies.

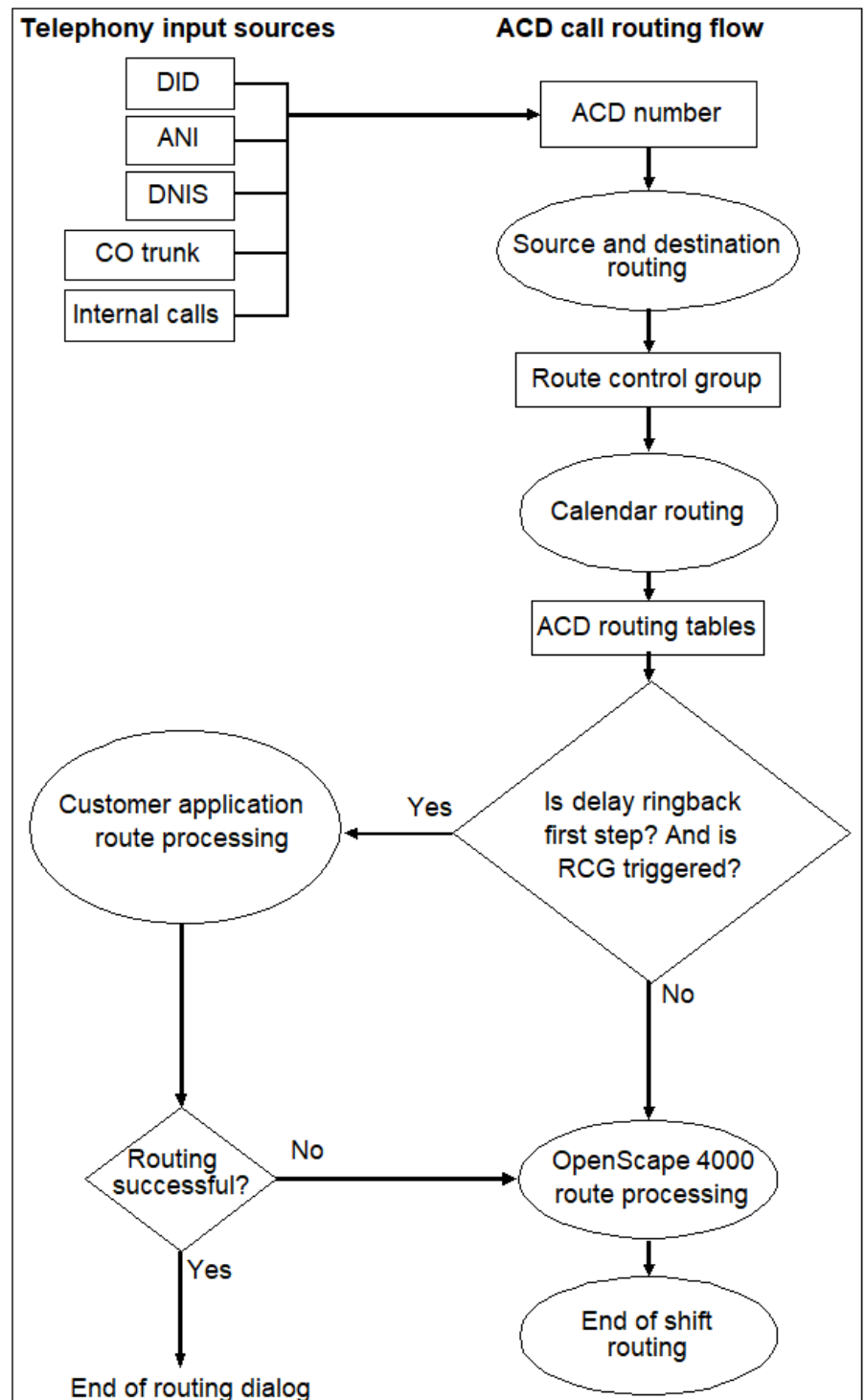
Each shift designates an ACD routing table to be used during that shift. As soon as the system assigns a call to an ART, it becomes an ACD call. A call

made directly to the extension number of a logged on agent is not an ACD call.

The computer application can activate or deactivate special handling of ACD calls using routing services. When a call comes into the OpenScape 4000, the computer application is notified, if special handling was set up. The computer application may route the call to a different destination based upon several factors: ANI number supplied, translated DNIS number; or the combination of ANI and DNIS numbers, if both are supplied. In these instances, ANI and DNIS get special routing instructions before normal processing.

- ACD Queuing

- To ensure that incoming calls are handled as efficiently as possible, a single call can queue for as many as 16 different ACD groups. CA 4000 can provide an application with events that monitor queuing activity.



ACD Routing Flow Diagram

### ACD Terminology

The following are some important ACD terms:

- ACD Calls
- An ACD call is an incoming call that reaches an ACD number. If an incoming call arrives on a trunk group dedicated to an ACD number, it immediately becomes an ACD call and begins to be routed to the system. However, if an incoming call arrives on a trunk that is not dedicated to an ACD number, the system does not route the call until it does reach an ACD number—for example, if the call is internally transferred.
- ACD Groups
- An ACD group is a group of agent extensions that receives calls.

When a group member (agent) is busy on a call, the system routes the calls to an available agent within the group. If all agents are busy, the system can also transfer incoming calls from one ACD group to another.

An ACD group can be a single extension, such as customer service. Larger departments can also be divided into many smaller ACD groups.

- ACD Number
- A diallable number that initiates processing of the call as an ACD call. The ACD number maps the call to a Route Control Group (RCG) depending on the source of the call (ANI), the destination of the call (DNIS), the day of the week, and the time of day.
- ACD RCG (Route Control Group)
- An Automatic Call Distribution (ACD) Route Control Group (RCG), configured in the OpenScape 4000 software, is the entry point used by ACD software for routing calls.

An ACD number directs a call to an RCG. Each RCG is identified by a unique, non-diallable number that is defined by configuration. If the specified routing is destination, call processing uses the specified RCG for the destination ACD number, if the routing option is set to source, the system uses the source ACD number and its associated RCG for routing the call.

The same RCG can be assigned to multiple ACD numbers.

- ACD Routing Table (ART)
- ARTs are tables that permit the configuration of the call routing. An ACD routing table is a set of instructions that an ACD call follows until an agent is available to answer the call.

For example, the caller may first hear a recorded message stating that all agents are still busy, followed by music for a certain number of seconds. If an agent is still not available, the call may be routed to another group of agents, or eventually to an off-site number.

Many routing tables can be configured for each ACD group. This permits customized routing to meet each group's requirements.

### 5.17.1.2 External ACD Call completed to agent

The external caller D1 (NID = N1) calls the ACD-pilot-number ( R2 intDnis ). No agent is available, the call is queued. As soon as an agent becomes available, the call is routed to ACD-group G. The call is routed to an agent.

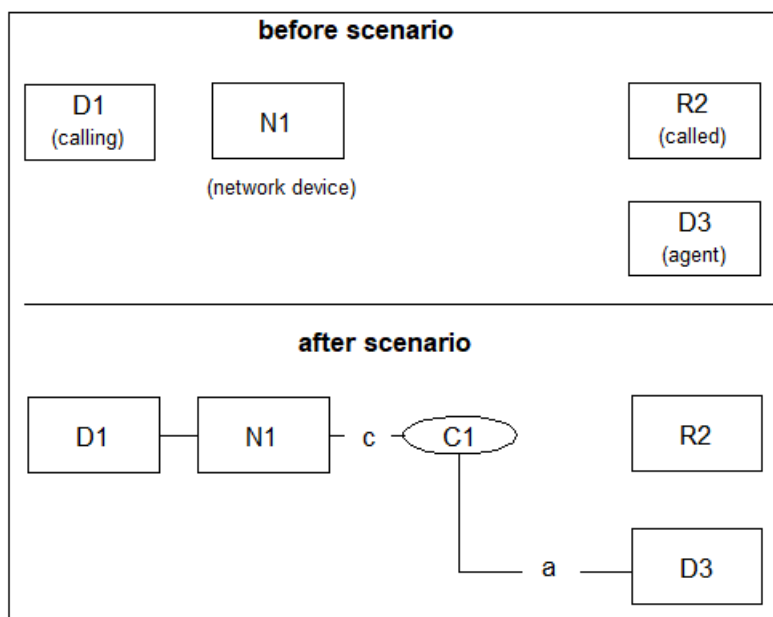


Figure 67: External ACD call completed to agent

Table 286: External Call Incoming to ACD Agent - Call Completed to Agent

Activity	Monitored Device N1 (Trunk)		Monitored Device R2 (RCG)		Monitored Device D3 (agent)		Comments
1) An incoming trunk is seized.	Service Initiated				None		
	• initiatedConnection	N1C1					
	• initiatingDevice	N1					
	• localConnection-Info	initiated					
	• cause	normal					
	• services-Permitted	ClearConn					
1) The call is routed to an ACD routing table.	Originated						(1) AssCledDev:
	• originatedConnection	N1C1					It is only provided for external calls to an RCG where the trunk is not monitored. It then contains the ACD-DNIS (e.g. R2 intDnis). In this scenario it is not provided because the trunk is monitored.
	• callingDevice	D1					
	• calledDevice	R2 intDnis					
	• NWCallingDevice	D1					Please note: this remark applies to all subsequent events for the RCG and the agent
	• AssCallingDevice	N1					
	• lastRedirection-Dev	NS					
	• localConnection-Info	connected					
	• cause	normal					
	• services-Permitted	ClearConn					

## Call Scenarios

Activity	Monitored Device N1 (Trunk)		Monitored Device R2 (RCG)		Monitored Device D3 (agent)		Comments
	Delivered		Delivered				
	• connection	R2C1	• connection	R2C1			
	• alertingDevice	R2	• alertingDevice	R2			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	R2 intDnis	• calledDevice	R2 intDnis			
	• OrigNIDConn	N1C1	• OrigNIDConn	N1C1			
	• NWCcallingDevice	D1	• NWCcallingDevice	D1			
	• AssCallingDevice	N1	• AssCallingDevice	N1			
			• AssCalledDevice	NP (1)			
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS			
	• localConnection-Info	connected	• localConnection-Info	alerting			
	• cause	enterDist	• cause	enterDist			
	• services-Permitted	ClearConn SendUI	• services-Permitted	ClearConn Deflect SendUI			
1) The call is routed to ACD G group.	None		None		None		
1) The call is queued.	Queued		Queued		None		G is the ACD-group.
	• queuedConnection	R2C1	• queuedConnection	R2C1			
	• queue	G	• queue	G			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	R2 intDnis	• calledDevice	R2 intDnis			
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS			
	• AssCallingDevice	N1	• AssCallingDevice	N1			
	• NWCcallingDevice	D1	• NWCcallingDevice	D1			
	• localConnection-Info	connected	• localConnection-Info	queued			
	• cause	NoAgents	• cause	NoAgents			

Activity	Monitored Device N1 (Trunk)		Monitored Device R2 (RCG)		Monitored Device D3 (agent)		Comments
	• services-Permitted	ClearConn SendUI	• services-Permitted	ClearConn Deflect SendUI			
1) An agent becomes free; the call is diverted from the RCG.			Diverted				
			• connection	R2C1			
			• divertingDevice	R2			
			• newDestination	D3			
			• callingDevice	D1			
			• calledDevice	R2 intDnis			
			• lastRedirection-Dev	NS			
			• AssCallingDevice	N1			
			• NWCallingDevice	D1			
			• localConnection-Info	null			
			• cause	Distributed			
			• services-Permitted	none			
			• services-Permitted	none			
1) The call is delivered to the agent.	Delivered				Delivered		Please note:  Even though the call was diverted from the RCG, the LastRedirectionDevice is not reported in the Delivered-Events.
	• connection	D3C1			• connection	D3C1	
	• alertingDevice	D3			• alertingDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	R2 intDnis			• calledDevice	R2 intDnis	
	• lastRedirection-Dev	NS			• lastRedirection-Dev	NS	
	• OrigNIDConn	N1C1			• OrigNIDConn	N1C1	
	• AssCallingDevice	N1			• AssCallingDevice	N1	
	• NWCallingDevice	D1			• NWCallingDevice	D1	
	• localConnection-Info	connected			• localConnection-Info	alerting	
	• cause	Distributed			• cause	Distributed	

## Call Scenarios

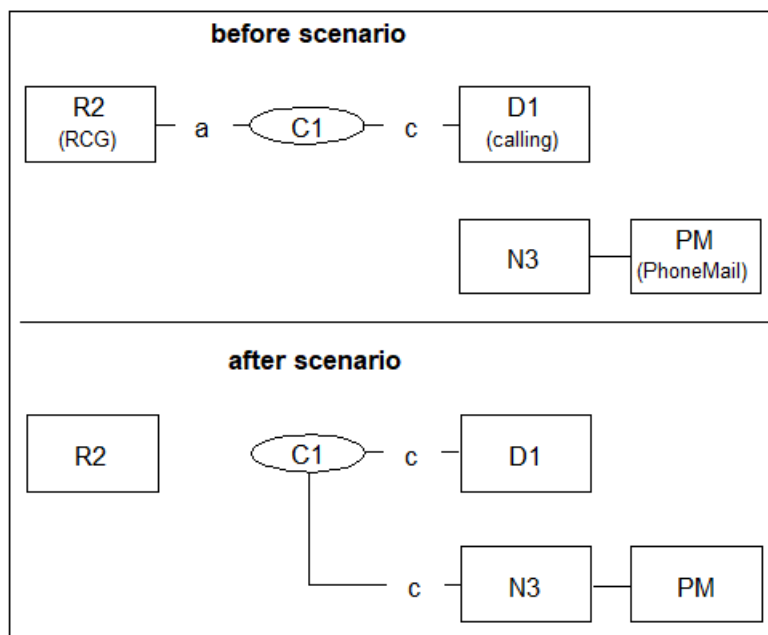
Activity	Monitored Device N1 (Trunk)		Monitored Device R2 (RCG)		Monitored Device D3 (agent)		Comments
	• services-Permitted	ClearConn SendUI			• services-Permitted	Answer ClearConn Deflect SendUI	

**Remark:**

None

### 5.17.1.3 Internal ACD call completed to Phone Mail agent=

This scenario describes a call flow when an RCG routes a call to a free Phonemail Agent.



**Figure 68: Internal ACD call completed to Phone Mail agent**

**Table 287: Route to free Phone Mail Agent**

Activity	Monitored Device D1		Monitored Device R2 (RCG)		Monitored Device N3		Comments
1) RCG R2 routes the call to the PM agent.			Diverted				
			• connection	R2C1			
			• divertingDevice	R2			
			• newDestination	PM			

Activity	Monitored Device D1		Monitored Device R2 (RCG)		Monitored Device N3		Comments
			• callingDevice	D1			
			• calledDevice	R2 pilot number			
			• lastRedirectionDevice	NS			
			• localConnectionInfo	null			
			• cause	distributed			
			• servicesPermitted	none			
1) The call leaves the CSTA subdomain.	Network Reached				Network Reached		
	• outboundConnection	N3C1			• outboundConnection	N3C1	
	• networkInterfaceUsed	N3			• networkInterfaceUsed	N3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	R2 pilot - number			• calledDevice	R2 pilot number	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	
	• localConnectionInfo	connected			• localConnectionInfo	connected	
	• cause	normal			• cause	normal	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTone, SendUserInfo			• servicesPermitted	ClearConn, SendUserInfo	
1) The Phone Mail answers the call immediately.	Established				Established		
	• establishedConnection	N3C1			• establishedConnection	N3C1	
	• answeringDevice	PM			• answeringDevice	PM	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	R2 pilot - number			• calledDevice	R2 pilot number	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device R2 (RCG)		Monitored Device N3		Comments
	• lastRedirection-Device	NS			• lastRedirection-Device	NS	
	• localConnection-Info	connected			• localConnection-Info	connected	
	• cause	distributed			• cause	distributed	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTone, SendUserInfo			• servicesPermitted	ClearConn, SendUserInfo	
	• assocCalledDevice	N3			• assocCalledDevice	N3	

**Remark:**

None

### 5.17.1.4 External call overflow to another RCG

The external caller D1 (NID = N1) calls the ACD-pilot-number ( R2 intDnis ). No agent is available, the call is queued. After a time, the call is routed to another RCG (R3). An agent is available, the call is routed to the agent.

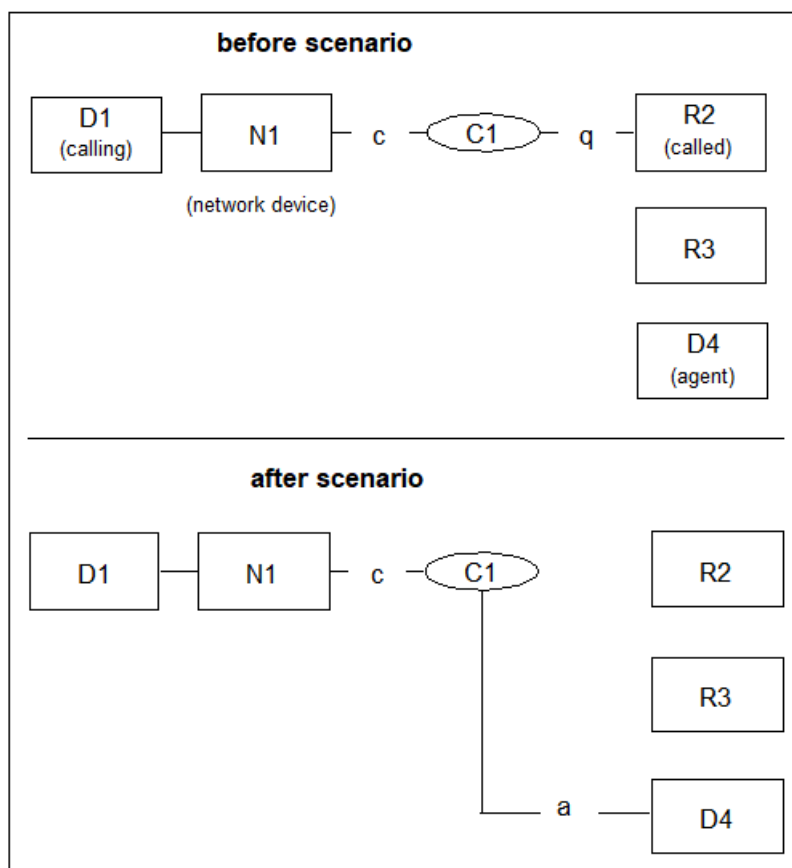


Figure 69: External call overflow to another RCG

Table 288: External overflow to another RCG

Activity	Monitored Device N1		Monitored Device R2		Monitored Device R3		Monitored Device D4 (agent)	
1) The call was previously queued at RCG2. After a timer elapses, RCG2 diverts the call to RCG3	None		Diverted					
			• connection	R2C1				
			• divertingDevice	R2				
			• newDestination	R3				
			• callingDevice	D1				
			• calledDevice	R2 intDnis				
			• lastRedirection-Dev	NS				
			• AssCallingDevice	N1				
			• NWCcallingDevice	D1				
			• localConnection-Info	null				
			• cause	normal				
			• services-Permitted	none				

## Call Scenarios

Activity	Monitored Device N1		Monitored Device R2		Monitored Device R3		Monitored Device D4 (agent)		Comments
1) The call rings at RCG3	Delivered				Delivered				
	• connection	R3C1			• connection	R3C1			
	• alertingDevice	R3			• alertingDevice	R3			
	• callingDevice	D1			• callingDevice	D1			
	• calledDevice	R2 intDnis			• calledDevice	R2 intDnis			
	• lastRedirection-Dev	NS			• lastRedirection-Dev	NS			
	• OrigNIDConn	N1C1			• OrigNIDConn	N1C1			
	• AssCallingDevice	N1			• AssCallingDevice	N1			
	• NWCcallingDevice	D1			• NWCcallingDevice	D1			
	• localConnection-Info	connected			• localConnection-Info	alerting			
	• cause	enterDist			• cause	enterDist			
	• services-Permitted	ClearConn SendUI			• services-Permitted	ClearConn Deflect SendUI			
1) The call is routed to an ACD Group	None		None		None		None		
1) An agent is available - the call is diverted from the RCG to the agent's phone					Diverted				
					• connection	R3C1			
					• divertingDevice	R3			
					• newDestination	D4			
					• callingDevice	D1			
					• calledDevice	R2 intDnis			
					• lastRedirection-Dev	NS			
					• AssCallingDevice	N1			
					• NWCcallingDevice	D1			
					• localConnection-Info	null			
					• cause	Distributed			
					• services-Permitted	none			

Activity	Monitored Device N1		Monitored Device R2		Monitored Device R3		Monitored Device D4 (agent)	
1) The call starts ringing at the agent's phone	Delivered						Delivered	
	• connection	D4C1					• connection	D4C1
	• alertingDevice	D4					• alertingDevice	D4
	• callingDevice	D1					• callingDevice	D1
	• calledDevice	R2 intDnis					• calledDevice	R2 intDnis
	• lastRedirection-Dev	NS					• lastRedirection-Dev	NS
	• OrigNIDConn	N1C1					• OrigNIDConn	N1C1
	• assCallingDevice	N1					• assCallingDevice	N1
	• NWCcallingDevice	D1					• NWCcallingDevice	D1
	• localConnection-Info	connected					• localConnect-ion-Info	alerting
	• cause	Distributed					• cause	Distributed
	• services-Permitted	ClearConn SendUI					• services-Permitted	Answer ClearConn Deflect SendUI

**Remark:**

None

## 5.17.2 Make Predictive Call

The Make Predictive Call Service originates a call between two devices by first creating a connection to the called device. The service returns a positive acknowledgment that provides the connection at the called device.

On OpenScape 4000, the calling device is always an RCG. The called device may be any station or trunk. After the called device has answered the call, the RCG proceeds with its ART-Table. Usually, the call is either distributed to an agent or diverted to a station.

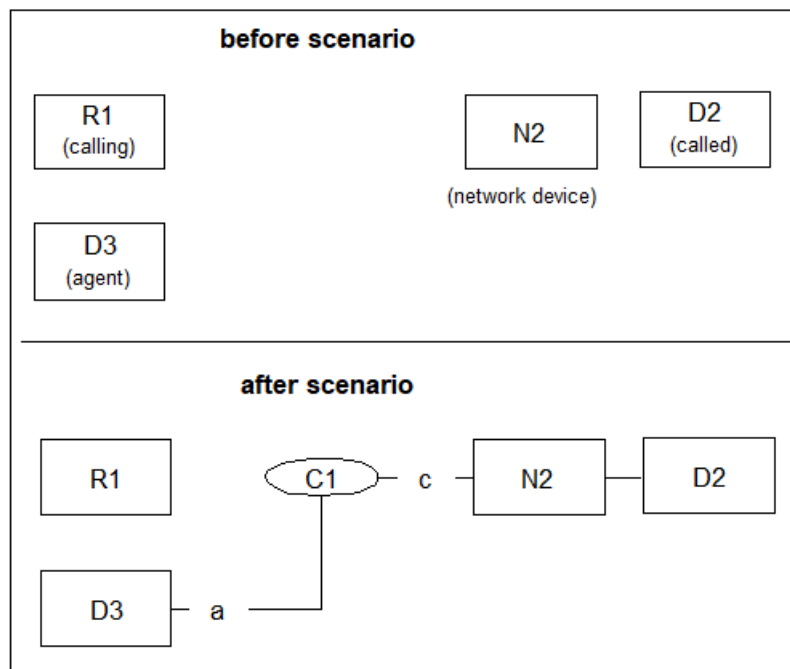
Connections created by Make Predictive Call are cleared after:

- the called party fails to answer within a certain amount of time
- the switch has detected that the called device is unable to answer (is busy, for example)

### 5.17.2.1 Make Predictive Call - to external free device

Make Predictive Call from RCG to external party outside the CSTA subdomain. The external party answers the call, the RCG routes the call to an agent.

## Call Scenarios



**Figure 70: Predictive call to external free device**

**Table 289: Make Predictive Call - to external free device**

Activity	Monitored Device R1 (RCG)		Monitored Device N2 (trunk)		Monitored Device D3 (agent)		Comments
1) A Make Predictive Call to a valid device is invoked on behalf of a RCG	Make Predictive Call - Service Request						
	• callingDevice	R1					
	• calledDirectoryNumber	D2					
	Make Predictive Call - Positive Response.						
	• initiatedCall	N2C1					
1) RCG device is initiated	Service Initiated						
	• initiatedConnection	R1C1					
	• initiatingDevice	R1					
	• localConnect-ionInfo	initiated					
	• cause	makePredCall					
	• servicesPermit- ted	ClearConn					
1) The call leaves the CSTA subdo- main	Network Reached		Network Reached				
	• outboundConn	N2C1	• outboundConn	N2C1			
	• NWInterfaceUsed	N2	• NWInterfaceUsed	N2			
	• callingDevice	R1	• callingDevice	R1			
	• calledDevice	D2	• calledDevice	D2			

Activity	Monitored Device R1 (RCG)		Monitored Device N2 (trunk)		Monitored Device D3 (agent)		Comments
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS			
	• NW-Capability	ISDN Public	• NW-Capability	ISDN Public			
	• localConnect-ionInfo	initiated	• localConnect-ionInfo	connected			
	• cause	normal	• cause	normal			
	• servicesPermit-ted	ClearConn	• servicesPermit-ted	ClearConn, Deflect SendUI			
1) D2 is alerted	Delivered		Delivered				
	• connection	N2C1	• connection	N2C1			
	• alertingDevice	D2	• alertingDevice	D2			
	• callingDevice	R1	• callingDevice	R1			
	• calledDevice	D2	• calledDevice	D2			
	• AssCalledDevice	N2	• AssCalledDevice	N2			
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS			
	• localConnect-ionInfo	initiated	• localConnect-ionInfo	connected			
	• cause	networkSignal	• cause	networkSignal			
	• servicesPermit-ted	ClearConn	• servicesPermit-ted	ClearConn, Deflect SendUI			
1) D2 answers the call	Established		Established				
	• establishedConn	N2C1	• establishedConn	N2C1			
	• answeringDevice	D2	• answeringDevice	D2			
	• callingDevice	R1	• callingDevice	R1			
	• calledDevice	D2	• calledDevice	D2			
	• AssCalledDevice	N2	• AssCalledDevice	N2			
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS			
	• localConnect-ionInfo	initiated	• localConnect-ionInfo	connected			
	• cause	networkSignal	• cause	networkSignal			
	• servicesPermit-ted	ClearConn, SendUI	• servicesPermit-ted	ClearConn, SendUI			

## Call Scenarios

Activity	Monitored Device R1 (RCG)		Monitored Device N2 (trunk)		Monitored Device D3 (agent)		Comments
1) Call comes back into the switching domain and is delivered to the RCG	Delivered		Delivered				
	• connection	R1C1	• connection	R1C1			
	• alertingDevice	R1	• alertingDevice	R1			
	• callingDevice	R1	• callingDevice	R1			
	• calledDevice	D2	• calledDevice	D2			
	• AssCalledDevice	N2	• AssCalledDevice	N2			
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS			
	• localConnect-ionInfo	alerting	• localConnect-ionInfo	connected			
	• cause	enteringDist	• cause	enteringDist			
	• servicesPermit-ted	ClearConn, SendUI	• servicesPermit-ted	ClearConn SendUI			
1) An Available agent at Device D3 is chosen and the call is diverted to that device.	Diverted						Please note: the calledDevice is optional and not provided. However, the AssCalled-Device is mandatory for outgoing external calls and therefore provided.
	• connection	R1C1					
	• divertingDevice	R1					
	• newDestination	D3					
	• AssCalledDevice	N2					
	• lastRedirection-Dev	NS					
	• localConnect-ionInfo	null					
	• cause	distributed					
	• servicesPermit-ted	none					
1) D3 is alerted			Delivered		Delivered		
			• connection	D3C1	• connection	D3C1	
			• alertingDevice	D3	• alertingDevice	D3	
			• callingDevice	D3	• callingDevice	D3	
			• calledDevice	D2	• calledDevice	D2	
			• AssCalledDevice	N2	• AssCalledDevice	N2	
			• lastRedirection-Dev	NS	• lastRedirection-Dev	NS	
			• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting	
			• cause	distributed	• cause	distributed	

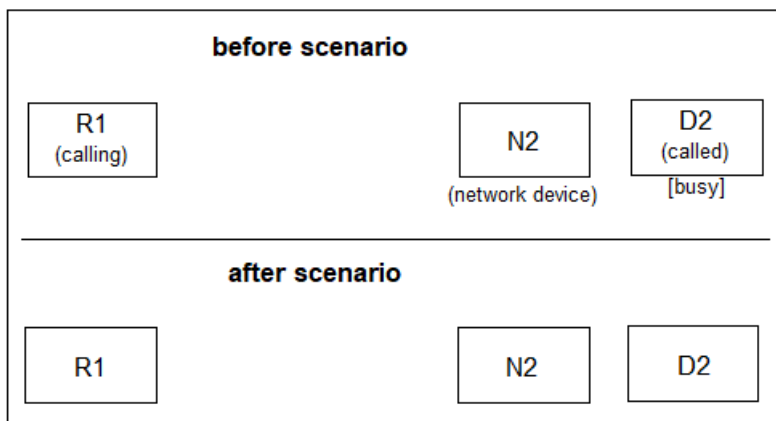
Activity	Monitored Device R1 (RCG)		Monitored Device N2 (trunk)		Monitored Device D3 (agent)		Comments
			• servicesPermit-ted	ClearConn SendUI	• servicesPermit-ted	Answer ClearConn Deflect SendUI	
1) D3 answers the call			Established		Established		
			• establishedConn	D3C1	• establishedConn	D3C1	
			• answeringDevice	D3	• answeringDevice	D3	
			• callingDevice	D3	• callingDevice	D3	
			• calledDevice	D2	• calledDevice	D2	
			• AssCalledDevice	N2	• AssCalledDevice	N2	
			• lastRedirection-Dev	NS	• lastRedirection-Dev	NS	
			• localConnection-Info	connected	• localConnection-Info	connected	
			• cause	NWSignal	• cause	NWSignal	
			• servicesPermit-ted	ClearConn, SendUI	• servicesPermit-ted	ClearConn, Consultation, Hold, SST, GenDG, GenTelTone,SendUI	

**Remark:**

None

### 5.17.2.2 Make Predictive Call - to external busy device

Make Predictive Call to a busy party outside the CSTA subdomain. The call cannot be completed.



**Figure 71: Predictive call to external busy device**

## Call Scenarios

**Table 290: Make Predictive Call - to external busy device**

Activity	Monitored Device R1		Monitored Device N2 (trunk)		Comments
1) A Make Predictive Call to a valid device is invoked on behalf of a RCG	Make Predictive Call - Service Request				
	• callingDevice	R1			
	• calledDirectoryNumber	D2			
	Make Predictive Call - Positive Response				
	• initiatedCall	N2C1			
1) RCG device is initiated	Service Initiated				
	• initiatedConnection	R1C1			
	• initiatingDevice	R1			
	• localConnection-Info	initiated			
	• cause	makePredCall			
	• services-Permitted	none			
1) The call leaves the CSTA subdomain	Network Reached		Network Reached		
	• outboundConn	N2C1	• outboundConn	N2C1	
	• NWInterface-Used	N2	• NWInterface-Used	N2	
	• callingDevice	R1	• callingDevice	R1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS	
	• NW-Capability	ISDN Private	• NW-Capability	ISDN Private	
	• localConnection-Info	initiated	• localConnection-Info	connected	
	• cause	normal	• cause	normal	
	• services-Permitted	none	• services-Permitted	none	
1) Device D2 is busy the call cannot be completed.	Failed		Failed		
	• failedConnection	N2C1	• failedConnection	N2C1	
	• failingDevice	D2	• failingDevice	D2	
	• callingDevice	R1	• callingDevice	R1	
	• calledDevice	D2	• calledDevice	D2	
	• AssCalledDevice	N2	• AssCalledDevice	N2	
	• lastRedirection-Dev	NS	• lastRedirection-Dev	NS	
	• localConnection-Info	initiated	• localConnection-Info	fail	

Activity	Monitored Device R1		Monitored Device N2 (trunk)		Comments
	• cause	busy	• cause	busy	
	• services-Permitted	none	• services-Permitted	none	
1) Connection is cleared for device D2	Connection Cleared		Connection Cleared		
	• droppedConnection	N2C1	• droppedConnection	N2C1	
	• releasingDevice	N2	• releasingDevice	N2	
	• localConnection-Info	initiated	• localConnection-Info	null	
	• cause	normalClr	• cause	normalClr	
	• services-Permitted	none	• services-Permitted	none	
1) Connection is cleared for device D1 (RCG)	Connection Cleared				Connection clears as a result of Make Predictive Call condition
	• droppedConnection	R1C1			
	• releasingDevice	R1			
	• localConnection-Info	null			
	• cause	normalClr			
	• services-Permitted	none			

**Remark:**

None

## 5.17.3 Route Services

The OpenScape 4000 is capable of allowing an external computer application to influence incoming calls. The application may divert incoming calls to a different agent, ACD group, or to another point within the telephone network. If the diverted destination is busy, the OpenScape 4000 allows the computer application further opportunities to reroute the call.

For an application to influence a call, two requirements must be met. First, the RCG to which the call was originally destined must be registered by the application or the gateway. The second requirement is that the ACD routing table (ART) within the RCG must have a delay ringback as its first programmed sequence. The delay ringback step has a timer associated with it that determines the length of time that the application may influence the call. If these two requirements are met, the application can influence the call.

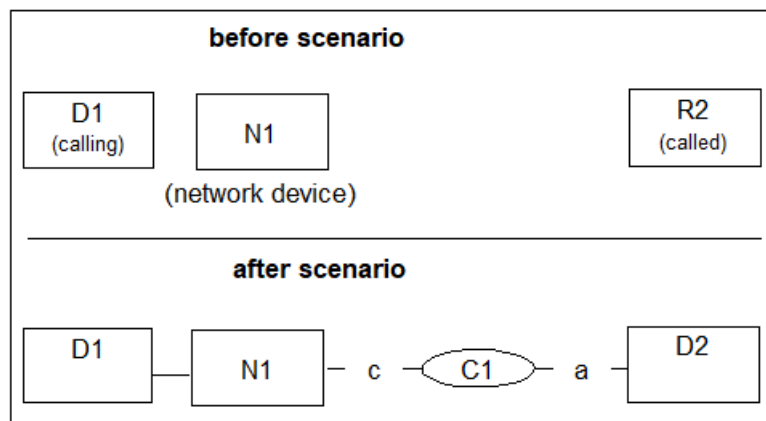
The delay ringback timer allows the computer application time to influence the routing of the incoming call before the OpenScape 4000 sends the caller ringback tone. Each time a call comes into a registered RCG, the OpenScape 4000 sends a Route Request to the gateway, thereby giving the computer application an opportunity to influence the call. The Route Request initiates a routing dialog with the application. The gateway has 5 seconds to respond. The response time is measured by a routing timer within the OpenScape 4000. If the routing timer expires, the OpenScape 4000 sends the gateway a

Route End signal to terminate the routing dialog. If the application responds to the OpenScape 4000 Route Request with a Route Select request within the required time, the OpenScape 4000 attempts to divert the call to the new destination specified by the computer application. If the OpenScape 4000 is successful in diverting the call, it sends a Route End signal to the application, terminating the routing dialog. If the OpenScape 4000 is not successful in diverting the call to the destination specified by the application, such as when the specified destination is busy, the OpenScape 4000 sends a Re-Route Request to the gateway. The route timer is reset to 5 seconds, and the application can initiate a new Route Select with a new destination. Sometimes, the application may not influence the call when it receives the Route Request from the OpenScape 4000. Instead of responding with a Route Select signal to the OpenScape 4000, it sends a Route End signal terminating the routing dialog. In this instance, the OpenScape 4000 continues processing the incoming call by using the next step on the RCG's ART table. In some cases, the application may reject the incoming call. If the ADR service is purchased and programmed, the network will redirect the call to another destination at some other location on the telephone network.

The computer application may route the incoming call to another RCG that is different from the RCG that the call was originally destined for. If the second RCG is registered and has delay ringback set in its ART table, call processing will bypass the delay ringback step and proceed to the next step in the ART table.

### 5.17.3.1 Route Request Scenario

This scenario describes an event flow of an external incoming call to an RCG that is redirected by the computer application to a different destination.



**Figure 72: Route request**

**Table 291: Route Request scenario**

Activity	Monitored Device N1	Monitored Device R2 (RCG)	Monitored Device D2	Comments
1) N1 seized.	Service Initiated			
	• initiatedConnection N1C1			

Activity	Monitored Device N1		Monitored Device R2 (RCG)		Monitored Device D2		Comments
	• initiatingDevice	N1					
	• localConnectionInfo	initiated					
	• cause	normal					
	• servicesPermitted	ClearConn					
1) N1 completes dialling the R2 RCG.	Originated						
	• originatedConnection	N1C1					
	• callingDevice	D1					
	• calledDevice	R2 intDNIS					
	• localConnectionInfo	connected					
	• cause	normal					
	• servicesPermitted	ClearConn					
	• networkCallingDevice	D1					
	• assocCallingDevice	N1					
	• assocCalledDevice	R2 intDNIS					
1) The call arrives at the RCG.	Delivered		Delivered				
	• deliveredConnection	R2C1	• deliveredConnection	R2C1			
	• alertingDevice	R2	• alertingDevice	R2			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	R2 intDNIS	• calledDevice	R2 intDNIS			
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS			
	• originalNID	N1C1	• originalNID	N1C1			
	• localConnectionInfo	connected	• localConnectionInfo	alerting			
	• cause	enterDistribution	• cause	enterDistribution			
	• servicesPermitted	ClearConn, SendUserInfo	• servicesPermitted	ClearConn, Deflect			
	• networkCallingDevice	D1	• networkCallingDevice	D1			

## Call Scenarios

Activity	Monitored Device N1		Monitored Device R2 (RCG)		Monitored Device D2		Comments
	• assocCallingDevice	N1	• assocCalling-Device	N1			
	• assocCalledDevice	R2 intDNIS	• assocCalledDe-vice	R2 intDNIS			
1) Route Request is sent to the application to let the computing function route the call.	Route Request						
	• crossRefID	1					
	• referenceID	1					
	• calledDevice	R2 intDNIS					
	• callingDevice	D1					
	• routingDevice	R2					
	• routedCall	R2C1					
	• assocCallingDevice	N1					
	• assocCalledDevice	R2 intDNIS					
1) The application sends the route destination.	Route Select Request						
	• crossRefID	1					
	• routeRegisterRequestID	1					
	• routeSelected	D2					
1) The routing is successful.			Diverted				The switching function sends the Diverted event only to the diverting-Device.
			• divertedConnection	R2C1			
			• divertingDevice	R2			
			• newDestinationDevice	D2			
			• callingDevice	D1			
			• calledDevice	R2 intDNIS			
			• lastRedirectionDevice	NS			
			• localConnectionInfo	null			
			• cause	normal			
			• servicesPermitted	SendUserInfo			

Activity	Monitored Device N1		Monitored Device R2 (RCG)		Monitored Device D2		Comments
			• networkCalling-Device	D1			
			• assocCalling-Device	N1			
			• assocCalledDevice	R2 intDNIS			
1) D2 alerts.	Delivered				Delivered		The switching function provides lastRedirection-Device NS instead of the proper value.
	• deliveredConnection	D2C1			• deliveredConnection	D2C1	
	• alertingDevice	D2			• alertingDevice	D2	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	R2 intDNIS			• calledDevice	R2 intDNIS	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	
	• originalNID	N1C1			• originalNID	N1C1	
	• localConnectionInfo	connected			• localConnectionInfo	alerting	
	• cause	distributed			• cause	distributed	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo			• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	
	• networkCallingDevice	D1			• networkCallingDevice	D1	
	• assocCallingDevice	N1			• assocCallingDevice	N1	
	• assocCalledDevice	R2 intDNIS			• assocCalledDevice	R2 intDNIS	
1) Route End Request is sent to the application to close the dialog.	Route End Request						
	• crossRefID	1					
	• routeRegisterRequestID	1					

**Remark:**

None

5.17.3.2 Re-Route Request Scenario

This scenario describes an event flow of an external incoming call to an RCG that is redirected by the computer application to a busy destination and after that, it is rerouted to another destination. No events will be reported for the busy destination.

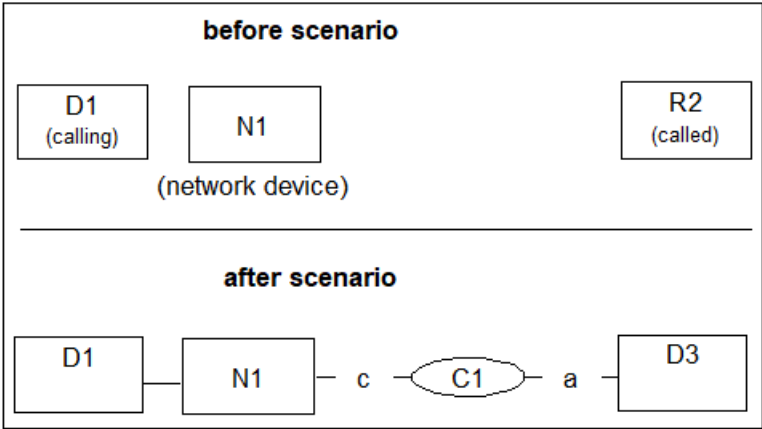


Figure 73: Re-route request

Table 292: Re-Route Request scenario

Activity	Monitored Device N1	Monitored Device R2 (RCG)	Monitored Device D3	Comments
Steps 1-3 are shown in "Route Request Scenario".				
1) Route Request is sent to the application to let the computing function route the call.	Route Request			
	• crossRefID	1		
	• referenceID	1		
	• calledDevice	R2 intDNIS		
	• callingDevice	D1		
	• routingDevice	R2		
	• routedCall	R2C1		
	• assocCallingDevice	N1		
1) The application sends the route destination.	• assocCalledDevice	R2 intDNIS		
	Route Select Request			
	• crossRefID	1		
	• routeRegisterRequestID	1		
	• routeSelected	D2		

Activity	Monitored Device N1		Monitored Device R2 (RCG)		Monitored Device D3		Comments
1) ReRoute Request is sent to the application to inform it that the routing destination was busy and to let the computing function choose a new destination.	ReRoute Request						
	• crossRefID	1					
	• routeRegisterRequestID	1					
1) The application sends its other destination.	Route Select Request						The new destination becomes D3.
	• crossRefID	1					
	• routeRegisterRequestID	1					
1) The routing is successful.	• routeSelected	D3					
			Diverted				The switching function sends the Diverted event only to the diverting-Device.
			• divertedConnection	R2C1			
			• divertingDevice	R2			
			• newDestinationDevice	D3			
			• callingDevice	D1			
			• calledDevice	R2 intDNIS			
			• lastRedirectionDevice	NS			
			• localConnectionInfo	null			
			• cause	normal			
			• servicesPermitted	SendUserInfo			
			• networkCallingDevice	D1			
			• assocCallingDevice	N1			

## Call Scenarios

Activity	Monitored Device N1		Monitored Device R2 (RCG)		Monitored Device D3		Comments
			• assocCalledDe- vice	R2 intDNIS			
1) D2 alerts.	Delivered				Delivered		The switching function provides lastRedirection-Device NS instead of the proper value.
	• deliveredConnec- tion	D3C1			• deliveredConnec- tion	D3C1	
	• alertingDevice	D3			• alertingDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	R2 intDNIS			• calledDevice	R2 intDNIS	
	• lastRedirection- Device	NS			• lastRedirection- Device	NS	
	• originalNID	N1C1			• originalNID	N1C1	
	• localConnection- Info	connected			• localConnec- tionInfo	alerting	
	• cause	distributed			• cause	distributed	
	• servicesPermit- ted	CallBack, ClearConn, SendUserInfo			• servicesPermit- ted	Answer, ClearConn, Deflect, SendUser- Info	
	• networkCalling- Device	D1			• networkCalling- Device	D1	
	• assocCallingDe- vice	N1			• assocCalling- Device	N1	
	• assocCalledDe- vice	R2 intDNIS			• assocCalledDe- vice	R2 intDNIS	
1) Route End Request is sent to the application to close the dialog.	Route End Request						
	• crossRefID	1					
	• routeRegisterRe- questID	1					

**Remark:**

None

### 5.17.3.3 Route End Request Scenario

This scenario describes an event flow of an external incoming call to an RCG for which the computer application is allowed to influence the destination

routing, but the computer application declines to use its rerouting. The next step in the RCG's ACD routing table will be selected.

**Table 293: Route End Request scenario**

Activity	Monitored Device N1	Monitored Device N2	Comments
Steps 1-3 are shown in <a href="#">Section 5.17.3.1, "Route Request Scenario"</a> .			
1) Route Request is sent to the application to let the computing function route the call.	Route Request		
	• crossRefID	1	
	• referenceID	1	
	• calledDevice	R2 intDNIS	
	• callingDevice	D1	
	• routingDevice	R2	
	• routedCall	R2C1	
	• assocCallingDevice	N1	
	• assocCalledDevice	R2 intDNIS	
1) The application sends Route End Request.	Route End Request		
	• crossRefID	1	
	• routeRegisterRequestID	1	

**Remark:**

None

### 5.17.3.4 Reject Call Scenario

The Reject Call request is sent by the computer application to the switching function during a routing dialog to indicate that the OpenScape 4000 should return busy to the network. If the customer has purchased the Alternate Destination Redirection (ADR) service from the network, the ADR service causes the incoming call to be routed to another (preconfigured) destination in the network.

**Table 294: Reject Call scenario**

Activity	Monitored Device N1	Monitored Device R2 (RCG)	Comments
Steps 1-3 are shown in <a href="#">Section 5.17.3.1, "Route Request Scenario"</a> .			
1) Route Request is sent to the application to let the computing function route the call.	Route Request		
	• crossRefID	1	
	• referenceID	1	

## Call Scenarios

Activity	Monitored Device N1		Monitored Device R2 (RCG)		Comments
	• calledDevice	R2 intDNIS			
	• callingDevice	D1			
	• routingDevice	R2			
	• routedCall	R2C1			
	• assocCallingDevice	N1			
	• assocCalledDevice	R2 intDNIS			
1) The application rejects the call.	Reject Request				
	• crossRefID	1			
	• routeRegisterRequestID	1			
1) Route End Request will be sent to the application to close the dialog.	Route End Request				
	• crossRefID	1			
	• routeRegisterRequestID	1			
1) RCG clears the far end will get busy tone.	Connection Cleared		Connection Cleared		
	• droppedConnection	R2C1	• droppedConnection	R2C1	
	• releasingDevice	R2	• releasingDevice	R2	
	• localConnectionInfo	connected	• localConnectionInfo	null	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	ClearConn	• servicesPermitted	none	
1) D1 hungs up.	Connection Cleared				
	• droppedConnection	N1C1			
	• releasingDevice	N1			
	• localConnectionInfo	null			
	• cause	normalClr			
	• servicesPermitted	none			

**Remark:**

None

## 5.17.4 Hunting Groups (HG)

### 5.17.4.1 General description Hunt Group

#### Introduction

Monitoring of Hunting Groups was provided in a non-standard way in previous versions. Please note that the call-flow has changed considerably!

CSTA III (ECMA 269) and the ECMA Call-Scenarios do not provide much information about how devices like HG should be modeled. Therefore it is necessary to describe the model CA 4000 uses.

#### Characteristics of Hunt Group

A Hunting Group is represented by a logical device, it does not have a physical appearance. Each HG is assigned at least one diallable number and a (unique) device-number. To monitor the HG, the device-number must be used (similar to RCGs, the number issued by HiPath 4000 must be masked). The mask for HG is 0x05000000. It is not possible to monitor the diallable number of a HG.

A HG has members and - usually - a queue where incoming calls are queued when no member is available. A monitor on the HG reports events for the HG-Device only. If an application wants to receive events for the members as well, it is necessary that it monitors all members individually. This is called the "Group-Exclusive-Model".

Remark: an application may obtain information about the members of a HG by invoking the GetLogicalDeviceInformation-Request.

If only the member of a HG is monitored but not the HG itself, an application will be able to tell the difference between a HG-call and a direct call to the member by looking at the event-cause in the Delivered-Event: if the cause is "Multi Alert" or "RemainsInQueue", the call has been distributed by the HG.

If a party picks up a HG-call ringing at a HG-member, the application needs to interpret the CSTA-CalledDevice to be able to tell whether the call was originally for a HG. Please note: this piece of information is not reliable, because the called device could be a device forwarded to the HG.

The main task of a HG is the distribution of calls to its members. In most cases, the HG remains involved in the call until

- the member answers the call or
- the caller has hung up

This means that both the HG and its member are ringing simultaneously. For handling these situations, CSTA III has introduced the new event-cause "MultipleAlerting". Please refer to [5.17.4.1, "General Rules concerning Multi-Alert-Situations"](#) for more information.

#### Deflect and Hunt Group

The following has to be considered when using the Deflect-Call-Request for calls where a HG is involved:

- For HG, Deflect-Call is only allowed when the call is queued
- Deflect-Call is not allowed in a Multiple-Alert-Situation (this includes all devices involved in the call).

#### Special features of Hunt Group

- HG-member: usually a normal station; the HG distributes incoming calls to its members. The type of distribution (linear, cyclic) is configured in the

switch. When a HG-member is ringing with a HG-call, two devices are ringing simultaneously: the HG and the HG-member. This is reflected in the Delivered-Event by the event-cause Multi-Alert. This indicates that the HG is still in control of the call. Please note: a HG-member can be on another node. In that case the used trunk will be treated as HG-member.

- Control of the call: the HG remains in control of the call until the call has left the HG for one of the following reasons:
  - the HG-member has answered the call
  - another device has picked up a call ringing at a HG-member
  - the caller has hung up
  - the call is deflected to another destination (only permitted when the call is in the HG-queue)
- Hunt Advance: a HG member fails to answer the call in time; the HG releases the device not answering the call and distributes the call to its next member
- HG-Queue: if a HG has no member configured or if no member is available, the call is queued in the HG-Queue. This is the only situation when HiPath 4000 permits a Deflect from a HG.
- Overflow-Destination: if the HG-Queue is full, HiPath 4000 seizes the overflow-destination for the HG (if configured). In this special case, HG withdraws from the call as soon as the overflow-destination starts ringing.

### Known Restrictions for Hunt Group

- Event-cause RemainInQ vs. MultiAlert for Non-Group-device
- If a call is distributed to a HG or a hunt-advance is performed, the status of the calling device in the HG remains unchanged: when the caller was queued, the position in the queue is kept. When the calling device was alerting at the HG, it stays alerting.

For the Delivered-Event there are two different event-causes that indicate whether the caller is queued or alerting at the Group-Device:

- Remains in Queue: the caller was queued before and keeps the position in the queue
- Multi Alert: the caller was alerting and is still alerting at the HG

Event-cause "RemainInQ" can only be provided for the calling and called party (= group member) if the HG is monitored. Otherwise, event-cause will be Multi Alert in both cases. This means e.g. that the calling party will receive a Queued-Event when the call is queued at the group-device and afterwards a Delivered-Event with event-cause MultiAlert when the call is distributed to the first HG-member, but physically the call will remain in the queue (refer to section [5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#) and section [5.17.4.4, "Call is queued at Hunt Group"](#)).

- ACD routes MakePredictiveCall-call into a HG
- If a call generated by Make-Predictive-Call is routed to a HG by the RCG (instead of an agent), the CallingDevice cannot be provided any longer as soon as the call hits the Group-Device and is therefore reported as "NotKnown".
- Called-Device
- There are situations where CA 4000 reports "NotKnown" as Called-Device when a HG is involved. Some of these situations are:
  - SST into a HG
  - Deflect into a HG (when the HG is the first device to be monitored)

### General Rules concerning Multi-Alert-Situations

- What is a Multi-Alert-Situation?
- Multi-Alerting means that a call is ringing at more than one devices. For HG, this happens when the HG distributes a call to one of its members. Both the member and the Group-Device are ringing simultaneously (see section [5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#))
- Rules for Multi-Alert-Situations concerning HG
- Generally, CA 4000 models diversions by sending the Diverted-Event for the Diverting-Device only. In case of a successful diversion, the calling device and the new destination will receive a Delivered-Event with LastRedirectionDevice = Diverting Device.

Here are some rules CA 4000 uses for handling Diversions / Multi-Alert-situations. Please note: these rules are not described in the ECMA CSTA III standard:

If a call is ringing at a device B and CA 4000 sends another Delivered-Event with a new AlertingDevice C, an application must infer that a diversion from B to C has taken place - unless the event cause is MultiAlert. (see rule # 2)

Examples:

#### 1) Deflect from B to C

- CallForward-NoAnswer from B to C

If the second Delivered-Event is sent with event-cause "MultiAlert", this means that a new alerting device has been added to the call. In this case, no diversion has taken place. Now, more than one devices are ringing simultaneously with the same call.

Example:

- HG distributes the call to its member; both the HG and the member are ringing (see section [5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#))

If an alerting device leaves a Multi-Alert-call and reduces the Multiple-Alerting to a "normal" alerting, it depends upon the situation whether Diverted or Conn-Cleared is sent for the device leaving the call. Diverted is only sent if the call has moved to a new destination that was not involved with the call before.

Example:

- Another party picks up a HG-call ringing at a HG-member (see section [Section 5.17.4.7, "Pick from Hunt Group Member"](#)) In all other situations, a Connection-Cleared-Event must be sent.

Example:

- The HG leaves the call because the HG-member has answered the call; a Conn-Cleared has to be sent for the HG (see [Section 5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#)).

Event-cause "RemainsInQueue": in special situations this event-cause is reported instead of "Multi-Alert": whenever a call was placed in the queue of a HG, the call remains in the queue until the HG leaves the call. In these cases, the event-cause "Remains In Queue" will be reported instead of "Multiple Alerting" to show that the call keeps its place in the queue. (More information about this event cause, refer to [Event-cause RemainInQ vs. MultiAlert for Non-Group-device on page 521](#)).

### 5.17.4.2 Successful Group Call (Multiple Alerting with Parallel Ringing)

In this scenario device D1 calls a group of distribution mechanism (device D2) with members D3 and D4. There are devices available and the call is successfully distributed to devices D3 and D4 by the Group itself.

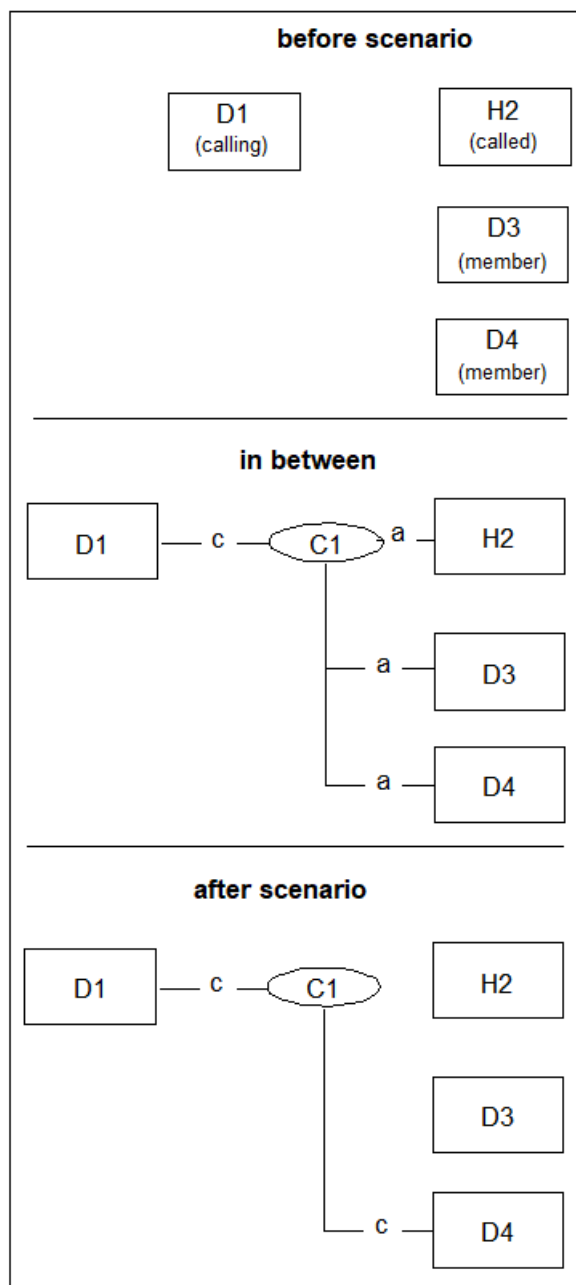


Figure 74: Successful group call (multiple alerting with parallel ringing)

**Table 295: Successful Group Call (Multiple Alerting with Parallel Ringing)**

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Com
1) D1 goes off-hook	Service Initiated								
	initiatedConnect-ion	D1C1							
	localConnect-ionInfo	initiated							
	cause	normal							
	services-Permitted								
1) D1 completes dialling HG-access-code (1234)	Digits Dialed								
	diallingConnection	D1C1							
	diallingDevice	D1							
	diallingSequence	"1234"							
	localConnectionInfo	initiated							
	cause	normal							
	Originated								
	originatedConnect-ion	D1C1							
	callingDevice	D1							
	calledDevice	H2							
	originatingDevice	D1							
	localConnect-ionInfo	connected							
	cause	normal							
	services-Permitted								
1) The call reaches th HG	Delivered		Delivered						
	Connect-ion	H2C1	Connect-ion	H2C1					
	alertingDevice	H2	alertingDevice	H2					
	callingDevice	D1	callingDevice	D1					
	calledDevice	H2	calledDevice	H2					
	lastRedirection-Dev	NS	lastRedirection-Dev	NS					
	localConnect-ionInfo	connected	localConnect-ionInfo	alerting					
	cause	enterDist	cause	enterDist					
	services-Permitted	ClearConn SendU	services-Permitted	SendUI					

## Call Scenarios

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Comments
1) Call reaches the HG queue	QueuedEvent		QueuedEvent						
	Connect-ion	H2C1	Connect-ion	H2C1					
	queuedDevice	H2	queuedDevice	H2					
	callingDevice	D1	callingDevice	D1					
	calledDevice	H2	calledDevice	H2					
	lastRedirection-Dev	NS	lastRedirection-Dev	NS					
	localConnect-ionInfo	connected	localConnect-ionInfo	queued					
	cause	NoAvail Agents	cause	NoAvail Agents					
	services-Permitted	ClearConn SendUI	services-Permitted	ClearConn, SendUI					
1) The call begins to alert at D3 and D4	DeliveredEvent		DeliveredEvent		DeliveredEvent		DeliveredEvent		Please note: No Deflect is allowed for D3 because this is a MultiAlert-situation
	Connect-ion	H2C1	Connect-ion	H2C1	Connect-ion	D3C1	Connect-ion	D4C1	
	alertingDevice	H2	alertingDevice	H2	alertingDevice	D3	alerting-Device	D4	
	callingDevice	D1	callingDevice	D1	callingDevice	D1	calling-Device	D1	
	calledDevice	H2	calledDevice	H2	calledDevice	H2	calledDevice	H2	
	lastRedirection-Dev	NS	lastRedirection-Dev	NS	lastRedirection-Dev	NS	lastRedirect-ionDev	NS	
	localConnect-ionInfo	connected	localConnect-ionInfo	alerting	localConnect-ionInfo	alerting	localConnect-ionInfo	alerting	
	cause	MultiAlert	cause	MultiAlert	cause	MultiAlert	cause	MultiAlert	
	services-Permitted	ClearConn SendUI	services-Permitted	SendUI	services-Permitted	Answer ClearConn SendUI	services-Permitted	Answer ClearConn SendUI	
1) D4 answers call - HG-device leaves the call	Connect-ion Cleared		Connect-ion Cleared						
	droppedConnect-ion	H2C1	droppedConnect-ion	H2C1					
	releasing-Device	H2	releasingDevice	H2					
	localConnect-ionInfo	connected	localConnect-ionInfo	null					
	cause	MultiAlert	cause	MultiAlert					

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Com
	services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI	services-Permitted	none					
1) D4 is connected to the original call	Established						Established		
	establishedConn	D4C1					establishedConn	D4C1	
	answeringDevice	D4					answering-Device	D4	
	callingDevice	D1					callingDevice	D1	
	calledDevice	H2					calledDevice	H2	
	lastRedirection-Dev	NS					lastRedirect-ionDev	NS	
	localConnect-ionInfo	connected					localConnect-ionInfo	connected	
	cause	normal					cause	normal	
	services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI					services-Permitted	ClearConn Consult Hold SST GenDg GenTelTon SendUI	
1) Alerting connection cleared on D3.					Connect-ion Cleared				
					droppedConnect-ion	D3C1			
					releasingDevice	D3			
					localConnect-ionInfo	Alerting			
					cause	CallNotAnswered			
					services-Permitted	none			
					Connect-ion Cleared				
					droppedConnect-ion	D3C1			
					releasingDevice	D3			
					localConnect-ionInfo	null			
					cause	normalClr			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Comments
					services-Permitted	none			

### 5.17.4.3 Internal call to Hunt Group, Hunt Advance

D1 calls H2 (HG; H2 pilot-number ). HG distributes the call to its member D3.  
D3 does not answer; HG performs a Hunt Advance to D4. D4 answers the call.

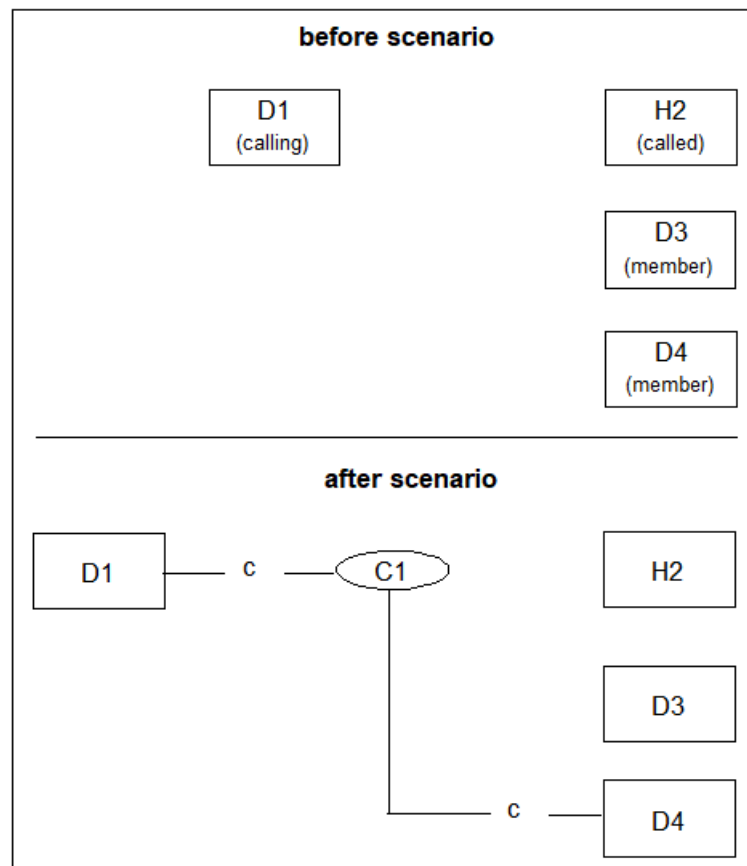


Figure 75: Internal call to hunt group, hunt advance

Table 296: Internal call to HG - Hunt-Advance

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Comments
1) D1 goes off-hook	Service Initiated								
	• initiated-Connection	D1C1							
	• initiatingDevice	D1							
	• localConnect-ionInfo	initiated							
	• cause	normal							

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Comments
	• services-Permitted	ClearConn, DialDg							
1) D1 completes dialling HG-access-code (1234)	Digits Dialed								
	• diallingConnection	D1C1							
	• diallingDevice	D1							
	• diallingSequence	"1234"							
	• localConnectionInfo	initiated							
	• cause	normal							
	Originated								
	• originatedConnection	D1C1							
	• callingDevice	D1							
	• calledDevice	H2 pilot							
	• lastRedirect-ionDev	NS							
	• localConnect-ionInfo	connected							
	• cause	normal							
	• services-Permitted	ClearConn							
1) The call hits the Hunt-Group-Device	Delivered		Delivered						
	• connection	H2C1	• connection	H2C1					
	• alertingDevice	H2	• alertingDevice	H2					
	• callingDevice	D1	• callingDevice	D1					
	• calledDevice	H2 pilot	• calledDevice	H2 pilot					
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS					
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting					
	• cause	enterDist	• cause	enterDist					
1) HG distributes the call to HG-member D3	• services-Permitted	ClearConn SendUI	• services-Permitted	SendUI					
	Delivered		Delivered		Delivered				Please not
	• connection	D3C1	• connection	D3C1	• connection	D3C1			No Deflect
	• alertingDevice	D3	• alertingDevice	D3	• alertingDevice	D3			allowed for
	• callingDevice	D1	• callingDevice	D1	• callingDevice	D1			D3 because
	• calledDevice	H2 pilot	• calledDevice	H2 pilot	• calledDevice	H2 pilot			this is a M
									Alert-situat

## Call Scenarios

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Comments
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS			
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting	• localConnect-ionInfo	alerting			
	• cause	multiAlert	• cause	multiAlert	• cause	multiAlert			
	• services-Permitted	ClearConn SendUI	• services-Permitted	SendUI	• services-Permitted	Answer ClearConn SendUI			
1) D3 did not answer the call HG performs a Hunt Advance: D3 is cleared from the call.	<b>Connection Cleared</b>		<b>Connection Cleared</b>		<b>Connection Cleared</b>				
	• dropped-Connection	D3C1	• dropped-Connection	D3C1	• dropped-Connection	D3C1			
	• releasingDevice	D3	• releasingDevice	D3	• releasingDevice	D3			
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting	• localConnect-ionInfo	null			
	• cause	normalClr	• cause	normalClr	• cause	normalClr			
	• services-Permitted	ClearConn SendUI	• services-Permitted	SendUI	• services-Permitted	none			
1) The next HG-member D4 is alerted	Delivered		Delivered				Delivered		
	• connection	D4C1	• connection	D4C1			• connection	D4C1	
	• alertingDevice	D4	• alertingDevice	D4			• alertingDevice	D4	
	• callingDevice	D1	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	H2 pilot	• calledDevice	H2 pilot			• calledDevice	H2 pilot	
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS			• lastRedirect-ionDev	NS	
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting			• localConnect-ionInfo	alerting	
	• cause	multiAlert	• cause	multiAlert			• cause	multiAlert	
	• services-Permitted	ClearConn SendUI	• services-Permitted	ClearConn SendUI			• services-Permitted	Answer ClearConn SendUI	

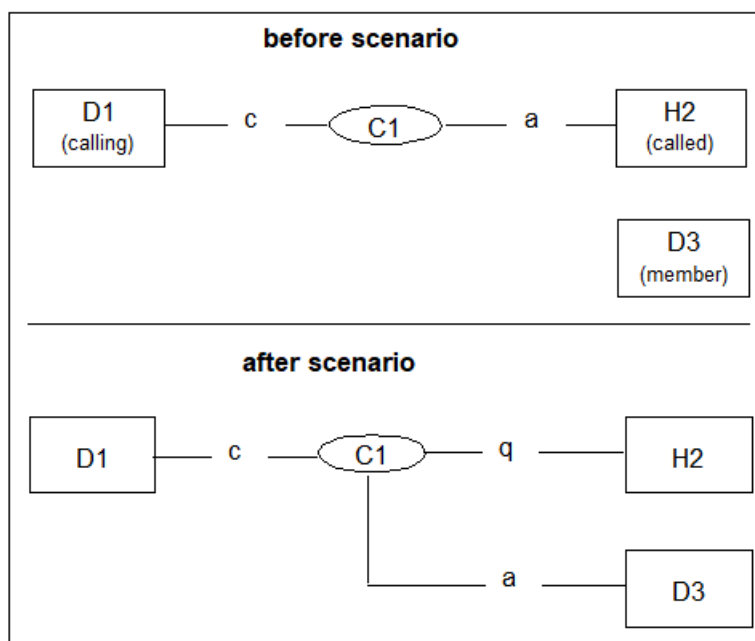
Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Monitored Device D4		Comments
1) D4 answers call - HG-device withdraws from the call	<b>Connection Cleared</b>		<b>Connection Cleared</b>				<b>Connection Cleared</b>		HG withdraws from the call as soon as the member answers the call
	• dropped-Connection	H2C1	• dropped-Connection	H2C1			• dropped-Connection	H2C1	
	• releasingDevice	H2	• releasingDevice	H2			• releasingDevice	H2	
	• localConnect-ionInfo	connected	• localConnect-ionInfo	null			• localConnect-ionInfo	alerting	
	• cause	multiAlert	• cause	multiAlert			• cause	multiAlert	
1) D4 is connected to D1	• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI	• services-Permitted	none			• services-Permitted	ClearConn Consult Hold SST GenDg GenTelTon SendUI	
	Established						Established		
	• establishedConn	D4C1					• established-Conn	D4C1	
	• answeringDevice	D4					• answeringDevice	D4	
	• callingDevice	D1					• callingDevice	D1	
	• calledDevice	H2 pilot					• calledDevice	H2 pilot	
	• lastRedirect-ionDev	NS					• lastRedirect-ionDev	NS	
	• localConnect-ionInfo	connected					• localConnect-ionInfo	connected	
	• cause	normal					• cause	normal	
	• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI					• services-Permitted	ClearConn Consult Hold SST GenDg GenTelTon SendUI	

**Remark:**

None

#### 5.17.4.4 Call is queued at Hunt Group

D1 calls H2 (HG; H2 pilot-number). No members are available - the call is queued. Eventually HG-member D3 becomes available. HG distributes the call to D3.



**Figure 76: Call is queued at hunt group**

Please refer to section [Section 5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#) for the event flow that leads to the "before"-state.

**Table 297: Internal Call to HG - call is queued**

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3 (member)		Comments
1) No member is available in the HG, the call is queued	Queued		Queued				Deflect:
	• queuedConnection	H2C1	• queuedConnection	H2C1			This is the only situation where Deflect from a HG is possible
	• queue	H2	• queue	H2			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	H2 pilot	• calledDevice	H2 pilot			
	• lastRedirectionDev	NS	• lastRedirectionDev	NS			
	• localConnectionInfo	connected	• localConnectionInfo	queued			
	• cause	NoAgents	• cause	NoAgents			
	• servicesPermitted	ClearConn SendUI	• servicesPermitted	Deflect SendUI			

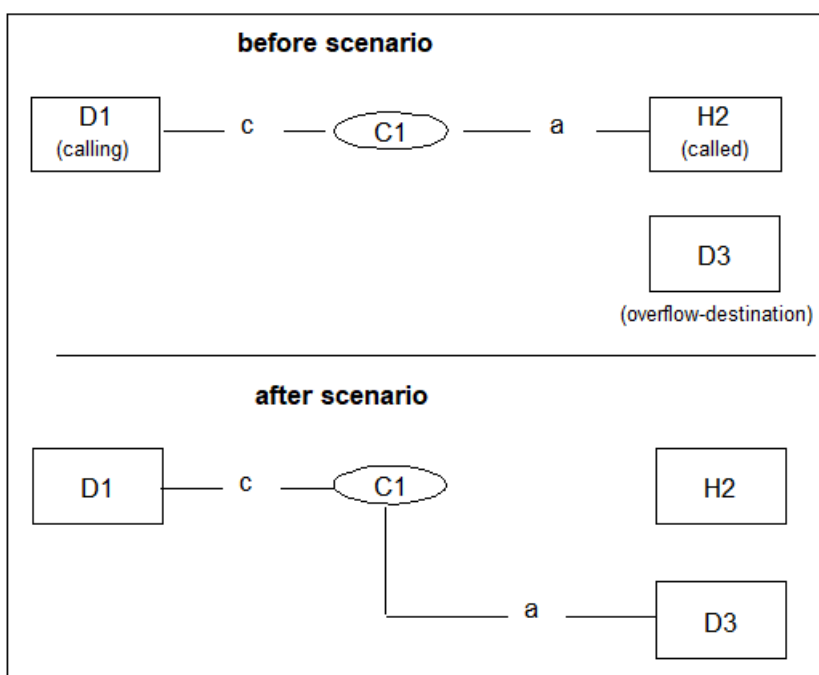
Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3 (member)		Comments
1) HG-member D3 becomes available; HG distributes the call to D3.	Delivered		Delivered		Delivered		remainsInQ:
	• connection	D3C1	• connection	D3C1	• connection	D3C1	This cause is only provided if the HG is monitored. If HG is not monitored, event-cause"multiAlert" will be provided instead.
	• alertingDevice	D3	• alertingDevice	D3	• alertingDevice	D3	
	• callingDevice	D1	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	H2 pilot	• calledDevice	H2 pilot	• calledDevice	H2 pilot	
	• lastRedirectionDev	NS	• lastRedirectionDev	NS	• lastRedirectionDev	NS	
	• localConnectionInfo	connected	• localConnectionInfo	queued	• localConnectionInfo	alerting	
	• cause	remainsInQ	• cause	remainsInQ	• cause	remainsInQ	
	• servicesPermitted	ClearConn SendUI	• servicesPermitted	SendUI	• servicesPermitted	Answer ClearConn SendUI	

**Remark:**

None

**5.17.4.5 Call is routed to Overflow Destination**

D1 calls H2 (HG; H2 pilot-number). No member is available, no queue is configured for the HG. The call is redirected immediately to the overflow-destination.

**Figure 77: Call is routed to overflow destination**

## Call Scenarios

Please refer to section [Section 5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#) for the event flow that leads to the "before"-state.

**Table 298: Internal Call to HG - redirected to overflow-destination**

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3		Comments
1) HG diverts the call to the overflow-destination			Diverted				
			• connection	H2C1			
			• divertingDevice	H2			
			• newDestination	D3			
			• callingDevice	D1			
			• calledDevice	H2 pilot			
			• lastRedirectionDev	NS			
			• localConnectionInfo	null			
			• cause	overflow			
			• servicesPermitted	none			
1) The overflow-destination is rung	Delivered				Delivered		
	• connection	D3C1			• connection	D3C1	
	• alertingDevice	D3			• alertingDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	H2 pilot			• calledDevice	H2 pilot	
	• lastRedirectionDev	H2			• lastRedirectionDev	H2	
	• localConnectionInfo	connected			• localConnectionInfo	alerting	
	• cause	overflow			• cause	overflow	
	• servicesPermitted	ClearConn CallBack SendUI			• servicesPermitted	Answer ClearConn Deflect SendUI	

### Remark:

Please note that the call-scenario is different if a queue is configured for the HG, but the queue is full: in that case, HiPath 4000 treats the scenario as if the HG was forwarded immediately to the overflow-destination.

### 5.17.4.6 Transfer Ringing into Hunt Group

D1 is in a two-party conversation with D2 (Call-Id C1) and has initiated a consultation to H3 (HG; H3-pilot-number) (Call-Id C2), HG rings its member D4. While both H3 and D4 are ringing (Multi-Alert), D2 transfers the call (Call-Id C3).

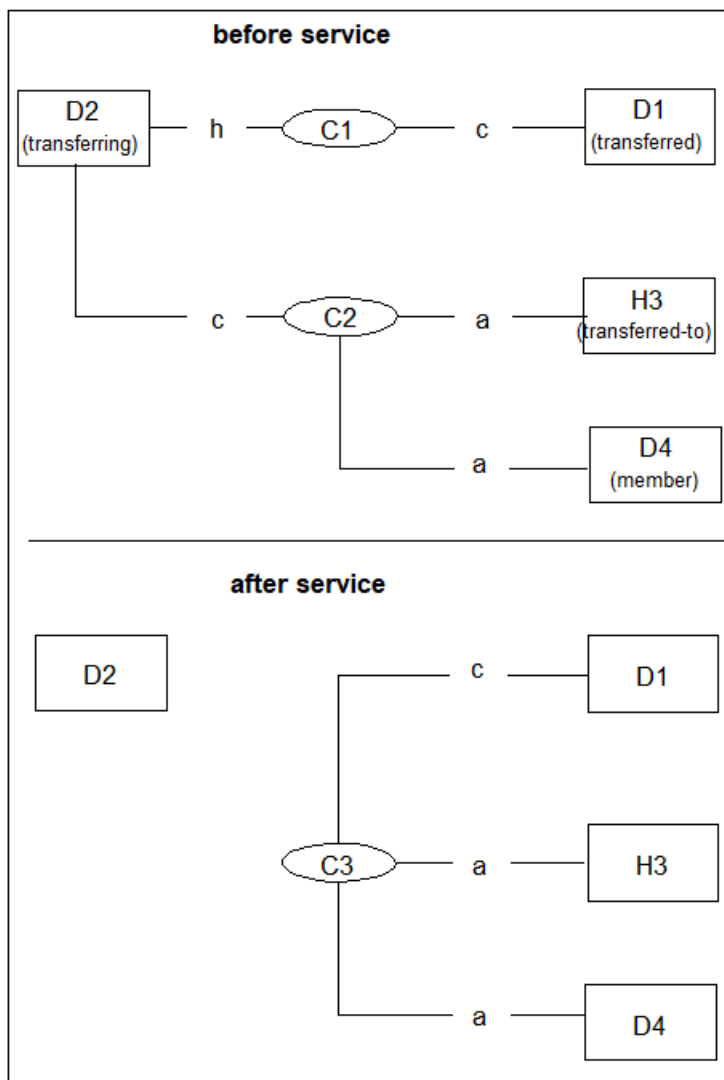


Figure 78: Transfer ringing into hunt group

**Table 299: Consultation to HG - Transfer ringing**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device H3 (HG)		Monitored Device D4 (member)		Comments
1) D2 transfers the call	Transferred		Transferred		Transferred		Transferred		Conn-List:
	• primary-OldCall	D1C1	• primary-OldCall	D2C1	• primary-OldCall	H3C2	• primary-OldCall	D4C2	For the HG (H3) and its member D4, both H3 and D4 are shown as old Conn-Ids in the list, because they belong to the same call
			• secondary-OldCall	D2C2					
	• transferring-Device	D2	• transferring-Device	D2	• transferring-Device	D2	• transferring-Device	D2	
	• transferred-Device	H3	• transferred-Device	H3	• transferred-Device	H3	• transferred-Device	H3D3	
	• TransferConnList 1.new / old 2. new 3. new	(D1C3)/ (D1C1) (D4C3) (H3C3)	• TransferConnList: 1.new / old 2. new / old 3. new / old	(D1C3)/ (D1C1) (D4C3)/ (D4C2) (H3C3)/ (H3C2)	• TransferConnList: 1.new 2. new / old 3. new / old	(D1C3) (D4C3)/ (D4C2) (H3C3)/ (H3C2)	• TransferConnList: 1.new 2. new/ old 3. new/old	(D1C3) (D4C3)/ (D4C2) (H3C3)/ (H3C2)	
	• localConnection-Info	connected	• localConnection-Info	null	• localConnection-Info	alerting	• localConnection-Info	alerting	
	• cause	Transfer	• cause	Transfer	• cause	Transfer	• cause	Transfer	
	• services-Permitted	ClearConn, SendUI	• services-Permitted	none	• services-Permitted	SendUI	• services-Permitted	Answer ClearConn SendUI	

**Remark:**

If D4 does not answer the call, HiPath 4000 will not perform a Transfer-Recall as would be the case in a Transfer-scenario without a Group-Device.

### 5.17.4.7 Pick from Hunt Group Member

D1 calls H2 (HG; H2 pilot-number), HG distributes the call to its member D3. While D3 is ringing, D4 picks the call.

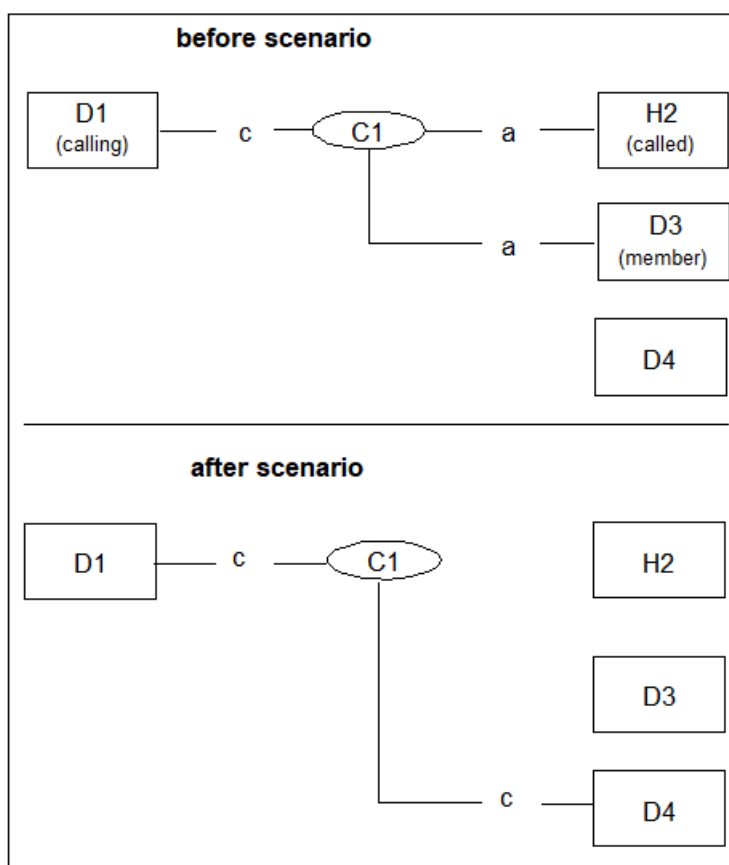


Figure 79: Pick from hunt group member

Please refer to section [Section 5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#) for the event flow that leads to the "before"-state.

Table 300: Pick from HG-member

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3 (member)		Monitored Device D4		Comments
1) D4 picks call from D3 - First, the Hunt-Group-Device leaves the call	<b>Connection Cleared</b>		<b>Connection Cleared</b>		<b>Connection Cleared</b>				
	• dropped-Connection	H2C1	• dropped-Connection	H2C1	• dropped-Connection	H2C1			
	• releasingDevice	H2	• releasingDevice	H2	• releasingDevice	H2			
	• local-Connection-Info	connected	• local-Connection-Info	null	• local-Connection-Info	alerting			
	• cause	normalClr	• cause	normalClr	• cause	normalClr			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device H2 (HG)		Monitored Device D3 (member)		Monitored Device D4		Comments
	• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI	• services-Permitted	none	• services-Permitted	none			
1) The call is diverted from member D3 to D4					Diverted				
					• connection	D3C1			
					• divertingDevice	D3			
					• newDestination	D4			
					• callingDevice	D1			
					• calledDevice	H2 pilot			
					• lastRedirect-ionDev	NS			
					• local-Connection-Info	null			
					• cause	pick			
					• servicesPermitted	none			
1) D4 is connected to D1	Established						Established		
	• establishedConn	D4C1					• establishedConn	D4C1	
	• answeringDevice	D4					• answeringDevice	D4	
	• callingDevice	D1					• callingDevice	D1	
	• calledDevice	H2 pilot					• calledDevice	H2 pilot	
	• lastRedirect-ionDev	D3					• lastRedirect-ionDev	D3	
	• local-Connection-Info	connected					• local-Connection-Info	connected	
	• cause	pick					• cause	pick	
	• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI					• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI	

**Remark:**

None

## 5.17.5 General Attendant (GA)

### 5.17.5.1 General description GA

#### Introduction

Monitoring of General Attendant (former "Attendant Queue") was provided in a non-standard way in previous versions. Please note that the call-flow has changed considerably!

CSTA III (ECMA 269) and the ECMA Call-Scenarios do not provide much information about how devices like GA should be modeled. Therefore it is necessary to describe the model CA 4000 uses.

#### Characteristics of GA

The basic characteristics for GA are the same as for the HG. Please refer to section [Characteristics of Hunt Group on page 520](#) for more information. The mask used for monitoring is 0x04000000.

#### Deflect and GA

The following has to be considered when using the Deflect-Call-Request for calls where GA is involved:

- Deflect is never allowed from the GA
- Deflect-Call is not allowed in a Multiple-Alert-Situation (this includes all devices involved in the call).

#### Different types of GA

There are two different types of GA. They differ in how the calls are distributed to their members (Attendant Console or ACs) and the number of queues provided:

- GA2Q
- 2Q means "Double Queue". It provides 2 queues:
  - for external calls (German: "Amt")
  - for internal calls (German: "Melde")

The distributing mechanism is similar to the Hunt-Groups - please refer to section [Section 5.17.4.1, "General description Hunt Group"](#) for more details.

- GAMQ

- MQ means "Multi Queue". This GA can have up to 12 queues configured. It depends upon HiPath 4000-configuration, which calls are placed in which queue.

The calls are not distributed by GAMQ, but every call is queued. The ACs pick the calls from the queue. This results in a different CSTA-event-flow when compared with GA2Q:

- Because all calls are queued in GAMQ, the event-cause for the Queued-Event is always "normal" (not "noAvailAgents" like for GA2Q or HG).
- Diverted-Event for GA when AC picks up call (event-cause = distributed) (the AC is a new destination for the call, therefore a Diverted-Event).
- Delayed Delivered-Event for GA after picking up (with LastRedir = GA), immediately followed by an Established-Event

Please refer to section [Section 5.17.5.2, "Internal call to GA2Q"](#) for more information.

### Special Features of GA

The following features are similar to HG and therefore not described in detail:

- Members: the members of a GA are Attendant Consoles (AC or ATC)
- Control of the call: like for HG, the GA remains in control of the call until the call is either successfully distributed or torn down. One exception is night-service: there are certain types of night-service where GA leaves the call as soon as the night-destination starts ringing (for more details refer to section [5.17.5.1, "Night-service"](#)).
- Queues: A GA usually has more than one queues. A Deflect from the GA-Queue is never permitted. (Refer to [GA2Q](#) and [GAMQ](#) for more information)

The following features are special GA-features:

- Intercept
- Intercept is an important feature for the GA. Here a few examples when intercept may occur:
  - A calls B, B does not answer; the call is intercepted to GA
  - without parallel call: B stops ringing (practically a diversion of the call)
  - with parallel call: after the call has been intercepted to GA, B keeps ringing. If it is an GA2Q, 3 devices may be ringing at the same time: B, GA and AC. Whoever answers the call first (B or AC) is connected to the call, all other connections are cleared.
  - A calls B, B is busy; the call is intercepted to GA
  - A dials an invalid extension, an incomplete extension or no extension at all; the call is intercepted to GA. Please note: the dialled-digits will be shown as CalledDevice in all subsequent events. If no digits were dialled (no extension), CallBridge will report "NotKnown".

When and if a call is intercepted depends upon HiPath 4000-configuration. Because CSTA III does not provide an appropriate event-cause for intercept, an application must infer from the event-flow that intercept has occurred. (See example-event-flows in [Section 5.17.5.5](#) to [Section 5.17.5.8](#)).

- Night-service

- After the last AC goes out-of-service, the GA switches to night-mode. There are different kinds of night-service (the desired kind of night-service can be configured on the switch):
  - Centralized attendant internal (ZVFINT) The GA (GA1) is forwarded to another GA (GA2) in the same node. GA1 leaves the call as soon as the call is diverted to GA2.
  - Centralized attendant external (ZVFEXT) The GA (GA1) is forwarded to another GA (GA2) in another node. GA1 remains involved in the call until an AC on GA2 answers the call.
  - Local night station(s) The GA forwards incoming calls to nightstations (DIGITEs, ANATEs) in the same node. If all night stations are busy, the call remains in the queue of the GA. As soon as the night-station starts ringing, the GA is not involved in the call any more. This is different to day-service, when GA remains involved until the AC answers the call
  - Universal night answer ("Allgemeines Abfragen") A special UNA-device (e.g. a bell) is configured, to which all incoming calls are forwarded. Subscribers (e.g. DIGITEs) can pick up calls from the UNA-device. The UNA-device is a so-called "specialDevice" - this device is represented by a device-number with the mask 0x0a000000. Please note: this specialDevice cannot be monitored!. GA leaves the call as soon as the UNA-device starts ringing.
  - No destination configured ("Leervariante") Incoming calls are queued in GA and remain queued until they either hang up, or an AC becomes available (day service is started)
  - Trunk Night Service (TNS) A specific trunk has its own night destination configured. An incoming call is forwarded to this destination (on the same node or another node). In this case, the GA is only involved in the call if the individual TNS destination is busy.
- Recalls to GA
- There are different reasons why a recall to GA occurs:
  - Serial Recall: Example: A and B are connected, the connection was formerly established by an AC using the serial call feature. When B goes onhook, A (=calling party) seizes the GA immediately again.
  - Trunk to trunk supervision Example: If two trunks without disconnect supervision are connected, HiPath 4000 cannot clear this call properly. Therefore HiPath 4000 starts a timer. Whenever this timer expires, a recall to the GA is started. The AC listens in on the call and decides whether to disconnect the call or not.
  - Transfer Recall: Example: A calls GA, an AC transfers the call to B. B does not answer - the GA is recalled.
  - Park Recall: Example: An AC parks a call with B (directed call park). The Park-Timer expires, B starts ringing. The Recall-Timer expires - the GA is recalled
- Personal calls
- All personal calls to the AC are handled via the GA. In previous switch-versions, personal calls to the AC did not involve the group-device.
- De-Queueing of calls
- GA is capable of removing calls from its queue without tearing down the call. Whenever GA de-queues a call, the calling party is in the state "Waits for place in queue". This state is similar to the originated state. A Conn-

Cleared-Event is sent for the monitor of GA and the calling device to show this situation.

When the call is finally accepted by GA, a Delivered-Event is sent. OpenScape 4000 has lost information of the history of the call - it will be shown as if the call has directly dialled the GA (no LastRedirectionDevice, no special event-cause, etc.).

- Diversion to GA fails
- When all ACs are busy, no place is available in the queue and no overflow-destination is configured, GA rejects calls that are diverted. After the rejection of GA, the call is in the state "Waits for place in queue".

Here is a list of some situations where this might happen:

- A ringing at B, recall to GA, no parallel call, GA rejects call
- A ringing at B, CF-NA to GA, no parallel call, GA rejects call
- A ringing at B, Intercept to GA, no parallel call, GA rejects call

In these cases, CA 4000 sends a Diverted-Event for the diverting party (B). No event is generated for the calling party (A). The next event the calling party will receive is the Delivered-Event when the call is finally accepted by GA (in this special case, LastRedirectionDevice will be NS because the history of the call is no longer available for HiPath 4000).

- Speed extend from Attendant Console
- Attendant console is able to speed transfer a connected call. The event flow is modelled like a single step transfer, without request and response involved. There are no events showing the speed dial of the destination. This is valid also for Attendant Console Light. In case of unsuccessful speed extend the call remains as it was before, no events will be provided unless the call was offered to the destination. The call flow for the speed extend to a busy destination with offered mode activated is the same as the single step transfer attempt to a busy destination with offered mode activated.

### Known Restrictions for GA

The restrictions are the same as for HG, please refer to section [Known Restrictions for Hunt Group on page 521](#).

### General Rules concerning Multi-Alert-Situations for GA

The same general rules as for HG apply for GA (please refer to section [5.17.5.1, "General Rules concerning Multi-Alert-Situations for GA"](#)). Additional situations where Multi-Alert may occur for GA are:

- Intercept with parallel call to the GA. Both the intercepted device and the GA are ringing. If GA distributes the call to one of its members, 3 devices are ringing at the same time. (see section [5.17.5.6, "Intercept with parallel call to GA2Q"](#))
- Recall to the GA with parallel call: a call that has been transferred by an AC is not answered => a recall to the GA is performed

Additional rules for GA:

- 1) Whenever a device leaves a call and the call remains in a Multiple-Alerting-Situation, a Conn-Cleared-Event must be sent.
- 2) Example:

Intercept with parallel call, three devices are ringing simultaneously (B, GA and AC), AC goes out-of-service, therefore the AC is dropped from the call

- 3) If a device leaves a call, leaving the call in Multiple-Alerting-Situation and another device joins the call in one step (e.g. overflow from one GA to another GA during Multiple-Alerting), this is not shown as a Diversion, but as one device leaving the call (Conn-Cleared) and another device added to the call (Delivered-Event with LastRedirection=NS and event-cause=MultiAlert). It would be wrong to send a Diverted-Event, because this would violate the rule that a diversion has not taken place when event-cause is Multi-Alert (see rule # 2 in section [General Rules concerning Multi-Alert-Situations on page 522](#)). The remaining devices need to be informed that a device has left the call - this is done by sending a Connection-Cleared event to all remaining devices.

4) Example:

A calls B, call is intercepted to GA1 with parallel call, no AC is free, the call is queued at GA1. After an overflow timer expires, the call is diverted to GA2. In this case, a Conn-Cleared-Event is sent for GA1 and Delivered-Event for GA2, but no Diverted-Event (see section [5.17.5.8, "Intercept with parallel call, call is routed to another GA after timeout"](#))

### 5.17.5.2 Internal call to GA2Q

The scenario is identical to the HG-scenario; please refer to [Section 5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#) for more information (please note that a Hunt-Advance as shown in this scenario is usually not performed at a GA-device).

### 5.17.5.3 Internal call to GAMQ

D1 calls G2 (GAMQ, internal attendant access code: G2-int), the call is queued. D3 (AC) picks the call from the queue.

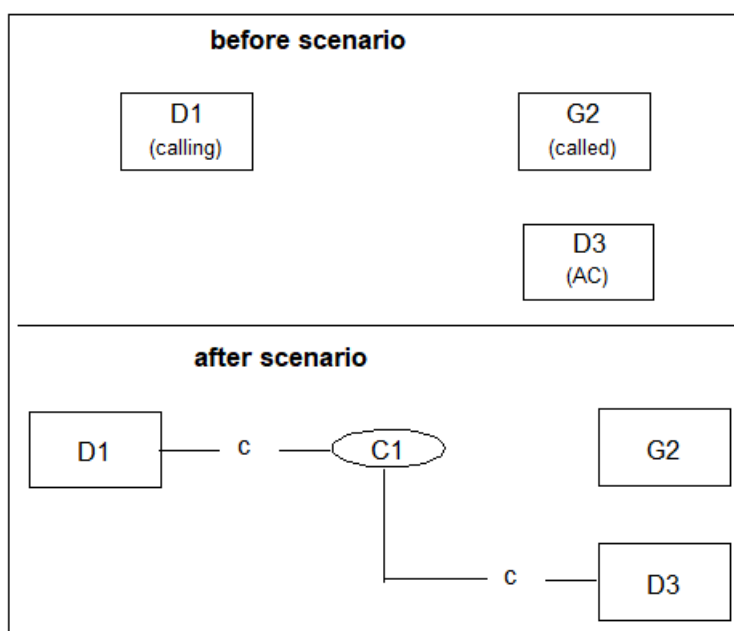


Figure 80: Internal call to GAMQ

## Call Scenarios

**Table 301: Internal Call to GAMQ**

Activity	Monitored Device D1		Monitored Device G2 (GAMQ)		Monitored Device D3 (AC)		Comments
1) D1 goes off-hook.	Service Initiated						
	• initiatedConnection	D1C1					
	• initiatingDevice	D1					
	• localConnectionInfo	initiated					
	• cause	normal					
1) D1 completes dialling the internal attendant access code.. (1234)	• servicesPermitted	ClearConn DialDg					
	Digits Dialed						
	• diallingConnection	D1C1					
	• diallingDevice	D1					
	• diallingSequence	"1234"					
	• localConnectionInfo	initiated					
	• cause	normal					
	• servicesPermitted	none					
	Originated						
	• originatedConnection	D1C1					
	• callingDevice	D1					
	• calledDevice	G2-int					
	• lastRedirectionDev	NS					
	• localConnectionInfo	connected					
	• cause	normal					
	• servicesPermitted	ClearConn					
1) The call hits the GAMQ .	Delivered		Delivered				
	• connection	G2C1	• connection	G2C1			
	• alertingDevice	G2	• alertingDevice	G2			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	G2-int	• calledDevice	G2-int			
	• lastRedirectionDev	NS	• lastRedirectionDev	NS			
	• localConnectionInfo	connected	• localConnectionInfo	alerting			
	• cause	enterDist	• cause	enterDist			

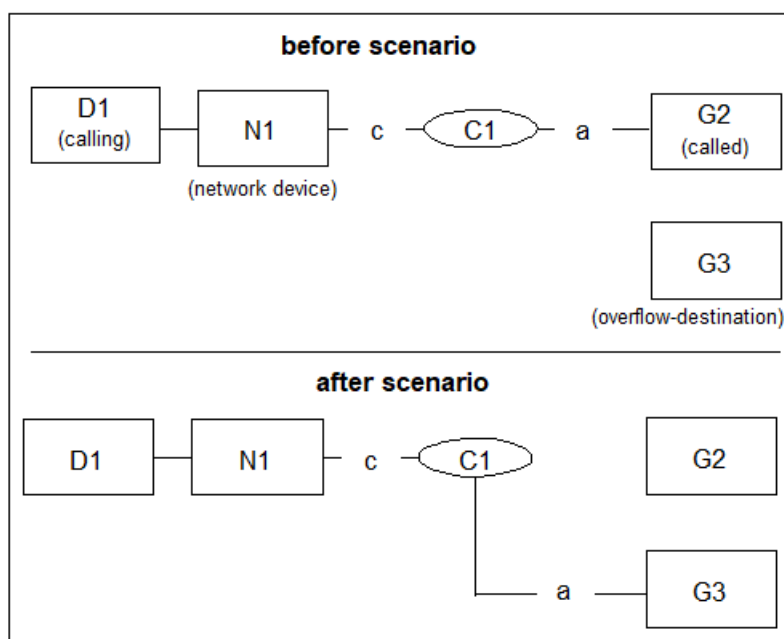
Activity	Monitored Device D1		Monitored Device G2 (GAMQ)		Monitored Device D3 (AC)		Comments
	• servicesPermitted	ClearConn SendUI	• servicesPermitted	Send-UserInfo			
<b>1)</b> The call is queued at GAMQ  <b>2)</b> Please note: all calls are queued at GAMQ. GAMQ does not distribute calls, but the ACs pick the calls from the queue	Queued		Queued				cause = normal: because all calls are queued at GAMQ, event-cause is "normal" instead of "noAgents"
	• queuedConnection	G2C1	• queuedConnection	G2C1			
	• queue	G2	• queue	G2			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	G2-int	• calledDevice	G2-int			
	• lastRedirectionDev	NS	• lastRedirectionDev	NS			
	• localConnectionInfo	connected	• localConnectionInfo	queued			
	• cause	normal	• cause	normal			
	• servicesPermitted	ClearConn SendUI	• servicesPermitted	SendUI			
<b>1)</b> AC picks the call from the queue - first, the call is diverted from the GAMQ			Diverted				
			• connection	G2C1			
			• divertingDevice	G2			
			• newDestination	D3			
			• callingDevice	D1			
			• calledDevice	G2-int			
			• lastRedirectionDev	NS			
			• localConnectionInfo	null			
			• cause	distributed			
<b>1)</b> The call arrives at the AC.	Delivered				Delivered		A delayed Delivered-Event is sent for the AC after the AC picks the call.
	• connection	D3C1			• connection	D3C1	
	• alertingDevice	D3			• alertingDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	G2-int			• calledDevice	G2-int	
	• lastRedirectionDev	G2			• lastRedirection-Dev	G2	
	• localConnectionInfo	connected			• localConnection-Info	alerting	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device G2 (GAMQ)		Monitored Device D3 (AC)		Comments
	• cause	distributed			• cause	distrib- uted	
	• servicesPermitted	ClearConn Consult Hold SST GenDg GenTelTon SendUI			• servicesPermittedSendUI		
1) The call is established.	Established				Established		
	• establishedConn	D3C1			• establishedConn	D3C1	
	• answeringDevice	D3			• answeringDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	G2-int			• calledDevice	G2-int	
	• lastRedirectionDev	G2			• lastRedirection- Dev	G2	
	• localConnectionInfo	connected			• localConnection- Info	con- nected	
	• cause	normal			• cause	normal	
	• servicesPermitted	ClearConn Consult Hold SST GenDg GenTelTon SendUI			• services- Permitted	SendUI	

### 5.17.5.4 Overflow from one GA to another GA

External caller D1 calls G2 (GAMQ, external attendant access code: G2-ext). The queue of GAMQ is full, G3 (GA2Q) is configured as overflow destination. The call is immediately forwarded to GA2Q (before it is queued at the GAMQ).



**Figure 81: Overflow from one GA to another GA**

Please refer to section [Section 5.17.5.3, "Internal call to GAMQ"](#) for the event flow that leads to the "before"-state.

**Table 302: Overflow from one GA to another GA**

Activity	Monitored Device N1		Monitored Device G2 (GA1)		Monitored Device G3 (GA2)		Comments
1) The queue of D2 is full, the call is diverted immediately to the overflow destination D3			Diverted				
			• connection	G2C1			
			• divertingDevice	G2			
			• newDestination	D3			
			• callingDevice	D1			
			• calledDevice	G2-ext			
			• AssCallingDevice	N1			
			• NWCcallingDevice	D1			
			• lastRedirectionDev	NS			
			• localConnectionInfo	null			
			• cause	over-flow			
			• servicesPermitted	none			

## Call Scenarios

Activity	Monitored Device N1		Monitored Device G2 (GA1)		Monitored Device G3 (GA2)		Comments
1) Overflow-destination D3 is rung	Delivered				Delivered		
	• connection	G3C1			• connection	G3C1	
	• alertingDevice	G3			• alertingDevice	G3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	G2-ext			• calledDevice	G2-ext	
	• OrigNIDConn	N1C1			• OrigNIDConn	N1C1	
	• NWCcallingDevice	D1			• NWCcallingDevice	D1	
	• AssCallingDevice	N1			• AssCallingDevice	N1	
	• lastRedirectionDev	G2			• lastRedirectionDev	G2	
	• localConnectionInfo	connected			• localConnectionInfo	alerting	
	• cause	overflow			• cause	overflow	
	• servicesPermitted	ClearCall SendUI			• servicesPermitted	SendUI	

### Remark:

None

### 5.17.5.5 Intercept without parallel call to GA2Q

An external caller (directory-number D1) calls D2 via the network-interface N1 (trunk). D2 does not answer the call. After a timeout, the call is intercepted without parallel call to the GA (G3). GA distributes the call to the AC D4.

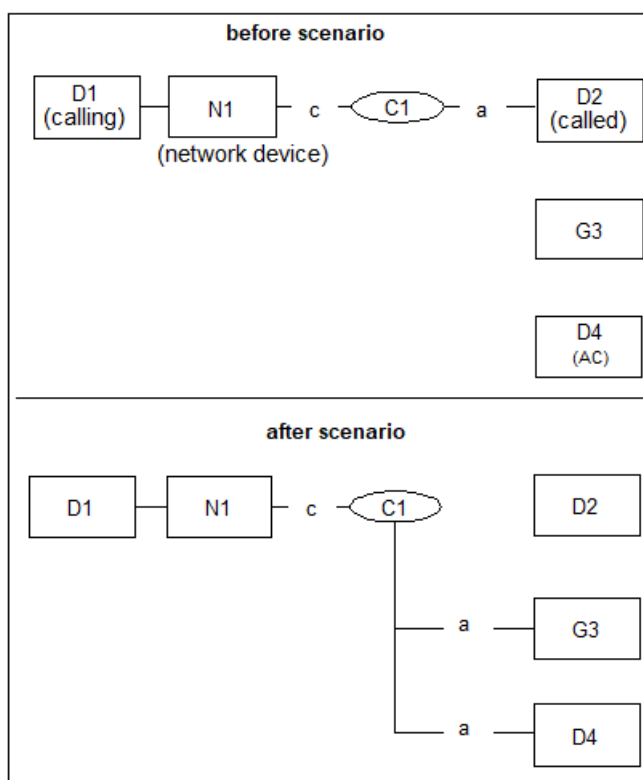


Figure 82: Intercept without parallel call to GA2Q

Table 303: Intercept without parallel call to GA2Q

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
1) The call was previously ringing at D2. Intercept-timer has expired, the call is intercepted without parallel call to the GA			Diverted						
			• connection	D2C1					
			• divertingDevice	D2					
			• newDestination	G3					
			• calledDevice	D2					
			• last-RedirectionDev	NS					
			• localConnectionInfo	null					
			• cause	callNot-Answered					
			• services-Permitted	none					
1) The GA is alerted	Delivered				Delivered				
	• connection	G3C1			• connection	G3C1			
	• alertingDevice	G3			• alertingDevice	G3			

## Call Scenarios

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
	• callingDevice	D1			• callingDevice	D1			
	• calledDevice	D2			• calledDevice	D2			
	• OrigNIDConn	N1C1			• OrigNIDConn	N1C1			
	• NWCcallingDevice	D1			• NWCcallingDevice	D1			
	• AssCallingDevice	N1			• AssCallingDevice	N1			
	• last-RedirectionDev	D2			• last-RedirectionDev	D2			
	• localConnect-ionInfo	connected			• localConnect-ionInfo	alerting			
	• cause	enterDist			• cause	enterDist			
	• services-Permitted	ClearCall SendUI			• services-Permitted	SendUI			
1) GA2Q distributes call to AC	Delivered				Delivered		Delivered		
	• connection	D4C1			• connection	D4C1	• connection	D4C1	
	• alertingDevice	D4			• alertingDevice	D4	• alertingDevice	D4	
	• callingDevice	D1			• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	• calledDevice	D2	
	• OrigNID-Conn	N1C1			• OrigNIDConn	N1C1	• OrigNIDConn	N1C1	
	• NWCcallingDevice	D1			• NWCcallingDevice	D1	• NWCcallingDevice	D1	
	• AssCallingDevice	N1			• AssCallingDevice	N1	• AssCallingDevice	N1	
	• last-RedirectionDev	NS			• last-RedirectionDev	NS	• last-RedirectionDev	NS	
	• localConnect-ionInfo	con- nected			• localConnect-ionInfo	alerting	• localConnect-ionInfo	alerting	
	• cause	mul- tiAlert			• cause	multiAlert	• cause	multiAlert	
	• services-Permitted	ClearConn SendUI			• services-Permitted	SendUI	• services-Permitted	SendUI	

**Remark:**

None

### 5.17.5.6 Intercept with parallel call to GA2Q

An external caller (directory-number D1) calls D2 via the network-interface N1 (trunk). D2 does not answer the call. After a timeout, the call is intercepted with parallel call to the GA (D3). GA distributes the call to AC D4.

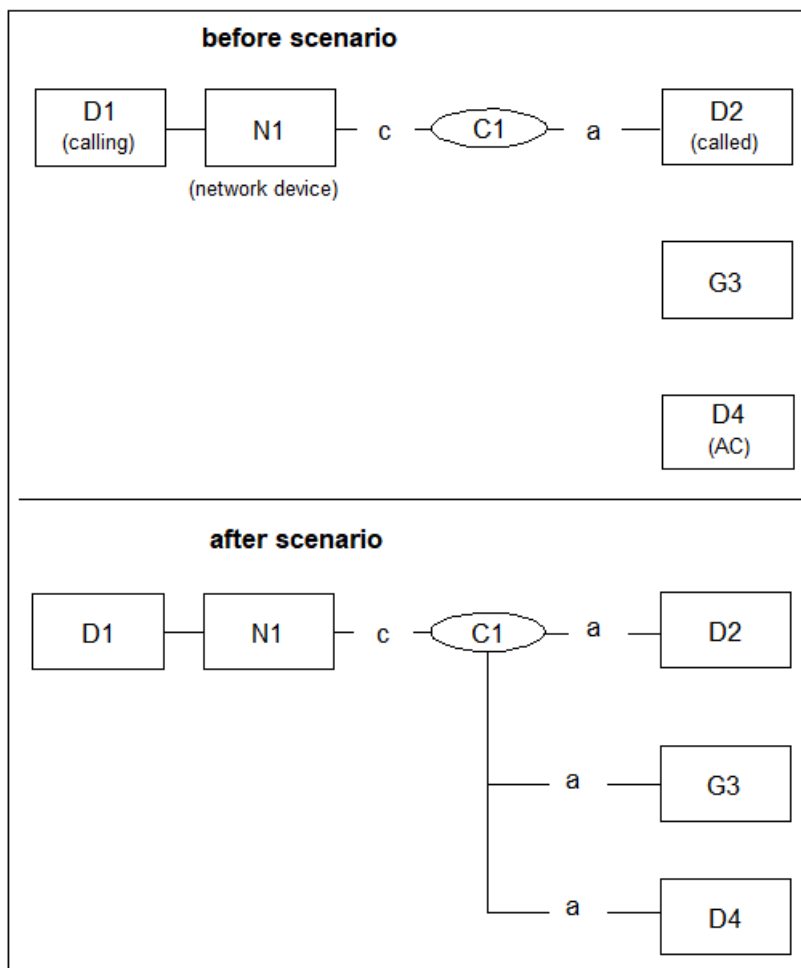


Figure 83: Intercept with parallel call to GA2Q

Table 304: Intercept with parallel call to GA2Q

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
1) The call was previously ringing at D2. Intercept-timer has expired, the call is intercepted to GA (with parallel call), the GA is alerted	Delivered		Delivered		Delivered				
	• connection	G3C1	• connection	G3C1	• connection	G3C1			
	• alerting-Device	G3	• alerting-Device	G3	• alerting-Device	G3			
	• calling-Device	D1	• calling-Device	D1	• calling-Device	D1			
	• called-Device	D2	• called-Device	D2	• called-Device	D2			
	• OrigNID-Conn	N1C1	• OrigNID-Conn	N1C1	• OrigNID-Conn	N1C1			

## Call Scenarios

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
	• NWCalling-Device	D1	• NWCalling-Device	D1	• NWCalling-Device	D1			
	• AssCalling-Device	N1	• AssCalling-Device	N1	• AssCalling-Device	N1			
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS			
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting	• localConnect-ionInfo	alerting			
	• cause	multiAlert	• cause	multiAlert	• cause	multiAlert			
	• services-Permitted	ClearCall SendUI	• services-Permitted	Answer ClearCall SendUI	• services-Permitted	SendUI			
1) GA2Q distributes call to AC	Delivered		Delivered		Delivered		Delivered		
	• connection	D4C1	• connection	D4C1	• connection	D4C1	• connection	D4C1	
	• alerting-Device	D4	• alerting-Device	D4	• alerting-Device	D4	• alerting-Device	D4	
	• calling-Device	D1	• calling-Device	D1	• calling-Device	D1	• calling-Device	D1	
	• called-Device	D2	• called-Device	D2	• called-Device	D2	• called-Device	D2	
	• OrigNID-Conn	N1C1	• OrigNID-Conn	N1C1	• OrigNID-Conn	N1C1	• OrigNID-Conn	N1C1	
	• NWCalling-Device	D1	• NWCalling-Device	D1	• NWCalling-Device	D1	• NWCalling-Device	D1	
	• AssCalling-Device	N1	• AssCalling-Device	N1	• AssCalling-Device	N1	• AssCalling-Device	N1	
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting	• localConnect-ionInfo	alerting	• localConnect-ionInfo	alerting	
	• cause	multiAlert	• cause	multiAlert	• cause	multiAlert	• cause	multiAlert	
	• services-Permitted	ClearConn SendUI	• services-Permitted	Answer ClearCall SendUI	• services-Permitted	SendUI	• services-Permitted	SendUI	

### Remark:

Immediately after the AC has answered the call, it automatically seizes D2 again in a consultation-call (a new call-id will be created).

### 5.17.5.7 Intercept with parallel call to GAMQ

An external caller (directory-number D1) calls D2 via the network-interface N1 (trunk). D2 does not answer the call. After a timeout, the call is intercepted with parallel call to the GA (G3). The Attendant Console D4 answers the call.

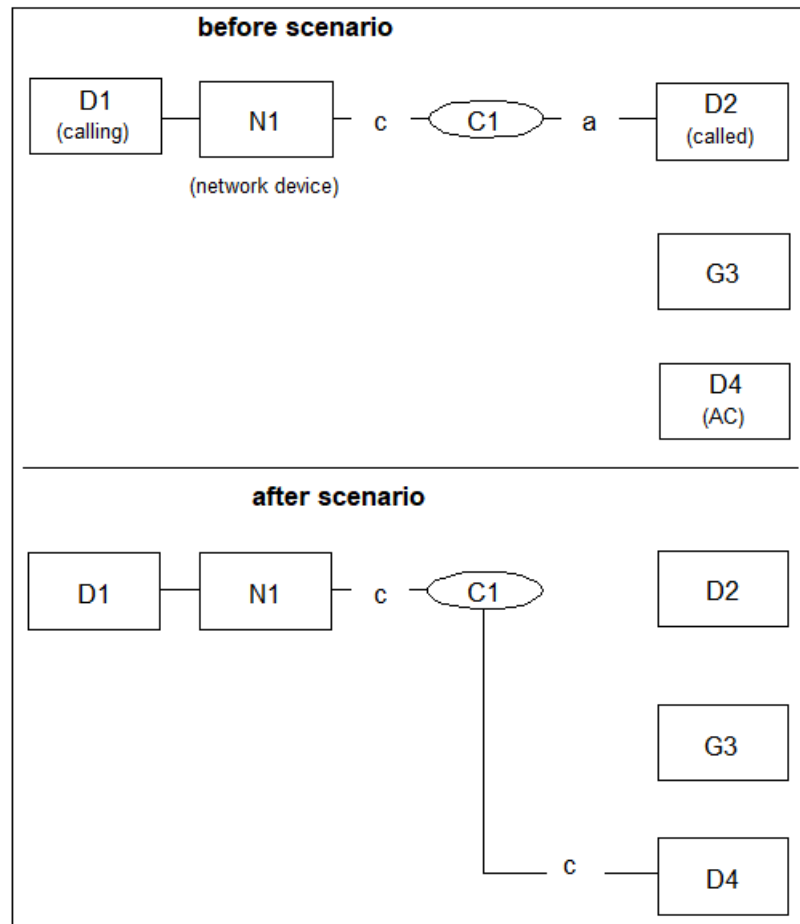


Figure 84: Intercept with parallel call to GAMQ

Table 305: Intercept with parallel call to GAMQ

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		
1) The call was previously ringing at D2. Intercept-timer has expired, the call is intercepted to GAMQ (with parallel call), the GAMQ is alerted	Delivered		Delivered		Delivered				
	• connection	G3C1	• connection	G3C1	• connection	G3C1			
	• alertingDevice	G3	• alertingDevice	G3	• alertingDevice	G3			
	• Calling-Device	D1	• Calling-Device	D1	• Calling-Device	D1			
	• calledDevice	D2	• calledDevice	D2	• calledDevice	D2			
	• OrigNIDConn	N1C1	• OrigNIDConn	N1C1	• OrigNIDConn	N1C1			
	• NWCalling-Device	D1	• NWCalling-Device	D1	• NWCalling-Device	D1			

## Call Scenarios

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
	• AssCalling-Device	N1	• AssCalling-Device	N1	• AssCalling-Device	N1			
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS			
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting	• localConnect-ionInfo	alerting			
	• cause	multiAlert	• cause	multiAlert	• cause	multiAlert			
	• services-Permitted	ClearCall SendUI	• services-Permitted	Answer ClearCall SendUI	• services-Permitted	SendUI			
1) Call is queued at GAMQ	Queued		Queued		Queued				
	• queued-Connection	G3C1	• queued-Connection	G3C1	• queued-Connection	G3C1			
	• queue	G3	• queue	G3	• queue	G3			
	• Calling-Device	D1	• Calling-Device	D1	• Calling-Device	D1			
	• calledDevice	D2	• calledDevice	D2	• calledDevice	D2			
	• NWCalling-Device	D1	• NWCalling-Device	D1	• NWCalling-Device	D1			
	• AssCalling-Device	N1	• AssCalling-Device	N1	• AssCalling-Device	N1			
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS			
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting	• localConnect-ionInfo	queued			
	• cause	normal	• cause	normal	• cause	normal			
	• services-Permitted	ClearConn SendUI	• services-Permitted	Answer ClearConn SendUI	• services-Permitted	SendUI			
1) AC picks - D2 leaves the call	<b>Connection Cleared</b>		<b>Connection Cleared</b>		<b>Connection Cleared</b>				
	• dropped-Connection	D2C1	• dropped-Connection	D2C1	• dropped-Connection	D2C1			
	• releasingDevice	D2	• releasingDevice	D2	• releasingDevice	D2			
	• localConnect-ionInfo	connected	• localConnect-ionInfo	null	• localConnect-ionInfo	queued			
	• cause	normalClr	• cause	normalClr	• cause	normalClr			

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		
	• services-Permitted	ClearConn SendUI	• services-Permitted	none	• services-Permitted	SendUI			
1) The call is diverted from GAMQ					<b>Diverted</b>				
					• connection	G3C1			
					• divertingDevice	G3			
					• newDestination	D4			
					• Calling-Device	D1			
					• calledDevice	D2			
					• lastRedirect-ionDev	NS			
					• localConnect-ionInfo	null			
					• cause	distributed			
					• services-Permitted	none			
1) The call arrives at the AC.	Delivered						Delivered		
	• connection	D4C1					• connection	D4C1	
	• alertingDevice	D4					• alertingDevice	D4	
	• Calling-Device	D1					• Calling-Device	D1	
	• calledDevice	D2					• calledDevice	D2	
	• OrigNIDConn	N1C1					• OrigNIDConn	N1C1	
	• NWCcalling-Device	D1					• NWCcalling-Device	D1	
	• AssCalling-Device	N1					• AssCalling-Device	N1	
	• lastRedirect-ionDev	G3					• lastRedirect-ionDev	G3	
	• localConnect-ionInfo	connected					• localConnect-ionInfo	alerting	
	• cause	distributed					• cause	distributed	
	• services-Permitted	ClearConn SendUI					• services-Permitted	SendUI	
1) The call is established	Established						Established		
	• establishedConn	D4C1					• establishedConn	D4C1	
	• answeringDevice	D4					• answeringDevice	D4	

## Call Scenarios

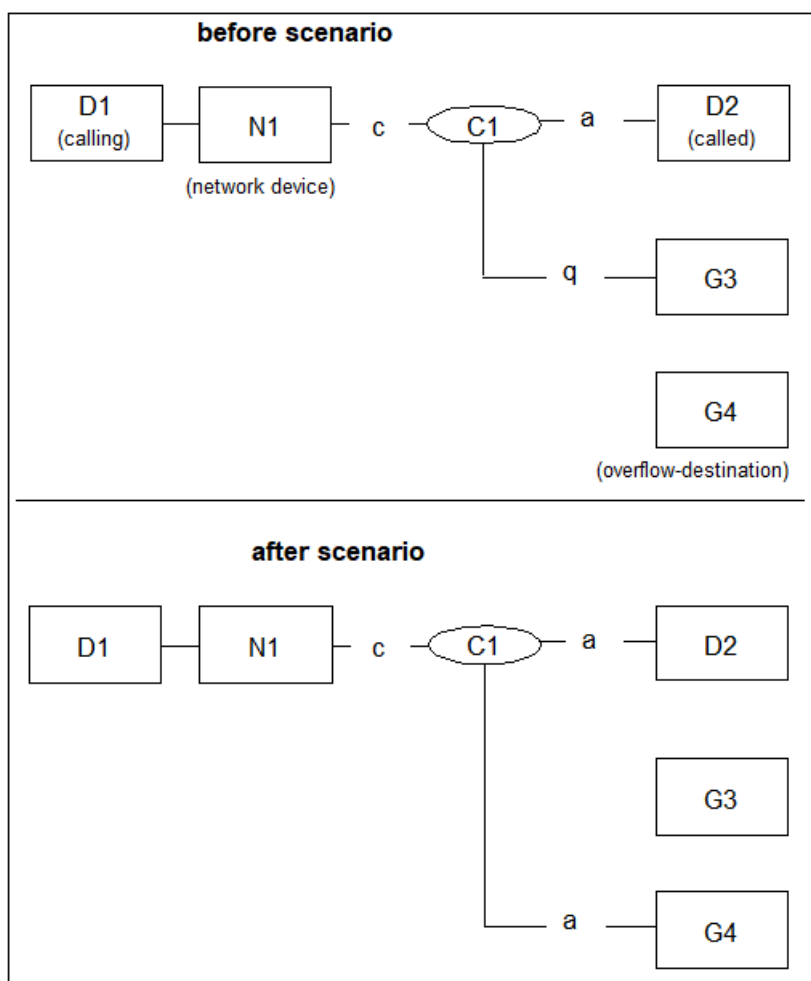
Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
	• Calling-Device	D1					• Calling-Device	D1	
	• calledDevice	D2					• calledDevice	D2	
	• OrigNIDConn	N1C1					• OrigNIDConn	N1C1	
	• NWCalling-Device	D1					• NWCalling-Device	D1	
	• AssCalling-Device	N1					• AssCalling-Device	N1	
	• lastRedirect-ionDev	G3					• lastRedirect-ionDev	G3	
	• localConnect-ionInfo	connected					• localConnect-ionInfo	connected	
	• cause	normal					• cause	normal	
	• services-Permitted	ClearConn SendUI					• services-Permitted	SendUI	

### Remark:

Immediately after the AC has answered the call, it automatically seizes D2 again in a consultation-call (a new call-id will be created).

### 5.17.5.8 Intercept with parallel call, call is routed to another GA after timeout

An external caller (directory-number D1) calls D2 via the network-interface N1 (trunk). D2 does not answer the call. After a timeout, the call is intercepted with parallel call to G3 (GA2Q) and queued there. The call is not answered by an AC within a certain amount of time; after a time-out, the call is overflowed to GA4 (GAMQ).



**Figure 85: Intercept with parallel call, call is routed to another GA after timeout**

## Call Scenarios

**Table 306: Intercept with parallel call to GA2Q**

Activity	Monitored Device N1		Monitored Device D2		Monitored Device G3 (GA2Q)		Monitored Device G4 (GAMQ)		Comments
1) The call was queued at GA2Q due to intercept with parallel call - an overflow-timer elapses, G3 withdraws from the call.	<b>Connection Cleared</b>		<b>Connection Cleared</b>		<b>Connection Cleared</b>				Please note: no Diverted-Event is sent for G3 because the call remains MultiAlert all the time (G3 leaves the call and D4 joins the call).  (Please refer to section 5.17.5.1, "General Rules concerning Multi-Alert-Situations for GA" for more information).
	• dropped-Connection	G3C1	• dropped-Connection	G3C1	• dropped-Connection	G3C1			
	• releasing-Device	G3	• releasing-Device	G3	• releasing-Device	G3			
	• localConnectionInfo	connected	• localConnectionInfo	alerting	• localConnectionInfo	null			
	• cause	normalClr	• cause	normalClr	• cause	normalClr			
	• services-Permitted	Clear-Call SendUI	• services-Permitted	ClearConn Answer SendUI	• services-Permitted	none			
1) Overflow-destination G4 is added to the call.	Delivered		Delivered				Delivered		
	• connection	G4C1	• connection	G4C1			• connection	G4C1	
	• alertingDevice	G4	• alertingDevice	G4			• alertingDevice	G4	
	• callingDevice	D1	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2			• calledDevice	D2	
	• OrigNIDConn	N1C1	• OrigNIDConn	N1C1			• OrigNIDConn	N1C1	
	• NWCcalling-Device	D1	• NWCcalling-Device	D1			• NWCcalling-Device	D1	
	• AssCalling-Device	N1	• AssCalling-Device	N1			• AssCalling-Device	N1	
	• lastRedirect-ionDev	D2	• lastRedirect-ionDev	NS			• lastRedirect-ionDev	NS	
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alerting			• localConnect-ionInfo	alerting	
	• cause	multiAlert	• cause	multiAlert			• cause	multiAlert	
	• services-Permitted	Clear-Call SendUI	• services-Permitted	ClearConn Answer SendUI			• services-Permitted	SendUI	

**Remark:**

None

### 5.17.5.9 Trunk-to-trunk supervision

Two NIDs (trunks) without disconnect supervision are connected to each other (Call-Id C2). When the supervision timer expires for N1, a recall to the GA is executed, an AC answers the call.

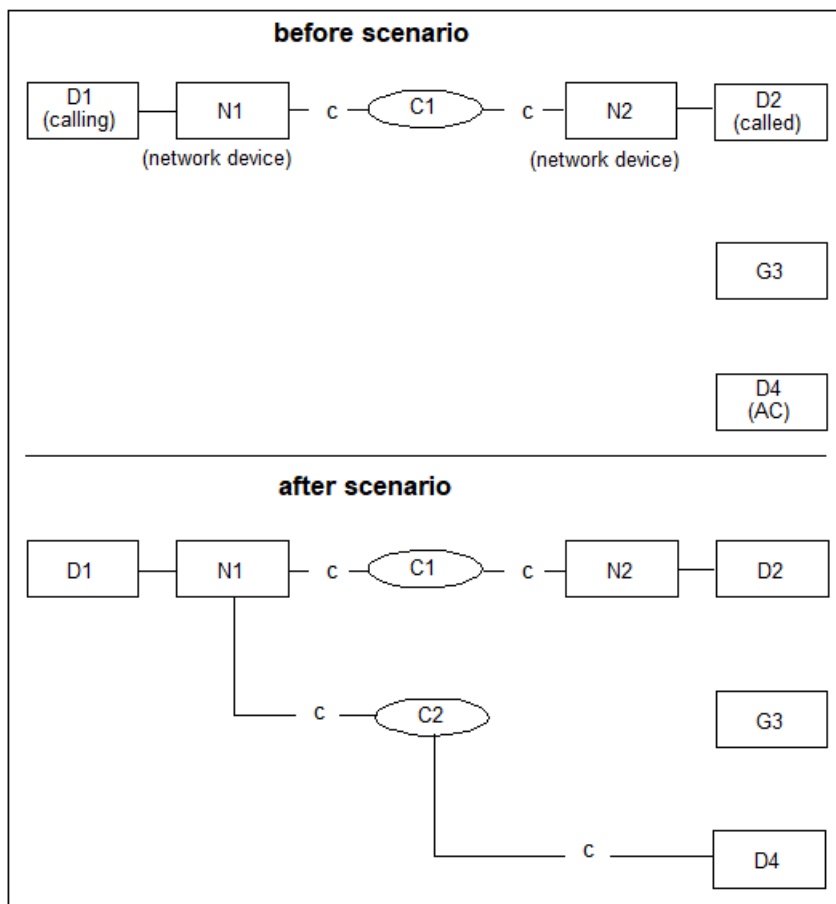


Figure 86: Trunk to trunk supervision

Table 307: Trunk to trunk supervision

Activity	Monitored Device N1 (trunk1)		Monitored Device N2 (trunk2)		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
1) Supervision timer expires for N1(trunk 1)	No event		No event		Delivered				No events are sent for the trunks - this is non-standard-behaviour. (as is the case for all override-scenarios).
					• connection	G3C2			
					• alertingDevice	G3			
					• callingDevice	D1			
					• calledDevice	D2			
					• OrigNIDConn	N1C2			
					• NWCcalling-Device	D1			
					• AssCalling-Device	N1			

## Call Scenarios

Activity	Monitored Device N1 (trunk1)		Monitored Device N2 (trunk2)		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
					• lastRedirect-ionDev	NS			
					• localConnect-ionInfo	alerting			
					• cause	enterDist			
					• services-Permitted	SendUI			
1) AC is rung	No event		No event		Delivered		Delivered		
					• connection	D4C2	• connection	D4C2	
					• alertingDevice	D4	• alertingDevice	D4	
					• callingDevice	D1	• callingDevice	D1	
					• calledDevice	D2	• calledDevice	D2	
					• OrigNIDConn	N1C2	• OrigNIDConn	N1C2	
					• NWCalling-Device	D1	• NWCalling-Device	D1	
					• AssCalling-Device	N1	• AssCalling-Device	N1	
					• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	
					• localConnect-ionInfo	alerting	• localConnect-ionInfo	alerting	
					• cause	multiAlert	• cause	multiAlert	
					• services-Permitted	SendUI	• services-Permitted	SendUI	
1) D4 answers call - GA withdraws from the call	No event		No event		Connection Cleared		Connection Cleared		
					• dropped-Connection	G3C2	• dropped-Connection	G3C2	
					• releasingDevice	G3	• releasingDevice	G3	
					• localConnect-ionInfo	null	• localConnect-ionInfo	alerting	
					• cause	multiAlert	• cause	multiAlert	
					• services-Permitted	none	• services-Permitted	SendUI	
1) D4 is connected to D1	No event		No event				Established		
							• establishedConn	D4C2	
							• answering-Device	D4	

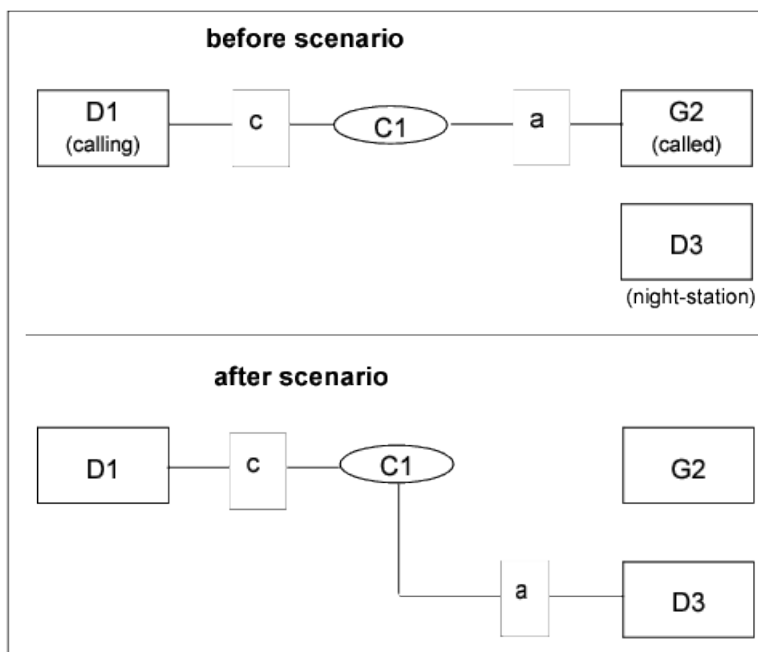
Activity	Monitored Device N1 (trunk1)		Monitored Device N2 (trunk2)		Monitored Device G3 (GA)		Monitored Device D4 (AC)		Comments
							• callingDevice	D1	
							• calledDevice	D2	
							• lastRedirect- ionDev	NS	
							• localConnect- ionInfo	connected	
							• cause	normal	
							• services- Permitted	SendUI	

**Remark:**

After having answered the call, the AC listens in on the call (like in an override situation) and decides whether to tear down the call or leave it.

### 5.17.5.10 Night-Service: General Night Station answers

D1 calls G2 (GA2Q, internal attendant access code: G2-int), the GA is in night-mode (General Night Service, GNS). The call is routed to the night-station D3.



**Figure 87: General night service answers**

Please refer to section [Section 5.17.4.3, "Internal call to Hunt Group, Hunt Advance"](#) for the event flow that leads to the "before"-state.

## Call Scenarios

**Table 308: Night-Service (GNS) - night station answers**

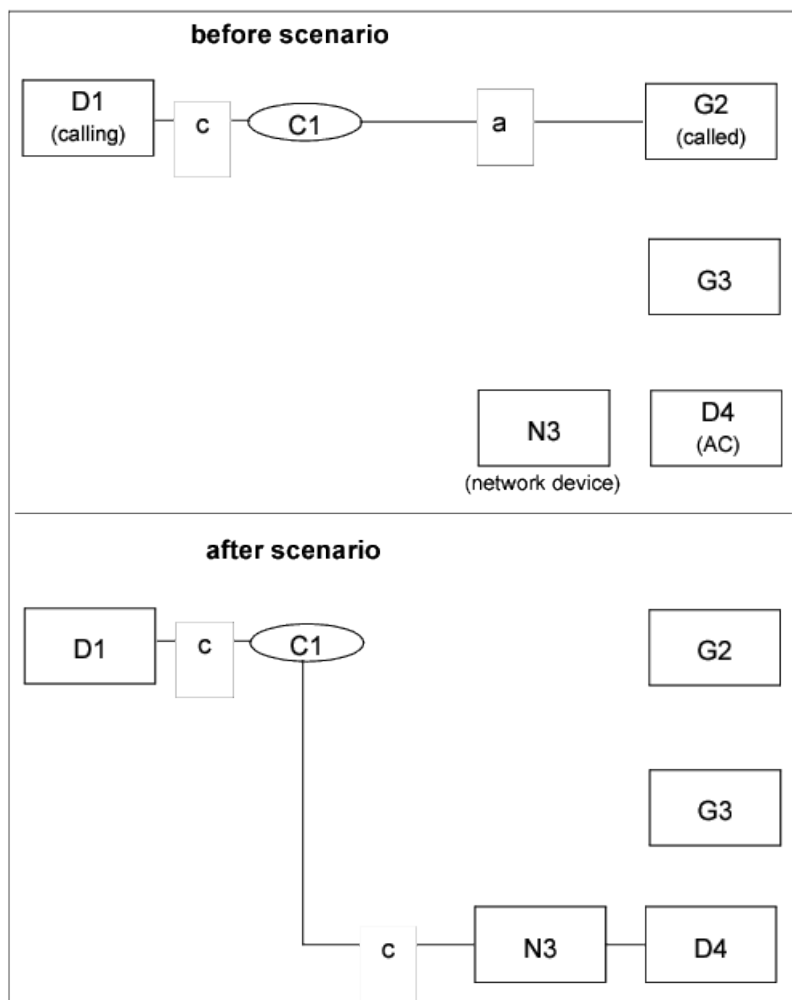
Activity	Monitored Device D1		Monitored Device G2 (GA2Q)		Monitored Device D3		Comments
1) The call was previously alerting at the GA. GA diverts the call to the night-station			Diverted				Please note:  A Diverted-Event is sent because the call leaves the GA as soon as the night-station starts ringing.
			• connection	G2C1			
			• divertingDevice	G2			
			• newDestination	D3			
			• callingDevice	D1			
			• calledDevice	G2-int			
			• lastRedirect-ionDev	NS			
			• localConn- ectionInfo	null			
			• cause	Cf-NA			
			• services- Permitted	none			
1) The night station is rung	Delivered				Delivered		
	• connection	D3C1			• connection	D3C1	
	• alertingDevice	D3			• alertingDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	G2-int			• calledDevice	G2-int	
	• lastRedirect-ionDev	G2			• lastRedirect-ionDev	G2	
	• localConn- ectionInfo	connected			• localConn- ectionInfo	alerting	
	• cause	Cf-NA			• cause	Cf-NA	
	• services- Permitted	ClearConn CallBack SendUI			• services-Permitted	Answer ClearConn Deflect SendUI	

**Remark:**

None

### 5.17.5.11 Night-Service: External Centralized Attendant Service group (CASEXT)

D1 calls G2 (GA2Q; internal attendant access code, diallable number: G2-int). G2 is in night-service (CASEXT). The call is forwarded to a GA2Q on another node (G3) and answered by an AC (D4) on the other node.



**Figure 88: External centralied attendant service group (CASEXT)**

**Remark:**

The call flow for the GA on the other node is like a basic external, incoming call to GA2Q and therefore not reproduced here.

**Table 309: Night-Service (ZVFEXT) - AC on other node answers**

Activity	Monitored Device D1		Monitored Device G2 (GA2Q)		Monitored Device N3 (trunk)		Comments
1) The call leaves the CSTA subdomain.	NW-Reached		NW-Reached		NW-Reached		
	• outboundConn	N3C1	• outboundConn	N3C1	• outboundConn	N3C1	
	• NWInterfaceUsed	N3	• NWInterfaceUsed	N3	• NWInterfaceUsed	N3	
	• callingDevice	D1	• callingDevice	D1	• callingDevice	D1	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device G2 (GA2Q)		Monitored Device N3 (trunk)		Comments
	• calledDevice	G2-int	• calledDevice	G2-int	• calledDevice	G2-int	
	• AssCalledDevice	N3	• AssCalledDevice	N3	• AssCalledDevice	N3	
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	
	• NW capabilities	ISDN Private	• NW capabilities	ISDN Private	• NW capabilities	ISDN Private	
	• localConn-ectionInfo	connected	• localConn-ectionInfo	alerting	• localConn-ectionInfo	connected	
	• cause	normal	• cause	normal	• cause	normal	
	• services-Permitted	ClearConn SendUI	• services-Permitted	SendUI	• services-Permitted	ClearConn CallBack SendUI	
1) The GA on the other node is alerted.	Delivered		Delivered		Delivered		Please note:  G3 (the GA) is alerting on the other node)
	• connection	N3C1	• connection	N3C1	• connection	N3C1	
	• alertingDevice	G3	• alertingDevice	G3	• alertingDevice	G3	
	• callingDevice	D1	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	G2-int	• calledDevice	G2-int	• calledDevice	G2-int	
	• AssCalledDevice	N3	• AssCalledDevice	N3	• AssCalledDevice	N3	
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS	
	• localConn-ectionInfo	connected	• localConn-ectionInfo	alerting	• localConn-ectionInfo	connected	
	• cause	multiAlert	• cause	multiAlert	• cause	multiAlert	
	• services-Permitted	ClearConn SendUI	• services-Permitted	SendUI	• services-Permitted	ClearConn CallBack SendUI	
1) AC on other node answers the call - the GA leaves the call.	<b>Connection Cleared</b>		<b>Connection Cleared</b>		<b>Connection Cleared</b>		
	• droppedConnection	G2C1	• droppedConnection	G2C1	• droppedConnection	G2C1	
	• releasingDevice	G2	• releasingDevice	G2	• releasingDevice	G2	
	• localConn-ectionInfo	connected	• localConn-ectionInfo	null	• localConn-ectionInfo	connected	
	• cause	multiAlert	• cause	multiAlert	• cause	multiAlert	

Activity	Monitored Device D1		Monitored Device G2 (GA2Q)		Monitored Device N3 (trunk)		Comments
	<ul style="list-style-type: none"> <li>services-Permitted</li> </ul>	ClearConn Consult, Hold SST GenDg GenTelTon SendUI	<ul style="list-style-type: none"> <li>services-Permitted</li> </ul>	none	<ul style="list-style-type: none"> <li>services-Permitted</li> </ul>	ClearConn SendUI	
1) The call is established between D1 and the AC on the other node (via NID N3).	Established				Established		answering Device = D4: As soon as the AC on the other node answers, the network-information changes. Therefore, D4 is shown as answering Device.
	<ul style="list-style-type: none"> <li>establishedConn</li> </ul>	N3C1			<ul style="list-style-type: none"> <li>establishedConn</li> </ul>	N3C1	
	<ul style="list-style-type: none"> <li>answeringDevice</li> </ul>	D4			<ul style="list-style-type: none"> <li>answeringDevice</li> </ul>	D4	
	<ul style="list-style-type: none"> <li>callingDevice</li> </ul>	D1			<ul style="list-style-type: none"> <li>callingDevice</li> </ul>	D1	
	<ul style="list-style-type: none"> <li>calledDevice</li> </ul>	G2-int			<ul style="list-style-type: none"> <li>calledDevice</li> </ul>	G2-int	
	<ul style="list-style-type: none"> <li>AssCalledDevice</li> </ul>	N3			<ul style="list-style-type: none"> <li>AssCalledDevice</li> </ul>	N3	
	<ul style="list-style-type: none"> <li>lastRedirect-ionDev</li> </ul>	NS			<ul style="list-style-type: none"> <li>lastRedirect-ionDev</li> </ul>	NS	
	<ul style="list-style-type: none"> <li>localConn-ectionInfo</li> </ul>	connected			<ul style="list-style-type: none"> <li>localConn-ectionInfo</li> </ul>	connected	
	<ul style="list-style-type: none"> <li>cause</li> </ul>	NW-signal			<ul style="list-style-type: none"> <li>cause</li> </ul>	NW-signal	
	<ul style="list-style-type: none"> <li>services-Permitted</li> </ul>	ClearConn Consult, Hold SST GenDg GenTelTon SendUI			<ul style="list-style-type: none"> <li>services-Permitted</li> </ul>	ClearConn SendUI	

**Remark:**

None

### 5.17.5.12 Intercept on a transit node

Node 2 is a transit-node: an external caller (directory-number D1 from node 1) calls a station on node 3(D2). D2 does not answer, the call is intercepted without parallel call to the GA2Q on node 2 G3.

## Call Scenarios

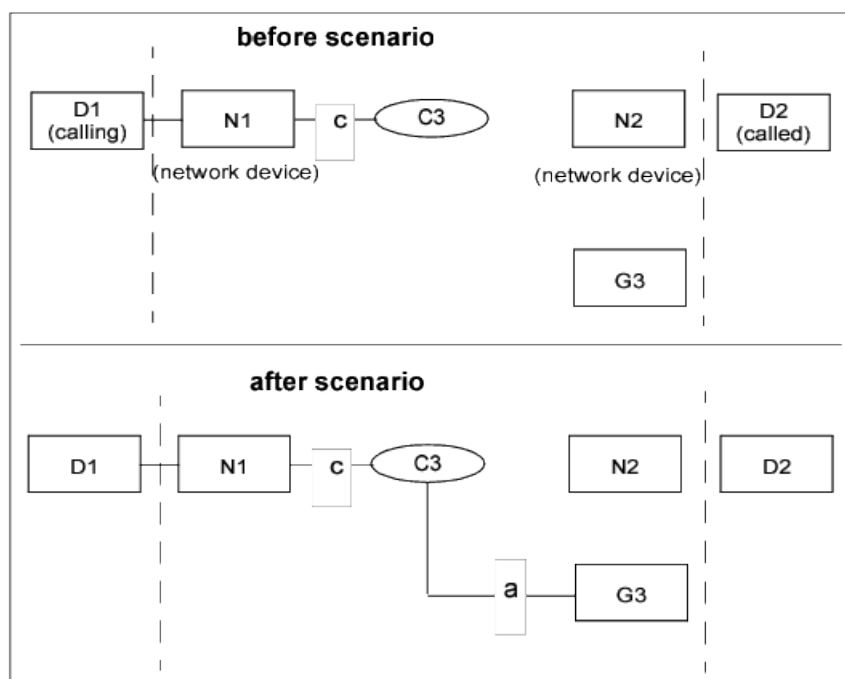


Figure 89: Intercept on a transit node

Table 310: Intercept on a transit node

Activity	Monitored Device N1		Monitored Device N2		Monitored Device G3 (GA)		Comments
1) D1 has finished dialing. The call leaves the CSTA subdomain	NW-Reached		NW-Reached				
	• outboundConn	N2C1	• outboundConn	N2C1			
	• NWInterfaceUsed	N2	• NWInterfaceUsed	N2			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	D2	• calledDevice	D2			
	• AssCallingDevice	N1	• AssCallingDevice	N1			
	• NWCcallingDevice	D1	• NWCcallingDevice	D1			
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS			
	• NW capabilities	ISDN Private	• NW capabilities	ISDN Private			
	• localConn- ectionInfo	connected	• localConn- ectionInfo	connected			
	• cause	normal	• cause	normal			
	• services-Permitted	ClearConn SendUI	• services- Permitted	ClearConn SendUI			

Activity	Monitored Device N1		Monitored Device N2		Monitored Device G3 (GA)		Comments
1) External destination D2 is reached	Established		Established				Established:
	• establishedConn	N2C1	• establishedConn	N2C1			Whenever HiPath
	• answeringDevice	D2' (1)	• answeringDevice	D2' (1)			4000 connects two
	• callingDevice	D1	• callingDevice	D1			trunks, this results
	• calledDevice	D2	• calledDevice	D2			in Established-
	• AssCallingDevice	N1	• AssCallingDevice	N1			Events for both
	• NWCallingDevice	D1	• NWCallingDevice	D1			trunks, even if
	• AssCalledDevice	N2	• AssCalledDevice	N2			the call was not
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS			answered yet
	• localConn- ectionInfo	connected	• localConn- ectionInfo	connected			(1) D2':
1) Intercept-timer ex- pires, the call is inter- cepted to GA			Diverted				The answeringDev
			• connection	N2C1			might differ from
			• divertingDevice	D2'			the calledDe-
			• newDestination	G3			vice, even if no
			• callingDevice	D1			forwarding or
			• calledDevice	D2			similar actions
			• lastRedirect-ionDev	NS			were performed on
			• localConn- ectionInfo	null			node 3. This is due
			• cause	callNotAnswer			to information ACL
1) The GA is alerted.			• services- Permitted	none			receives from the
	Delivered				Delivered		network.
	• connection	G3C1			• connection	G3C1	
	• alertingDevice	G3			• alertingDevice	G3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	

## Call Scenarios

Activity	Monitored Device N1		Monitored Device N2		Monitored Device G3 (GA)		Comments
	• OrigNIDConn	N1C1			• OrigNIDConn	N1C1	
	• NWCallingDevice	D1			• NWCallingDevice	D1	
	• AssCallingDevice	N1			• AssCallingDevice	N1	
	• lastRedirect-ionDev	D2			• lastRedirect-ionDev	D2	
	• localConn- ectionInfo	connected			• localConn- ectionInfo	alerting	
	• cause	enterDist			• cause	enterDist	
	• services-Permitted	ClearCall SendUI			• services-Permitted	SendUI	

**Remark:**

None

### 5.17.5.13 Call forwarded (CFNA) to Attendant, Multiple alerting, providing Diverted event to calling side is enabled

D1 calls D2 which has call forward no answer configured to GA (G3), call is answered by an AC (D4).

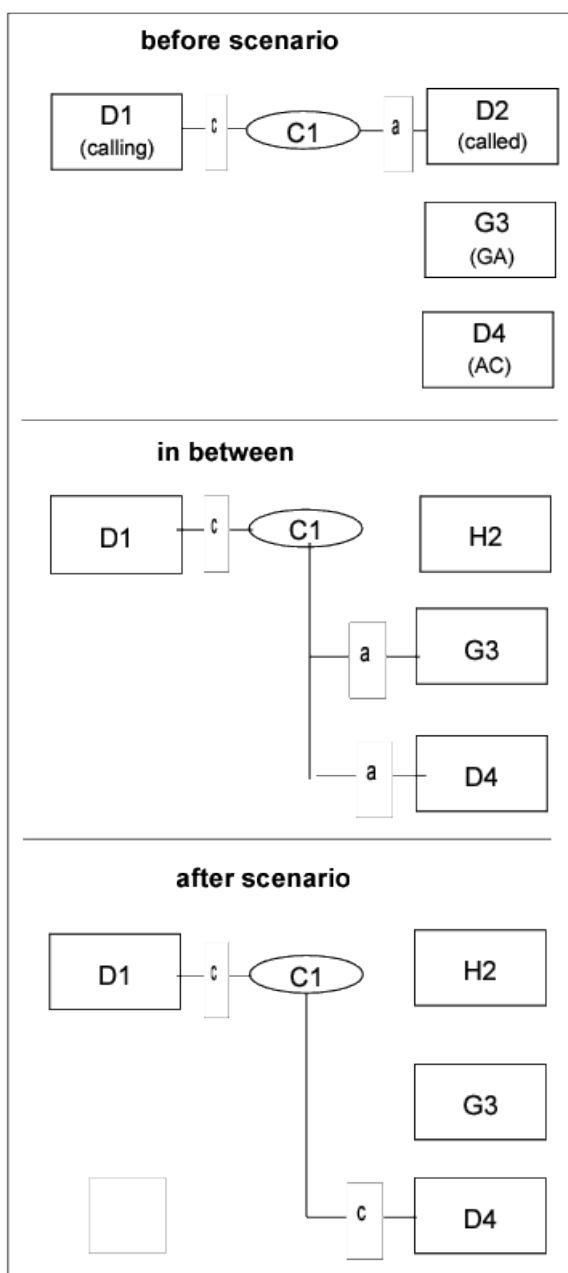


Figure 90: Call forward (CFNA) to attendant

Activity	Monitored Device D1		Monitored Device D2		Monitored Device G3 (General Attendant)		Monitored Device D4		Comments
1) CFNA timer expires	DivertedEvent		DivertedEvent						
	• connection	D2C1	• connection	D2C1					
	• divertingDevice	D2	• divertingDevice	D2					
	• newDest	G3	• newDest	G3					
	• callingDevice	D1	• callingDevice	D1					

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device G3 (General Attendant)		Monitored Device D4		Comments
	• calledDevice	D2	• calledDevice	D2					
	• lastRedirect-ionDev	NS	• lastRedirect-ionDev	NS					
	• localConn- ectionInfo	connected	• localConn- ectionInfo	null					
	• cause	forwardNoAns	• cause	forward- NoAns					
	• services- Permitted	none	• services- Permitted	none					
1) General attendant is in alerting state	DeliveredEvent				DeliveredEvent				Please note: No Defect is allowed for D3 because this is a MultiAlert-situation
	• connection	G3C1			• connection	G3C1			
	• alertingDevice	G3			• alertingDevice	G3			
	• callingDevice	D1			• callingDevice	D1			
	• calledDevice	D2			• calledDevice	D2			
	• lastRedirect-ionDev	D2			• lastRedirect-ionDev	D2			
	• localConn- ectionInfo	connected			• localConn- ectionInfo	alerting			
	• cause	enterDist			• cause	enteDist			
	• services- Permitted	ClearConn SendUI			• services- Permitted	SendUI			
1) Attendant console starts alerting	DeliveredEvent				DeliveredEvent		DeliveredEvent		Please note: No Defect is allowed for D3 because this is a MultiAlert-situation
	• connection	D4C1			• connection	D4C1	• connection	D4C1	
	• alertingDevice	D4			• alertingDevice	D4	• alertingDevice	D4	
	• callingDevice	D1			• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2			• calledDevice	D2	• calledDevice	D2	
	• lastRedirect-ionDev	D2			• lastRedirect-ionDev	D2	• lastRedirect-ionDev	D2	
	• localConn- ectionInfo	connected			• localConn- ectionInfo	alerting	• localConn- ectionInfo	alerting	
	• cause	multiAlert			• cause	multiAlert	• cause	multiAlert	
	• services- Permitted	ClearConn SendUI			• services- Permitted	SendUI	• services- Permitted	SendUI	

Activity	Monitored Device D1		Monitored Device D2		Monitored Device G3 (General Attendant		Monitored Device D4		Comments
1) D4 answers call - GA-device leaves the call	<b>Connection Cleared</b>				<b>Connection Cleared</b>		<b>Connection Cleared</b>		
	• dropped-Connection	G3C1			• dropped-Connection	H2C1	• dropped-Connection	G3C1	
	• releasingDevice	G3			• releasingDevice	H2	• releasingDevice	G3	
	• localConnectionInfo	connected			• localConnectionInfo	null	• localConnectionInfo	alerting	
	• cause	multiAlert			• cause	multiAlert	• cause	multiAlert	
1) D4 is connected to the original call	• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI			• services-Permitted	none	• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI	
	Established						Established		
	• established-Conn	D4C1					• established-Conn	D4C1	
	• answeringDevice	D4					• answeringDevice	D4	
	• callingDevice	D1					• callingDevice	D1	
	• calledDevice	D2					• calledDevice	D2	
	• lastRedirectionDev	NS					• lastRedirectionDev	NS	
	• localConnectionInfo	connected					• localConnectionInfo	connected	
	• cause	normal					• cause	normal	
	• services-Permitted	ClearConn Consult, Hold SST GenDg GenTelTon SendUI					• services-Permitted	ClearConn Consult Hold SST GenDg GenTelTon SendUI	

**Remark:**

None

## 5.18 Recall Scenarios

### 5.18.1 Softhold Recall

The consulting party clears its secondary call and it will be immediately recalled by the held party.

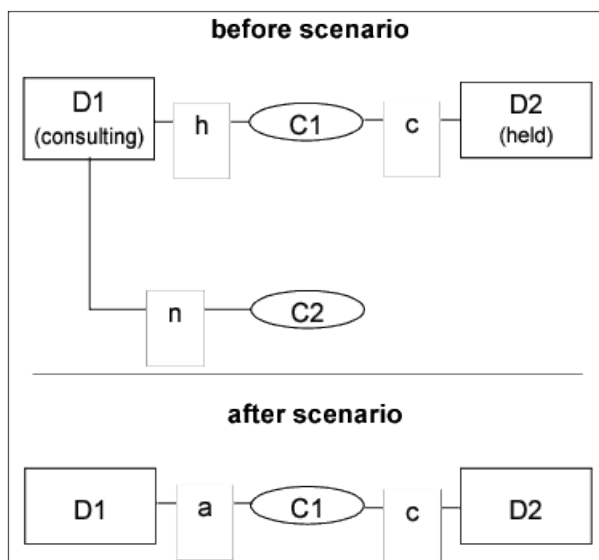


Figure 91: Softhold recall

Table 311: Softhold Recall

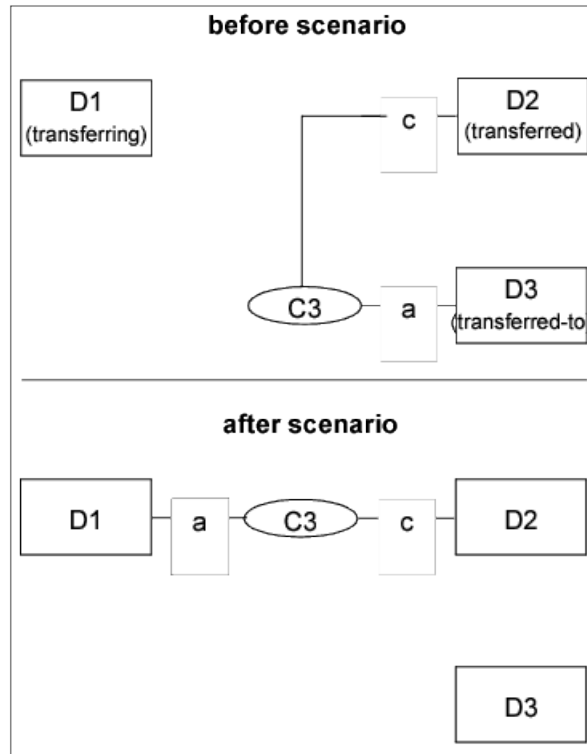
Activity	Monitored Device D1		Monitored Device D2		Comments
1) D1 will be recalled.	Delivered		Delivered		
	• deliveredConnection	D1C1	• deliveredConnection	D1C1	
	• alertingDevice	D1	• alertingDevice	D1	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	alerting	• localConnectionInfo	connected	
	• cause	recall	• cause	recall	
	• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	• servicesPermitted	CallBack, ClearConn, SendUserInfo	

**Remark:**

None

## 5.18.2 Transfer Recall

If the transferred-to party does not answer until a specified time interval, the transferring device will be recalled by the transferred party.



**Figure 92: Transfer recall**

See [Section 5.14.2, "Blind Transfer \(with local view in Transferred event\)"](#) for the event flow to get into the "before scenario" state.

**Table 312: Transfer Recall**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Since device D3 does not answer, device D1 will be recalled.					Diverted		The switching function sends the Diverted event only to the diverting-Device.
					• divertedConnection	D3C3	
					• divertingDevice	D3	
					• newDestinationDevice	D1	
					• calledDevice	D3	
					• lastRedirectionDevice	NS	
					• localConnectionInfo	null	

## Call Scenarios

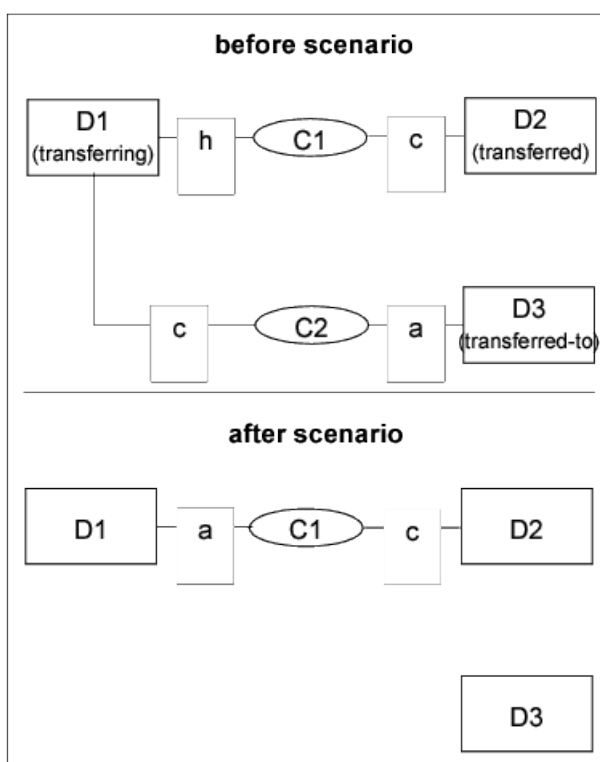
Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
					• cause	recall	
					• servicesPermitted	none	
1) D1 alerts.	Delivered		Delivered				
	• deliveredConnection	D1C3	• deliveredConnection	D1C3			
	• alertingDevice	D1	• alertingDevice	D1			
	• callingDevice	D2	• callingDevice	D2			
	• calledDevice	D3	• calledDevice	D3			
	• lastRedirection-Device	D3	• lastRedirection-Device	D3			
	• localConnectionInfo	alerting	• localConnectionInfo	connected			
	• cause	recall	• cause	recall			
	• servicesPermitted	Answer, ClearConn, Deflect, DialDgt, SendUserInfo	• servicesPermitted	CallBack, ClearConn, SendUserInfo			

**Remark:**

None

### 5.18.3 Transfer with restricted Connection

As the connection between the transferred and the transferred-to device is restricted, the transfer will result in a recall from the transferred device to the transferring device.



**Figure 93: Transfer with restricted connection**

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before scenario" state.

**Table 313: Transfer with Restricted Connection**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) D1 goes on hook to transfer, but it is not possible, so the call will be cleared.	Connection Cleared				Connection Cleared		
	• droppedConnection	D1C2			• droppedConnection	D1C2	
	• releasingDevice	D1			• releasingDevice	D1	
	• localConnectionInfo	null			• localConnectionInfo	alerting	
	• cause	callNotAnswered			• cause	callNotAnswered	
	• servicesPermitted	ClearConn			• servicesPermitted	none	
1) Device D3 is cleared from the call.					Connection Cleared		
					• droppedConnection	D3C2	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
					• releasingDevice	D3	
					• localConnectionInfo	null	
					• cause	normalClr	
					• servicesPermitted	none	
1) D2 will be recalled.	Delivered		Delivered				
	• deliveredConnection	D1C1	• deliveredConnection	D1C1			
	• alertingDevice	D1	• alertingDevice	D1			
	• callingDevice	D2	• callingDevice	D2			
	• calledDevice	D1	• calledDevice	D1			
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS			
	• localConnectionInfo	alerting	• localConnectionInfo	connected			
	• cause	recall	• cause	recall			
	• servicesPermitted	Answer, ClearConn, Deflect, DialDgt, SendUserInfo	• servicesPermitted	CallBack, ClearConn, SendUserInfo			

**Remark:**

None

### 5.18.4 Conference Recall

A participating party of the conference consults and afterwards clears its secondary call. It will be immediately recalled by the conference.

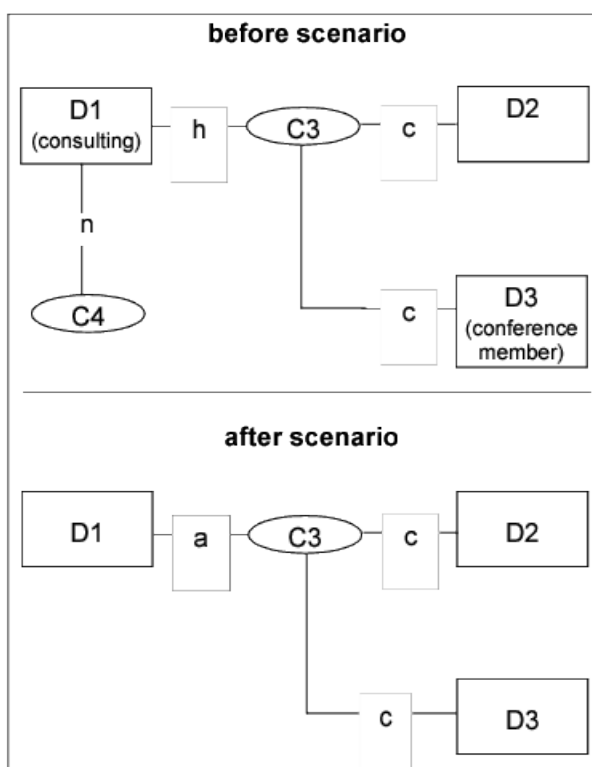


Figure 94: Conference recall

Table 314: Conference Recall

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Device D1 is recalled.	Delivered		None		None		Note that only the recalled party gets the Delivered event.
	• deliveredConnection	D1C3					
	• alertingDevice	D1					If D1 answers, the Established event will also be reported only for D1.
	• callingDevice	NK					
	• calledDevice	D2					
	• lastRedirection-Device	NS					Note, that the originally called device remains D2.
	• localConnection-Info	alerting					
	• cause	recall					
	• servicesPermitted	ClearConn, Answer, Deflect, DialDgt, SendUserInfo					

**Remark:**

None

5.18.5 Park Timer expires

After the park timer expires, the parkTo party will be notified of the parked party.

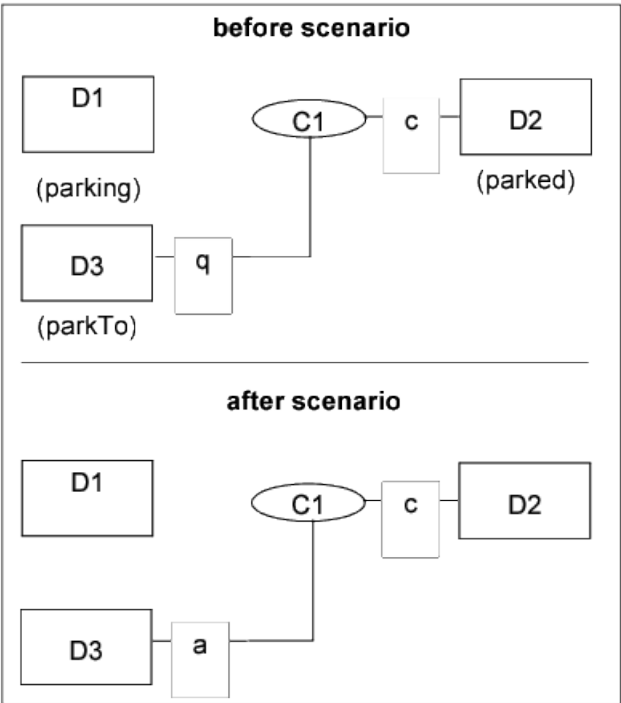


Figure 95: Park timer expiration

See [Section 5.11.4, "Manual directed park call"](#) for the event flow to get into the "before scenario" state.

Table 315: Parktimer expires

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Com-ments
1) Device D3 starts ringing after the park timer expired.	none		Delivered		Delivered		
			• connection	D3C1	• connection	D3C1	
			• alertingDevice	D3	• alertingDevice	D3	
			• callingDevice	D1	• callingDevice	D1	
			• calledDevice	D2	• calledDevice	D2	
			• lastRedirectionDe-vice	NS	• lastRedirectionDe-vice	NS	
			• localConnectionIn-fo	connected	• localConnectionInfo	alert	
			• cause	normal	• cause	normal	

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
			<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	CallBack, ClearConn, SendUserInfo	<ul style="list-style-type: none"> <li>servicesPermitted</li> </ul>	Answer, ClearConn, Deflect, SendUserInfo	

**Remark:**

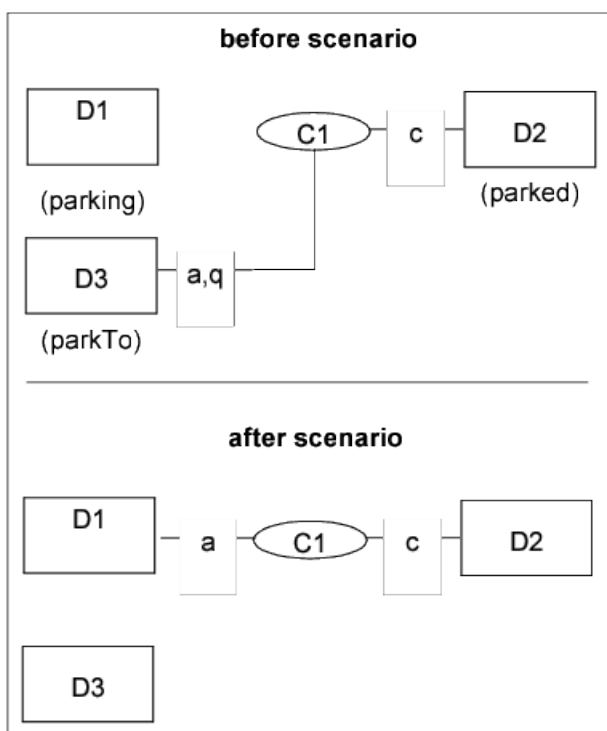
The switching function does not change the calling, called parameters in the event flow.

ECMA TR/82 reports changing calling, called devices in the related scenario.

See [Section 5.11.4, "Manual directed park call"](#).

### 5.18.6 Park Recall Timer Expires

After the park recall timer expires, the parked party recalls the parking party .



**Figure 96: Park recall timer expiration**

See [Section 5.18.5, "Park Timer expires"](#) or [Section 5.11.4, "Manual directed park call"](#) for the event flow to get into the "before scenario" state.

**Table 316: Park Recall Timer Expires**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) The call has been diverted from D3 due to the park recall timer expired.					Diverted		The switching function sends the Diverted event only to the diverting-Device.
					• connection	D3C1	
					• divertingDevice	D3	
					• newDestination	D1	
					• callingDevice	D1	
					• calledDevice	D2	
					• lastRedirectionDevice	NS	
					• localConnectionInfo	null	
					• cause	recall	
1) D1 alerts.	Delivered		Delivered				
	• connection	D1C1	• connection	D1C1			
	• alertingDevice	D1	• alertingDevice	D1			
	• callingDevice	D1	• callingDevice	D1			
	• calledDevice	D2	• calledDevice	D2			
	• lastRedirectionDevice	D3	• lastRedirectionDevice	D3			
	• localConnectionInfo	alert	• localConnectionInfo	connected			
	• cause	recall	• cause	recall			
	• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	• servicesPermitted	CallBack, ClearConn, SendUserInfo			

**Remark:**

None

## 5.18.7 Hard Hold Recall

This scenario describes the event flow of the Hard Hold Recall feature. An analog device puts an incoming external call on Hard Hold. After the timer expires the external party recalls the analog device.

D1 is anate ( analog telephone ).

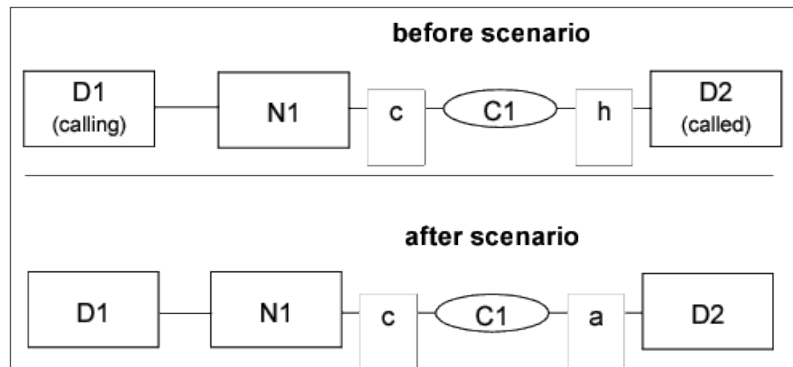


Figure 97: Kard hold recall

Table 317: Hard Hold Recall

Activity	Monitored Device N1		Monitored Device D2		Comments
1) Network device N1 recalls device D2.	Delivered		Delivered		
	• connection	D2C1	• connection	D2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• origNID connID	N1C1	• origNID connID	N1C1	
	• localConnectionInfo	connected	• localConnectionInfo	alerting	
	• cause	recall	• cause	recall	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	Answer, ClearConn, Deflect, DialDgt, SendUserInfo	

**Remark:**

None

## 5.19 OpenScape 4000 Specific Features

This section describes OpenScape 4000 features, that are either not described by ECMA 269 or they are not CSTA III standard compliant.

## 5.19.1 Route Optimization

This scenario describes an event flow, where the local route optimization feature is performed.

Device D1 and D2 are connected via network interface device N11. Device D2 consults to D3 and transfers D1 to D3. The switch optimizes the route to the local node and releases the involved network interface devices.

The Route Optimization feature is executed in the following cases:

- 1) After a netwide Transfer
- 2) After a netwide Pickup
- 3) After a netwide Park

---

**NOTICE:** In case of an ACD call Route Optimization is not possible.

---

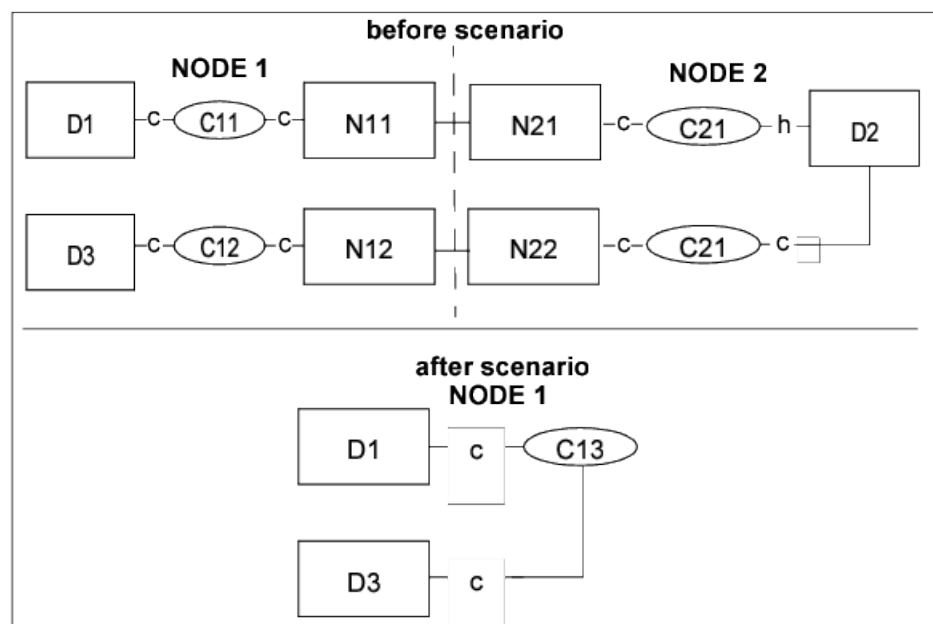


Figure 98: Route optimization

Table 318: Route Optimization

Activity	Monitored Device D1		Monitored Device N11		Monitored Device D2		Monitored Device N12		Comments
1) D2 hits transfer. The route will be optimized.	Transferred		Transferred		Transferred		Transferred		The event flow of the route optimization is not compliant with the ECMA 269 standard.
	• primaryOld-Call	D1C11	• primaryOld-Call	N11C11	• primaryOld-Call	D3C12	• primaryOld-Call	N12C12	
	• transferring	N11	• transferring	N11	• transferring	N12	• transferring	N12	
	• transferred-To	D3	• transferred-To	NK	• transferred-To	D3	• transferred-To	NK	
	• transferred-Conn 1. new / old 2. new	(D1C13) / (D1C11) (D3C13)	• transferred-Conn 1.old	N11C11	• transferred-Conn 1. new / old 2. new	(D3C13) / (D3C12) (D1C13)	• transferred-Conn 1. old	N12C12	
	• localConnection	connected	• localConnection	null	• localConnection	connected	• localConnection	null	
	• cause	SST	• cause	SST	• cause	SST	• cause	SST	
1) Device D1 and D3 are connected locally in a call.	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• servicesPermitted	none	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• servicesPermitted	none	
	Established				Established				
	• established-Conn	D3C13			• established-Conn	D3C13			
	• answeringDevice	D3			• answeringDevice	D3			
	• callingDevice	D1			• callingDevice	D1			
	• calledDevice	D3			• calledDevice	D3			
	• lastRedirection	NS			• lastRedirection	NS			
	• localConnectionInfo	connected			• localConnectionInfo	connected			
	• cause	NWSignal			• cause	NWSignal			

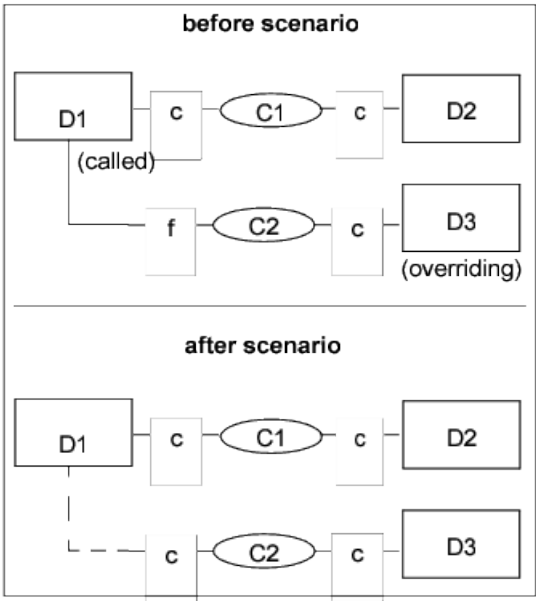
Activity	Monitored Device D1	Monitored Device N11	Monitored Device D2	Monitored Device N12	Comments
	<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo		<ul style="list-style-type: none"><li>servicesPermitted</li></ul>	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo

**Remark:**

Established event is sent only in those cases where there was no Established event provided already before the transfer.

5.19.2 Override

The D3 overriding party intrudes in an established call (C1) by pressing the override key. The called party will have 2 active calls.



**Figure 99: Override**

Table 319: Override

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) The busy connection is cleared immediately.	Connection Cleared		None		Connection Cleared		
	• droppedConnection	D1C2			• droppedConnection	D1C2	
	• releasingDevice	D2			• releasingDevice	D2	
	• localConnectionInfo	null			• localConnectionInfo	connected	
	• cause	normalClr			• cause	normalClr	
	• servicesPermitted	ClearConn			• servicesPermitted	none	
1) D3 hits override.					Established		The Established event is only reported on the overriding device monitor.
					• establishedConnection	D1C2	
					• answeringDevice	D1	
					• callingDevice	D3	
					• calledDevice	D1	
					• lastRedirectionDevice	NS	
					• localConnectionInfo	connected	
					• cause	override	
					• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	

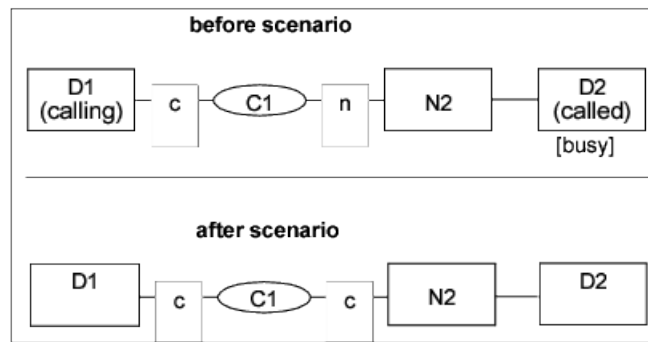
Remark:

None

### 5.19.2.1 Network-wide override

This scenario illustrates on override to a network party.

## Call Scenarios



**Figure 100: Network-wide override**

See [Section 5.7.1, "Manual call to a device outside the CSTA subdomain"](#) for the event flow to get into the "before scenario" state.

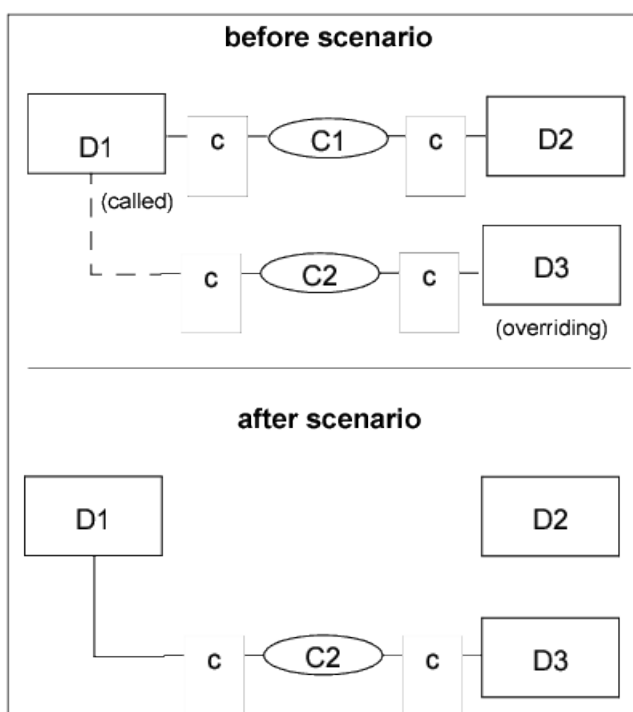
**Table 320: Netwide override**

Activity	Monitored Device D1		Monitored Device N2		Comments
1) The call reaches the network again.			Network Reached		The Network Reached event is not provided by the switching function on the monitor of device D1 .
			• outboundConnection	N2C1	
			• networkInterfaceUsed	N2	
			• callingDevice	D1	
			• calledDevice	D2	
			• lastRedirectionDevice	NS	
			• localConnectionInfo	connected	
			• cause	normal	
			• servicesPermitted	ClearConn, SendUserInfo	
1) The connection will be established immediately.	Established		Established		
	• established-Connection	N2C1	• establishedConnection	N2C1	
	• answeringDevice	D2	• answeringDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	nwSignal	• cause	nwSignal	

Activity	Monitored Device D1		Monitored Device N2		Comments
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTone, SendUserInfo	• servicesPermitted	ClearConn, SendUserInfo	
	• assocCalledDevice	N2	• assocCalledDevice	N2	

**Remark:**

Due to switching function limitation the Network Reached event cannot be provided for both participants.

**5.19.2.2 D2 goes onhook after the override****Figure 101: D2 goes onhook after override**

See [Section 5.19.2, "Override", on page 590](#) for the event flow to get into the "before scenario" state.

**Table 321: D2 goes on-hook after the override**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Device D2 goes onhook.	Connection Cleared		Connection Cleared		None		The Connection Cleared for D1C1 is missing.
	• droppedConnection	D2C1	• droppedConnection	D2C1			
	• releasingDevice	D2	• releasingDevice	D2			

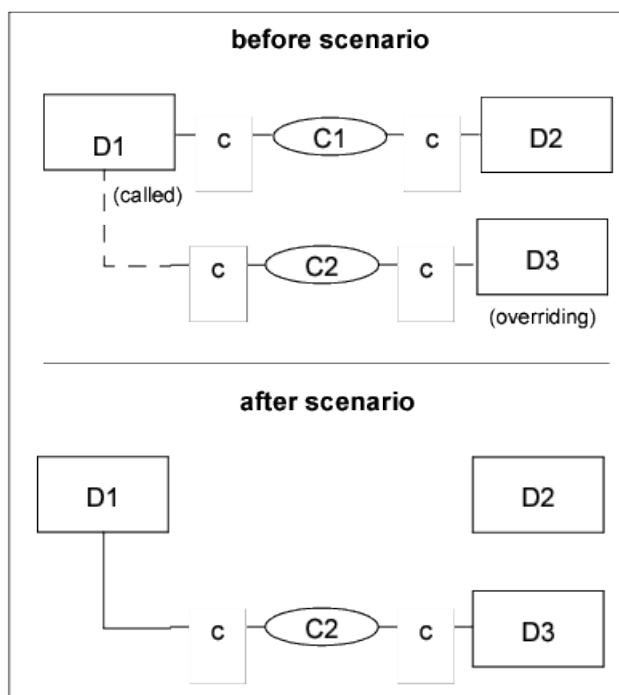
## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
	• localConnectionInfo	null	• localConnectionInfo	null			
	• cause	normalClr	• cause	normalClr			
	• servicesPermitted	ClearConn	• servicesPermitted	none			
1) The override tone is stopped,D1 is connected immediately to D3.	Established						The Established is reported only on the called device monitor.
	• established-Connection	D1C2					
	• answeringDevice	D1					
	• callingDevice	D3					
	• calledDevice	D1					
	• lastRedirectionDevice	NS					
	• localConnectionInfo	connected					
	• cause	normal					
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo					

### Remark:

None

### 5.19.2.3 D1 hits clear after the override



**Figure 102: D1 hits clear after override**

See [Section 5.19.2, "Override", on page 590](#) for the event flow to get into the "before scenario" state.

**Table 322: D1 hits clear after the override**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Device D1 hits the clear key.	Connection Cleared		Connection Cleared				
	• droppedConnection	D1C1	• droppedConnection	D1C1			
	• releasingDevice	D1	• releasingDevice	D1			
	• localConnectionInfo	null	• localConnectionInfo	connected			
	• cause	normalClr	• cause	normalClr			
	• servicesPermitted	ClearConn	• servicesPermitted	ClearConn			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) As a result of D1C1 clearing, remaining device D2 goes blocked.			Failed				
			• failedConnection	D2C1			
			• failingDevice	D2			
			• callingDevice	D2			
			• calledDevice	D1			
			• lastRedirectionDevice	NS			
			• localConnectionInfo	fail			
			• cause	blocked			
1) As a result of D1C1 clearing, D2C1 is also cleared.			• servicesPermitted	ClearConn			
			Connection Cleared				
			• droppedConnection	D2C1			
			• releasingDevice	D2			
			• localConnectionInfo	null			
1) The override tone is stopped, D1 is connected immediately to D3.			• cause	normalClr			
			• servicesPermitted	none			
	Established				Established		
	• established-Connection	D1C2			• established-Connection	D1C2	
	• answeringDevice	D1			• answeringDevice	D1	
	• callingDevice	D3			• callingDevice	D3	
	• calledDevice	D1			• calledDevice	D1	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	
	• localConnectionInfo	connected			• localConnectionInfo	connected	
	• cause	normal			• cause	normal	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo			• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	

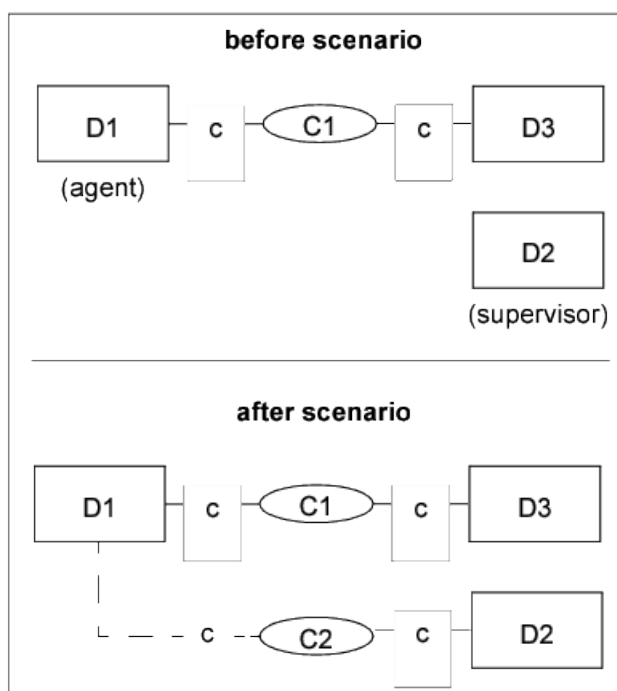
**Remark:**

None

**5.19.3 Silent Monitor**

This scenario describes an event flow where silent monitoring is used.

An ACD agent has a call. The supervisor hits the silent monitor key and dials the number of the agent. The supervisor will be connected in the call, and it can listen into the conversation of the agent.

**Figure 103: Silent momitor****Table 323: Silent Monitoring**

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
1) D2 supervisor presses the silent monitor key and dials the agent. (1234)	None	Service Initiated	None	
		• initiatedConnection D2C2		
		• initiatingDevice D2		
		• localConnectionInfo initiated		
		• cause consultation		
		• servicesPermitted ClearConn, DialDgt		
		Digits Dialed		
		• diallingConnection D2C2		
		• diallingDevice D2		

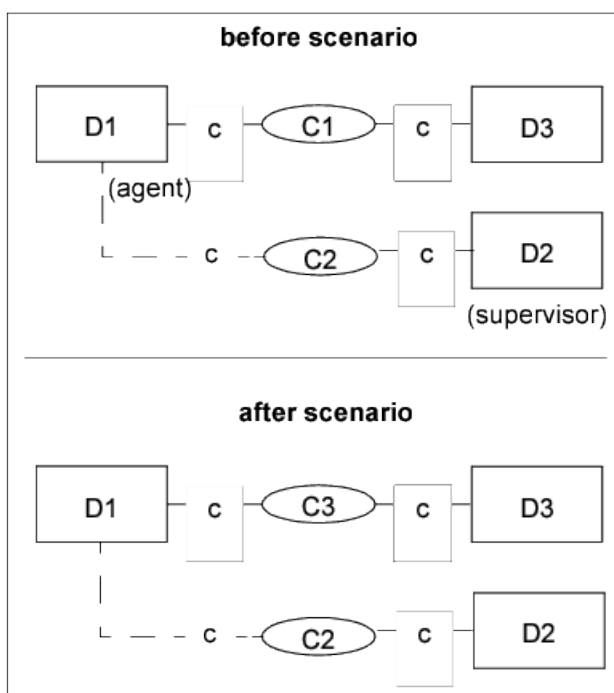
## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
			• diallingSequence	"1234"			
			• localConnectionInfo	initiated			
			• cause	normal			
			• servicesPermitted	none			
1) Device D2 is connecting to the agent.			Held				
			• heldConnection	D1C2			
			• holdingDevice	D1			
			• localConnectionInfo	connected			
			• cause	silentParticipation			
			• servicesPermitted	ClearConn			
1) Device D2 is connected to the agent.			Retrieved				
			• retrievedConnection	D1C2			
			• Retrieving	D1			
			• localConnectionInfo	connected			
			• cause	silentParticipation			
			• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo			

### Remark:

None

### 5.19.3.1 Agent goes onhook, afterwards original caller calls agent again



**Figure 104: Agent goes onhook, afterwards original caller calls agent again**

**Table 324: The agent goes onhook and afterwards the original caller calls the agent again.**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Device D1, the agent goes onhook.	Connection Cleared				Connection Cleared		
	• droppedConnection	D1C1			• droppedConnection	D1C1	
	• releasingDevice	D1			• releasingDevice	D1	
	• localConnectionInfo	null			• localConnectionInfo	connected	
	• cause	normalClr			• cause	normalClr	
1) The Held event shows, that the agent is idle.			Held				
			• heldConnection	D1C2			
			• holdingDevice	D1			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
			• localConnection-Info	connected			
			• cause	silentParticipation			
			• servicesPermitted	ClearConn			
1) As a result of D1C1 clearing, remaining device D3 goes blocked.					Failed		
					• failedConnection	D3C1	
					• failingDevice	D3	
					• callingDevice	D3	
					• calledDevice	RCG intDNIS	
					• lastRedirectionDevice	NS	
					• localConnectionInfo	fail	
					• cause	blocked	
					• servicesPermitted	ClearConn	
1) As a result of D1C1 clearing, D3C1 is also cleared.					Connection Cleared		
					• droppedConnection	D3C1	
					• releasingDevice	D3	
					• localConnectionInfo	null	
					• cause	normalClr	
					• servicesPermitted	none	
1) A new call arrives at the RCG which is distributed to the agent.							

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) D1 starts ringing.	Delivered				Delivered		
	• deliveredConnection	D1C3			• deliveredConnection	D1C3	
	• alertingDevice	D1			• alertingDevice	D1	
	• callingDevice	D3			• callingDevice	D3	
	• calledDevice	RCG intDNIS			• calledDevice	RCG intDNIS	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	
	• localConnectionInfo	alert			• localConnectionInfo	connected	
	• cause	distributed			• cause	distributed	
	• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo, DialDgt			• servicesPermitted	CallBack, ClearConn, SendUserInfo	
1) D1 answers the call.	Established				Established		
	• establishedConnection	D1C3			• establishedConnection	D1C3	
	• answeringDevice	D1			• answeringDevice	D1	
	• callingDevice	D3			• callingDevice	D3	
	• calledDevice	RCG intDNIS			• calledDevice	RCG intDNIS	
	• lastRedirectionDevice	NS			• lastRedirectionDevice	NS	
	• localConnectionInfo	connected			• localConnectionInfo	connected	
	• cause	normal			• cause	normal	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo			• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) The Retrieved event shows that the agent has a call.			Retrieved				
			• retrievedConnection	D1C2			
			• Retrieving	D1			
			• localConnectionInfo	connected			
			• cause	normal			
			• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo			

**Remark:**

None

### 5.19.4 Transfer before ALERT

This scenario describes an event flow where the transfer before alert feature is performed.

Device D1 calls D2 via network device N2. In the rare case of very slow network interfaces and appropriate OpenScape 4000 configuration, device D1 can consult to D3 before N2 connects to the call (C1) . Then D1 makes a screened transfer.

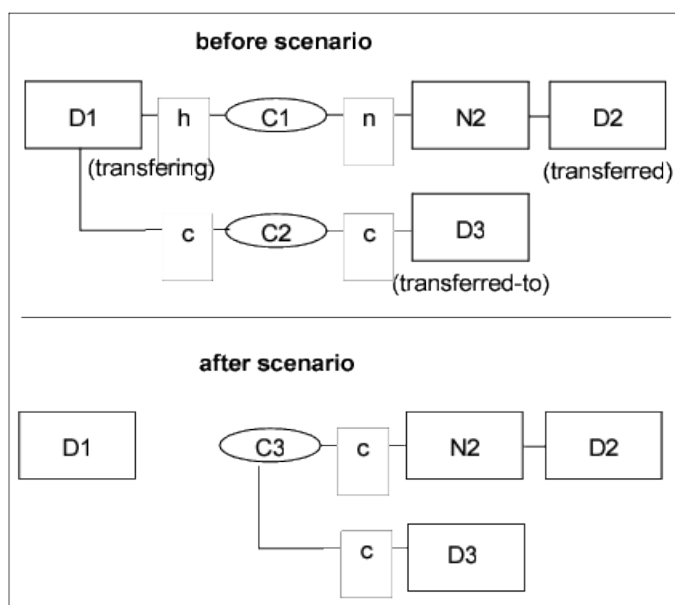


Figure 105: Transfer before ALERT

Table 325: Transfer before answer

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
1) D1 transfers by going on-hook.	Transferred				Transferred		The transferred-Connections parameter only includes the transferredTo connectionID.
	• primaryOldCall	D1C1			• primaryOldCall	D3C2	
	• secondaryOld-Call	D1C2					
	• transferringDevice	D1			• transferringDevice	D1	
	• transferred-ToDevice	D3			• transferredToDevice	D3	
	• transferredConnections 1. new / old	(D3C3) / (D3C2)			• transferredConnections 1. new / old	(D3C3) / (D3C2)	
	• localConnectionInfo	null			• localConnectionInfo	connected	
	• cause	Transfer			• cause	Transfer	
	• servicesPermitted	none			• servicesPermitted	ClearConn	
1) The call reaches the network.			Network Reached		Network Reached		
			• outboundConnection	N2C3	• outbound connection	N2C3	
			• NID device	N2	• NID device	N2	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
			• callingDe- vice	D3	• callingDevice	D3	
			• calledDevice	D2	• calledDevice	D2	
			• lastRedirec- tionDevice	NS	• lastRedirectionDe- vice	NS	
			• localConnec- tionInfo	connected	• localConnectionIn- fo	connected	
			• cause	normal	• cause	normal	
			• servicesPer- mitted	ClearConn	• servicesPermitted	ClearConn	
1) D2 starts ringing.			Delivered		Delivered		
			• deliveredCon- nection	N2C3	• deliveredConnec- tion	N2C3	
			• alertingDe- vice	D2	• alertingDevice	D2	
			• callingDe- vice	D3	• callingDevice	D3	
			• calledDevice	D2	• calledDevice	D2	
			• lastRedirec- tionDevice	NS	• lastRedirectionDe- vice	NS	
			• localConnec- tionInfo	connected	• localConnectionIn- fo	connected	
			• cause	NWSignal	• cause	NWSignal	
			• assocCalled	N2	• assocCalled	N2	
			• servicesPer- mitted	ClearConn, SendUserInfo	• servicesPermitted	CallBack, ClearConn, SendUserInfo	

### Remark:

None

## 5.19.5 Keyset Call (Multiline device call)

Logical device D1 (keyset) has two appearances associated with devices D1 and D2. D1 is the so called primary line, D2 is the secondary line of the keyset.

D3 dials keyset D1. D1/D1 answers the call. D1/D2 joins the call.

OpenScape CA 4000 V7.0 cannot handle multiple appearances, so the below event flow is non standard.

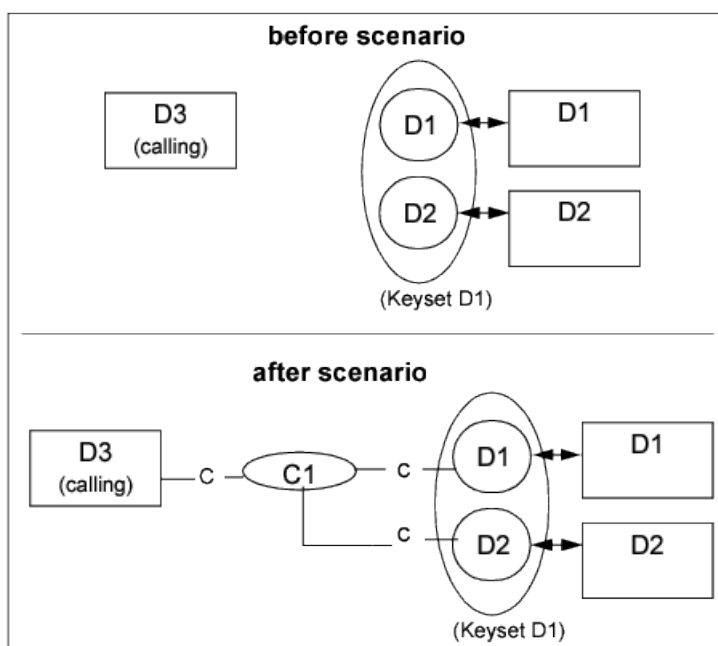


Figure 106: Keyset call (multi line device call)

Table 326: Keyset Call

Activity	Monitored Device D3		Monitored Device D1		Comments
1) D3 goes off-hook.	Service Initiated				
	• initiated-Connection	D3C1			
	• initiatingDevice	D3			
	• localConnect-ionInfo	initiated			
	• cause	normal			
	• services-Permitted	ClearConn, DialDgt			
1) D3 completes dialling D2.	Digits Dialed				D2's number is 1234
	• diallingConnection	D3C1			
	• diallingDevice	D3			
	• diallingSequence	"1234"			
	• localConnectionInfo	initiated			
	• cause	normal			
	• servicesPermitted	none			
	Originated				
	• originated-Connection	D3C1			
	• callingDevice	D3			

## Call Scenarios

Activity	Monitored Device D3		Monitored Device D1		Comments
	• calledDevice	D1			
	• lastRedirection-Dev	NS			
	• localConnect-ionInfo	connected			
	• cause	normal			
	• services-Permitted	ClearConn			
1) D1 is rung	Delivered		Delivered		
	• deliveredConnection	D1C1	• deliveredConnection	D1C1	
	• alertingDevice	D1	• alertingDevice	D1	
	• callingDevice	D3	• callingDevice	D3	
	• calledDevice	D1	• calledDevice	D1	
	• lastRedirection-Dev	NS	• lastRedirectionDev	NS	
	• localConnect-ionInfo	connected	• localConnect-ionInfo	alert	
	• cause	normal	• cause	normal	
	• services-Permitted	ClearConn, Callback, SendUI	• servicesPermitted	ClearConn, Answer, Deflect, SendUI	
1) D2 answers call by going offhook	Established		Established		
	• estbConnection	D1C1	• estbConnection	D1C1	
	• AnsweringDevice	D1	• AnsweringDevice	D1	
	• callingDevice	D3	• callingDevice	D3	
	• calledDevice	D1	• calledDevice	D1	
	• lastRedirection-Dev	NS	• lastRedirectionDev	NS	
	• localConnect-ionInfo	connected	• localConnect-ionInfo	connected	
	• cause	normal	• cause	normal	
	• servicesPermitted	ClearConn, Hold, Consultatiol, SST, GenDigits, GenTelTon, SendUI	• servicesPermitted	ClearConn, Hold, Consultation, SST, GenDigits, GenTelTon, SendUI	

Activity	Monitored Device D3		Monitored Device D1		Comments
1) D2 bridges in by pressing the key for line D1	Conferenced		Conferenced		Please note that Device D2 does not receive ANY event, because it does not use its own primary line, but bridges on to the primary line of D1.
	• primaryOldCall	D3C1	• primaryOldCall	D1C1	
	• secondaryOld-Call	NP	• secondaryOldCall	NP	
	• conferencing-Device	D1	• conferencingDevice	D1	
	• addedDevice	D1	• addedDevice	D1	
	• ConnectionList 1.new / old 2. new 3. new / old	(D3C2) / (D3C1) / (D1C2) (D1C2) / (D1C1)	• ConnectionList 1.new / old 2. new 3. new / old	(D3C2) / (D3C1) / (D1C2) (D1C2) / (D1C1)	
	• localConnect-ionInfo	connected	• localConnect-ionInfo	connected	
	• cause	keyOperation	• cause	keyOperation	
	• servicesPermitted	SendUI	• servicesPermitted	SendUI	

**Remark:**

A keyset is a multiline device. CSTA Phase III models it as independent shared bridge appearances. See ECMA 269 A.2.3. However this is not supported by OpenScape 4000 CSTA V1.

## 5.19.6 Making Calls in an Executive-Secretary team (CheSe feature)

### 5.19.6.1 General Remarks

In an Executive-Secretary team calls to the Executive are redirected to a Secretary. The configuration can consist of a maximum of 4 Executives and 2 Secretaries.

The Secretary can have a Representative. When the Secretary activates its Representative, then all Chese calls will be redirected to the Representative.

It is possible to deactivate the CheSe feature temporarily by either the Executive or the Secretary.

The Chese feature has higher priority in the following situations:

- Call Forward Immediate - if the secretary activates the Call Forward Immediate feature then the forwarding won't take effect, but a normal CheSe call flow happens instead.
- Call during the ParkTimer time - if a third party is parked to the Executive then it is not possible to call the parked-to party, but a basic CheSe call happens instead.

The Chese feature has lower priority in the following situations:

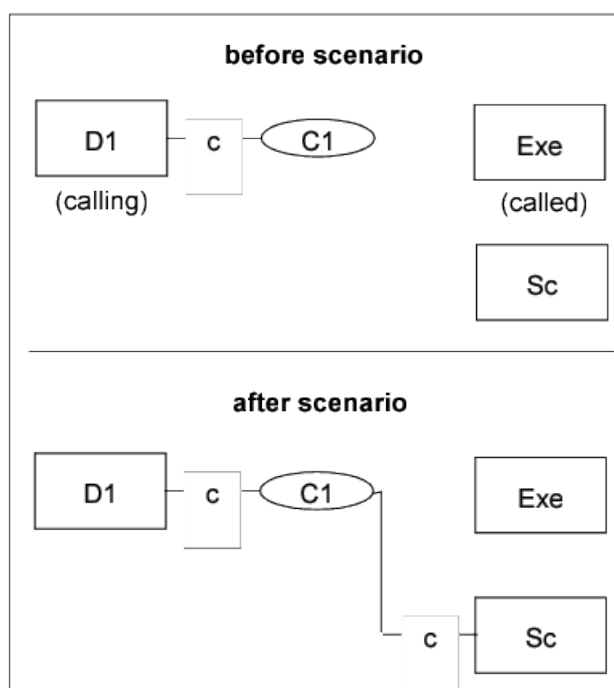
- Secretary and Executive are in the same HG - the normal HG call flow happens.
- Conference - the Executive can be reached from a conference immediately.

- Call Forward Immediate - if the Executive activates the Call Forward Immediate feature then a normal call forwarding happens.

There is a CSTA extension for this feature to make it possible to distinguish between the Secretary's private calls and CheSe calls. A privateData parameter (ExecutiveDeviceID) will be provided in the following events for the Secretary: Delivered, Established, Failed, Queued. This information element contains the extension number of the Executive involved in the CheSe call.

### 5.19.6.2 Successful basic call - call to Executive

This scenario illustrates the event flow of a successful basic CheSe call. A call comes to the Executive and it is redirected to the Secretary.



**Figure 107: Successful basic call - call to executive**

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before scenario" state.

**Table 327: Successful basic call - call to Executive**

Activity	Monitored Device D1		Monitored Device Exe		Monitored Device Sc		Comments
1) Secretary starts ringing.	Delivered		none		Delivered		The privateData indicates that it is a CheSe call.
	• connection	Sc C1			• connection	Sc C1	
	• alertingDevice	Sc			• alertingDevice	Sc	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	Exe			• calledDevice	Exe	

Activity	Monitored Device D1		Monitored Device Exe		Monitored Device Sc		Comments
	• lastRedirection-Device	NK			• lastRedirection-Device	NK	
	• localConnection-Info	connected			• localConnection-Info	alert	
	• cause	forwardImmediate			• cause	forwardImmediate	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo			• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	
					• ExecutiveDeviceID	Exe	
1) Secondary answers the call.	Established		none		Established		The privateData indicates that it is a Chase call.
	• establishedConnection	Sc C1			• establishedConnection	Sc C1	
	• answeringDevice	Sc			• answeringDevice	Sc	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	Exe			• calledDevice	Exe	
	• lastRedirection-Device	NS			• lastRedirection-Device	NS	
	• localConnection-Info	connected			• localConnection-Info	connected	
	• cause	normal			• cause	normal	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo			• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	
					• ExecutiveDeviceID	Exe	

**Remark:**

None

### 5.19.6.3 Successful basic call - Representative is activated

In case of a "Successful basic call - Representative is activated" similar event flow will be generated to the "Successful basic call - call to Executive" case.

See [Section 5.19.6.2, "Successful basic call - call to Executive"](#).

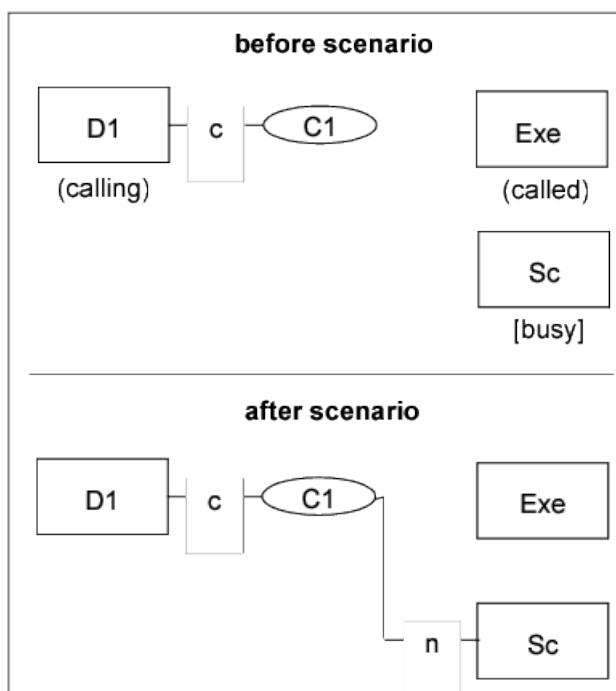
The only difference is that device Sc will be device Rep in this case.

### Remark:

The Secretary has activated a preconfigured Representative. See [Section 5.9.1, "Call Forward No Answer"](#).

### 5.19.6.4 Unsuccessful basic call - Secretary is busy

This scenario illustrates the event flow of an unsuccessful basic CheSe call. A call comes to the Executive and it is redirected to the Secretary who has another call also for the same Executive.



**Figure 108: Unsuccessful basic call - secretary is busy**

See [Section 7.4.1, "Manually dialled call"](#) for the event flow to get into the "before scenario" state.

Activity	Monitored Device D1		Monitored Device Exe		Monitored Device Sc		Comments
1) Secretary is busy. The call can not be completed. D1 hears busy tone.	Failed		none		Failed		The private-Data indicates that it is a Chese call.
	• failedConnection	Sc C1			• failedConnection	Sc C1	
	• failingDevice	Sc			• failingDevice	Sc	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	Exe			• calledDevice	Exe	
	• lastRedirection-Device	NS			• lastRedirection-Device	NS	

Activity	Monitored Device D1		Monitored Device Exe		Monitored Device Sc		Comments
	• localConnection-Info	connected			• localConnection-Info	fail	
	• cause	busy			• cause	busy	
	• servicesPermitted	ClearConn			• servicesPermitted	ClearConn	
					• ExecutiveDeviceID	Exe	
1) The busy connection is cleared immediately.	Connection Cleared		none		Connection Cleared		
	• droppedConnection	Sc C1			• droppedConnection	Sc C1	
	• releasingDevice	Sc			• releasingDevice	Sc	
	• localConnection-Info	connected			• localConnection-Info	null	
	• cause	normalClr			• cause	normalClr	
	• servicesPermitted	ClearConn			• servicesPermitted	none	

**Remark:**

None

**5.19.6.5 Unsuccessful basic call - Executive is busy**

In case of a "Unsuccessful basic call - Executive is busy" similar event flow will be generated to the "Unsuccessful basic call - Secretary is busy" case.

[Section 5.19.6.4, "Unsuccessful basic call - Secretary is busy".](#)

The only difference is that device Exe is busy in this case.

**Remark:**

Although in this case the Executive is busy, the event flow will be the same.

**5.19.6.6 Camp on Executive - Secretary is busy**

This scenario illustrates the event flow of an unsuccessful basic CheSe call followed by a manual camp on. A call comes to the Executive and it is redirected to the Secretary who has another call also for the same Executive. After receiving the busy tone, the calling device hits the camp on key.

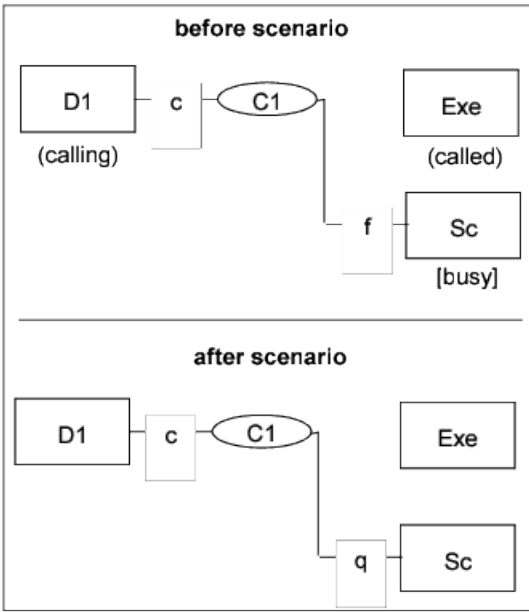


Figure 109: Camp on executive - secretary is busy

See [Section 5.19.6.4, "Unsuccessful basic call - Secretary is busy"](#) for the event flow to get into the "before scenario" state.

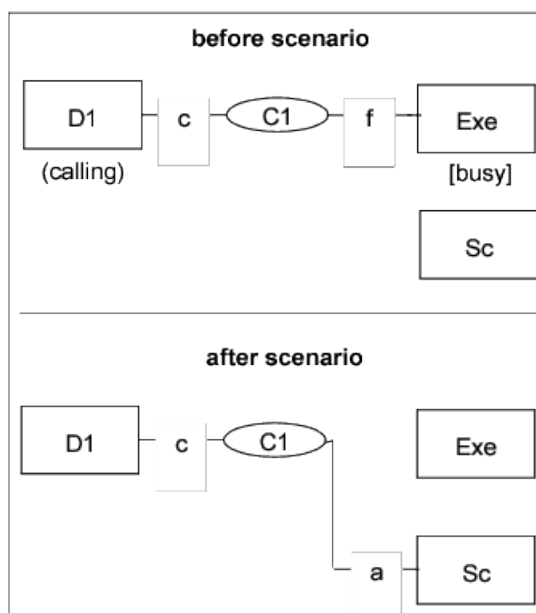
Activity	Monitored Device D1		Monitored Device Exe		Monitored Device Sc		Com-ments
1) D1 presses the camp on key.	Queued		none		Queued		The privateData indicates that it is a Chese call.
	• queuedConne-ction	Sc C1			• queuedConne-ction	Sc C1	
	• queue	Sc			• queue	Sc	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	Exe			• calledDevice	Exe	
	• lastRedirectionDe-vice	NS			• lastRedirection-Device	NS	
	• localConnectionIn-fo	connected			• localConnection-Info	queued	
	• cause	campOn			• cause	campOn	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo			• servicesPermit-tered	SendUser-Info	
					• ExecutiveDevi-celD	Exe	

Remark:

After the Queued event no privateData is present in the events of device Sc due to the limitation of the switch.

### 5.19.6.7 Camp on Executive - Executive is busy

This scenario illustrates the event flow of an unsuccessful basic CheSe call followed by a manual camp on. A call comes to the Executive who has another call. After receiving the busy tone, the calling device hits the camp on key. The call is redirected to the Secretary.



**Figure 110: Camp on executive - executive is busy**

See [Section 5.19.6.5, "Unsuccessful basic call - Executive is busy"](#) for the event flow to get into the "before scenario" state.

Activity	Monitored Device D1		Monitored Device Exe		Monitored Device Sc		Comments
1) Secretary starts ringing.	Delivered		none		Delivered		The privateData indicates that it is a Chese call.
	• connection	Sc C1			• connection	Sc C1	
	• alertingDevice	Sc			• alertingDevice	Sc	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	Exe			• calledDevice	Exe	
	• lastRedirection-Device	NK			• lastRedirection-Device	NK	
	• localConnectionInfo	connected			• localConnectionInfo	alert	
	• cause	forwardImmediate			• cause	forwardImmediate	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo			• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device Exe		Monitored Device Sc		Comments
					<ul style="list-style-type: none"> <li>ExecutiveDevi- celD</li> </ul>	Exe	

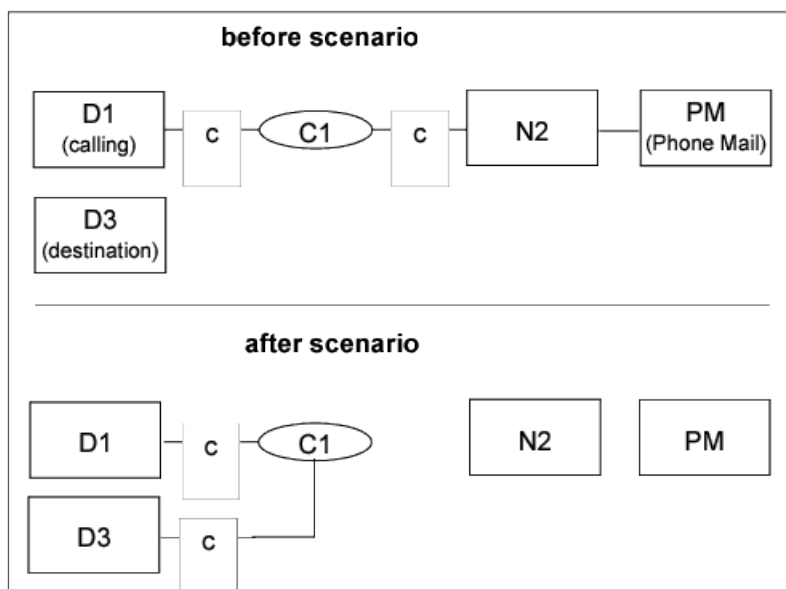
**Remark:**

None

## 5.19.7 Single Step Call Transfer with Await Connect

### 5.19.7.1 Successful basic SSCT call with Await Connect

This scenario describes a call flow when Phone Mail transfers with the Await Connect feature.



**Figure 111: Basis successful SSCT call with await connect**

See [Section 5.17.1.3, "Internal ACD call completed to Phone Mail agent"](#) for the event flow to get into the "before scenario" state.

**Table 328: Single Step Call Transfer (Await connect)**

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
1) D3 is prompted to go offhook.					Service Initiated		The SSCT private event cause shows, that this feature has been used.
					initiatedConnection	D3C2	
					initiatingDevice	D3	
					localConnectionInfo	initiated	

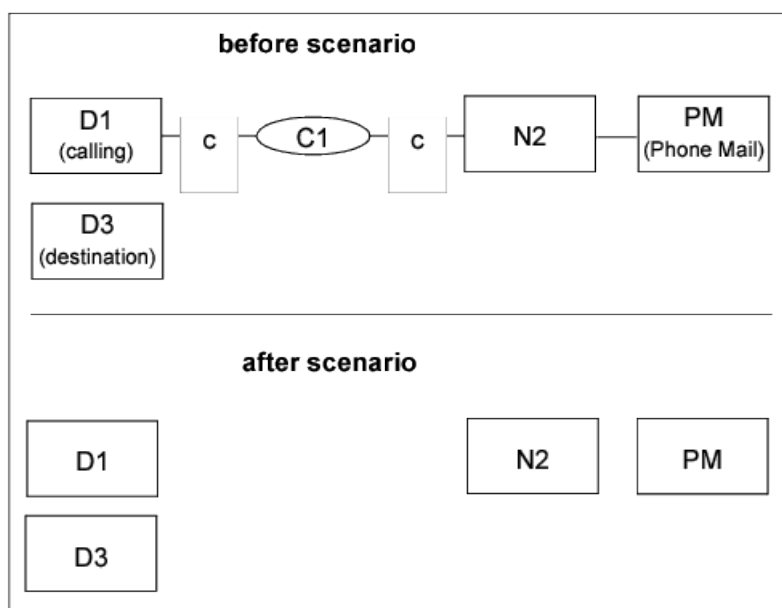
Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
					• cause	transfer	
					• servicesPermitted	Answer, ClearConn, Deflect, SendUserInfo	
					• privateEventCause	SSCT	
1) Device D3 goes off-hook.					Connection Cleared		The SSCT private event cause shows, that the Connection Cleared does not really mean, that D3 is idle. The next event will be an Established on this monitor.
					• droppedConnection	D3C2	
					• releasingDevice	D3	
					• localConnectionInfo	null	
					• cause	normalClr	
					• servicesPermitted	none	
					• privateEventCause	SSCT	
1) Phone Mail transfers.	Transferred		Transferred				OpenScape 4000 models a SSCT as a Single Step Transfer, that is why this cause is sent with the Transferred event.
	• primaryOld-Call	D1C1	• primaryOld-Call	N2C1			
	• transferringDevice	PM	• transferringDevice	PM			
	• transferred-ToDevice	D3	• transferred-ToDevice	D3			
	• transferred-Connections 1.new 2.new	(D1C1) (D3C1)	• transferred-Connections 1.new 2.new	(D1C1) (D3C1)			
	• localConnectionInfo	connected	• localConnectionInfo	null			
	• cause	SST	• cause	SST			
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTel-Tones, SendUserInfo	• servicesPermitted	none			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
1) The connection is immediately established.	Established				Established		OpenScape 4000 models a SSCT as a Single Step Transfer, that is why this cause is sent with the Established event.
	• established-Connection	D3C1			• establishedConnection	D3C1	
	• answering-Device	D3			• answeringDevice	D3	
	• callingDevice	D1			• callingDevice	D1	
	• calledDevice	D3			• calledDevice	D3	
	• lastRedirectionDevice	PM			• lastRedirectionDevice	PM	
	• localConnectionInfo	connected			• localConnectionInfo	connected	
	• cause	SST			• cause	SST	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTel-Tone, SendUserInfo			• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTel-Tone, SendUserInfo	

### 5.19.7.2 Unsuccessful basic SSCT call with Await Connect

This scenario describes a call flow when Phone Mail tries to transfers with the Await connect feature, but the calling party hangs up.



**Figure 112: Unsuccessful basic SSCT call with await connect**

See [Section 5.17.1.3, "Internal ACD call completed to Phone Mail agent"](#) for the event flow to get into the "before scenario" state.

**Table 329: Unsuccessful Single Step Call Transfer (Await connect)**

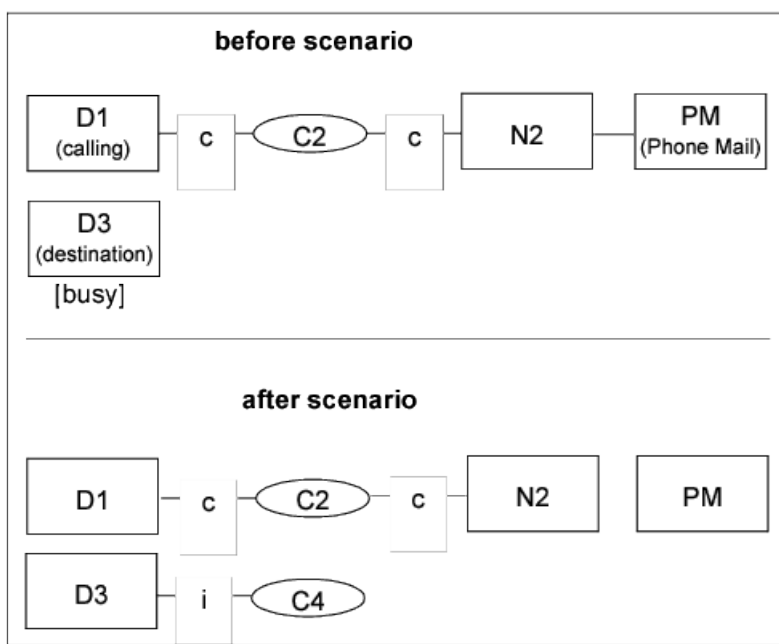
Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
1) D3 is prompted to go offhook.					Service Initiated		
					• initiatedConnection	D3C2	
					• initiatingDevice	D3	
					• localConnectionInfo	initiated	
					• cause	transfer	
					• servicesPermitted	Answer, Clear-Conn, Deflect, SendUserInfo	
1) Device D1 goes onhook. This results in a D3C2 clearing.					• privateEventCause	SST	
					• Connection Cleared		
					• droppedConnection	D3C2	
					• releasingDevice	D3	
					• localConnectionInfo	null	
					• cause	normalClr	
					• servicesPermitted	none	

## Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
					• privateEventCause	notPresent	
1) Device D1 goes onhook.	Connection Cleared		Connection Cleared				
	• droppedCon- nection	D1C1	• droppedCon- nection	D1C1			
	• releasingDe- vice	D1	• releasingDe- vice	D1			
	• localConnec- tionInfo	null	• localConnec- tionInfo	connected			
	• cause	normalClr	• cause	normalClr			
	• servicesPer- mitted	none	• servicesPermit- ted	ClearConn			
1) The re- main- ing con- nec- tion is cleared.			Connection Cleared				
			• droppedCon- nection	N2C1			
			• releasingDe- vice	N2			
			• localConnec- tionInfo	null			
			• cause	normalClr			
			• servicesPermit- ted	none			

### 5.19.7.3 SSCT call with Await Connect, Camp On

This scenario describes a call flow when Phone Mail transfers with the Await connect feature. Destination is non-idle but Camp On is possible.



**Figure 113: SSCT call with await connect, camp on**

See [Section 5.17.1.3, "Internal ACD call completed to Phone Mail agent"](#) for the event flow to get into the "before scenario" state.

**Table 330: Single Step Call Transfer, Camp On**

Activity	Monitored Device D1	Monitored Device N2	Monitored Device D3	Comments
1) The SSCT call camps onto Device D3.			<b>Queued</b>	
			• queuedConnection D3C3	
			• queue D3	
			• callingDevice D1	
			• calledDevice D3	
			• lastRedirectionDevice NS	
			• localConnectionInfo queued	
			• cause campon	
1) Device D3 clears itself from its previous call.			• servicesPermitted SendUserInfo	
			• privateEventCause SST	
			Connection Cleared	C1 was the active call of D3.
			• droppedConnection D3C1	
			• releasingDevice D3	
			• localConnectionInfo null	
			• cause normalClr	

Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Comments
1) D3 clears the camped connection.					• servicesPermitted	none	
					Connection Cleared		
					• droppedConnection	D3C3	
					• releasingDevice	D3	
					• localConnectionInfo	null	
					• cause	normalClr	
1) D3 is prompted to go offhook.					• servicesPermitted	none	
					Service Initiated		
					• initiatedConnection	D3C2	
					• initiatingDevice	D3	
					• localConnectionInfo	initiated	
					• cause	transfer	
					• servicesPermitted	Answer, Clear-Conn, Deflect, SendUserInfo	
					• privateEventCause	SST	

5.19.7.4 SSCT call with Await Connect , Pick Up

This scenario describes a call flow when Phone Mail tries to transfers with the Await connect feature. A device in the destinations's group picks the call up.

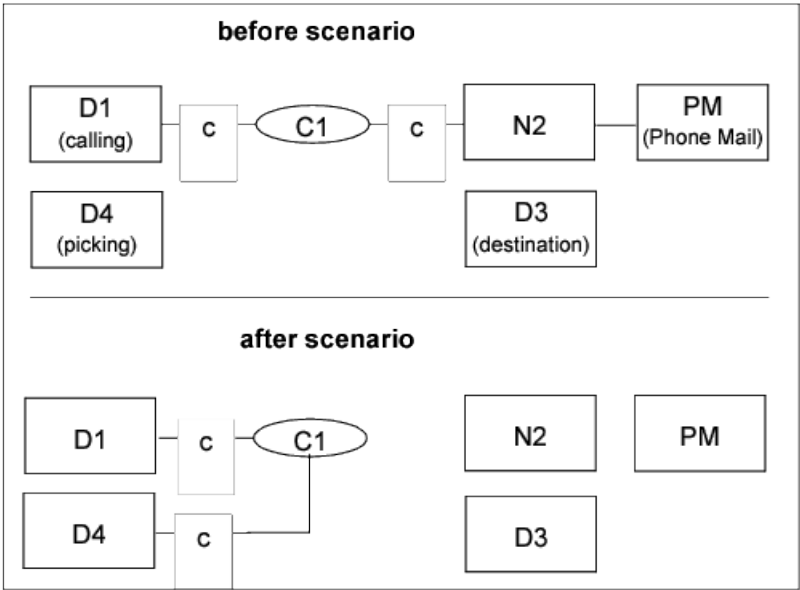


Figure 114: SSCT call with await connect, Pick up

See [Section 5.17.1.3, "Internal ACD call completed to Phone Mail agent"](#) for the event flow to get into the "before scenario" state.

**Table 331: Single Step Call Transfer, Pick Up**

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Monitored Device D4		Comments
1) D3 is prompted to go off-hook.					Service Initiated				
					• initiatedCon-	D3C2			
					• initiatingDe-	D3			
					• localConnec-	initiated			
					• cause	transfer			
					• servicesPer-	Answer, ClearConn, Deflect, SendU-			
1) D4 goes offhook.					• privateEvent-	SST			
					Service Initiated				
					• initiatedCon-	D4C3			
					• initiatingDevice	D4			
					• localConnec-	initiated			
					• cause	normal			
1) Device D4 picks.					• servicesPermit-	ClearConn, DialDgt			
					Connection Cleared				
					• droppedCon-	D3C2			
					• releasingDe-	D3			
					• localConnec-	null			
					• cause	normalClr			
					• servicesPer-	none			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Monitored Device D4		Comments
					• privateEvent-Cause	notPresent			
1) Device D4 clears the call from which it picked up.							Connection Cleared		
							• droppedCon- nection	D4C3	
							• releasingDe- vice	D4	
							• localConne- ctionInfo	null	
							• cause	normalClr	
							• servicesPermit- ted	none	
1) Phone Mail transfers.	Transferred		Transferred						
	• prima- ryOldCall	D1C1	• primaryOld- Call	N2C1					
	• transfer- ringDe- vice	PM	• transfer- ringDevice	PM					
	• trans- ferred- ToDevice	D4	• transferred- ToDevice	D4					
	• trans- ferred- Conne- ctions 1.new 2.new	(D1C1) (D4C1)	• transferred- Connections 1.new 2.new	(D1C1) (D4C1)					
	• localCon- nection- Info	conn	• localConne- ctionInfo	null					
	• cause	SST	• cause	SST					
	• services- Permit- ted	ClearConn, Consult, Hold, SST, GenDgt, GenTel- Tones, SendUse- rInfo	• servicesPer- mitted	none					

Activity	Monitored Device D1		Monitored Device N2		Monitored Device D3		Monitored Device D4		Comments
1) The connection is immediately established.	Established						Established		
	• established-Connection	D4C1					• established-Connection	D4C1	
	• answering-Device	D4					• answeringDevice	D4	
	• callingDevice	D1					• callingDevice	D1	
	• calledDevice	D4					• calledDevice	D4	
	• lastRedirection-Device	PM					• lastRedirection-Device	PM	
	• localConnection-Info	connected					• localConnectionInfo	connected	
	• cause	pickup					• cause	pickup	
	• services-Permitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTone, SendUserInfo					• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTone, SendUserInfo	
	• assoc-Called-Device	N1					• assocCalled-Device	N1	

### 5.19.8 Concept of "presentation indicator for devices" in CSTA events

To provide adaptable solution for every application, CA4000 will have three different solution to handle the presentation indicator for devices. The different solutions can be activated in the Advanced Configuration page of the Connectivity Adapter. The parameter PRESENTATION\_RESTRICTED should be set to the following values:

- "normal" : to provide the old concept as it worked in the past (valid for CSTA III)
- "ignore" : to have the presentation restricted partially ignored (valid for CSTA III); this is only a work around for OpenScape ProCenter, which should also use "private data" from now on

- "private data" : presentation indicator will be represented by private data (valid only for CSTA III interface) e.g.:PRESENTATION\_RESTRICTED = private data
- "extended private data": a new concept of the presentation indicator introduced in OpenScape 4000 V4.0 and V5.0 with that the switch delivers presentation restricted information not only in case of calling and called party. All presentation indicator will be represented by private data (valid only for CSTA III interface) PRESENTATION\_RESTRICTED=private data ALLOW\_ALL\_PRIVATE\_DATA=True
- "special": similar to the functionality of "normal" but provides possibility for the OpenScape ContactCenter (special customer change request for Bundestag) to replace the "not Known" with the given <special value> PRESENTATION\_RESTRICTED=special PRESENTATION\_RESTRICTED\_SPECIAL\_VALUE=<special value>

The application has the choice when to switch-over to one of the "private data". By default the parameter PRESENTATION\_RESTRICTED and ALLOW\_ALL\_PRIVATE\_DATA are not included in the configurable parameters in the Connectivity Adapter's Advanced Configuration page.

If these parameters are not in the config file then the "normal" behaviour will be activated automatically. CA4000 reads the parameter in case of start or restart, so any change will be valid only after the start or restart.

Remark: Presentation restricted info is not stored in CA4000 due to consistency considerations. Therefore OpenScape 4000 CSTA can only handle devices as secret, if the switch informs it explicitly. That means the device might not signaled as restricted, if it was signaled only earlier. In example if secret party A calls B, B does not answer the call, and a ForwardNoAnswer triggers and the call is diverted to C, then A will be signaled in the DIVERTED event as visible.

### 5.19.8.1 The old concept of presentation indicator ("normal")

This concept handles the private data as it was handled in the past. It means that if ACL indicates either the calling or the called party presentation restriction field then CA4000 will report these devices as "Not Known" in the CSTA events. This behaviour is valid for both the CSTA III interface. The following scenarios describe this behaviour:

### 5.19.8.2 Basic Internal Call with presentation restricted devices (CSTA III)

The scenario describes a basic internal call scenario when both devices have presentation restriction.

The parties in this call scenario are:

- Party A: station (such as a digital or analog telephone) (presentation restricted)
- Party B: station (presentation restricted)

Activity	Monitored Device D1		Monitored Device D2	
1) D1 goes off-hook.	Service Initiated			
	• initiatedConnection	NP, C1		

Activity	Monitored Device D1		Monitored Device D2	
	• initiatingDevice	NK		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	ClearConn, DialDgt		
1) D1 completes dialling D2.	Digits Dialed			
	• diallingConnection	NP, C1		
	• diallingDevice	NK		
	• diallingSequence	"1234"		
	• localConnectionInfo	initiated		
	• cause	normal		
	• servicesPermitted	none		
	Originated			
	• originatedConnection	NP, C1		
	• callingDevice	NK		
	• calledDevice	D2		
	• localConnectionInfo	connected		
	• cause	normal		
	• servicesPermitted	ClearConn		
D2 starts ringing.	Delivered		Delivered	
	• connection	NP,C1	• connection	NP, C1
	• alertingDevice	NK	• alertingDevice	NK
	• callingDevice	NK	• callingDevice	NK
	• calledDevice	NK	• calledDevice	NK
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS
	• localConnectionInfo	connected	• localConnectionInfo	alert
	• cause	normal	• cause	normal
D2 answers the call.	Established		Established	
	• establishedConnection	NP, C1	• establishedConnection	NP, C1

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2	
	• answeringDevice	NK	• answeringDevice	NK
	• callingDevice	NK	• callingDevice	NK
	• calledDevice	NK	• calledDevice	NK
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS
	• localConnectionInfo	connected	• localConnectionInfo	connected
	• cause	normal	• cause	normal
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo

### 5.19.8.3 Blind Transfer

The following table shows a call scenario where blind transfer of call is made.

The parties in these call scenarios are:

- Party A: held (presentation restricted)
- Party B: consulting (presentation restricted)
- Party C: consulted (presentation restricted)

**Table 332: Blind Transfer - Consulted Party Answers**

Telephony Activity	Event (Party A)	Event (Party B)	Event (Party C)
Party A and Party B are in conversation.  <b>1)</b> Party B presses the consultation/conference key.	Held <ul style="list-style-type: none"> <li>• CID: 1, NP</li> <li>• Holding DID: NK</li> <li>• LCS: connected</li> <li>• Private cause: consultation hold</li> </ul>	Held <ul style="list-style-type: none"> <li>• CID: 1, B</li> <li>• Holding DID: B</li> <li>• LCS: held</li> <li>• Private cause: consultation hold</li> </ul> Service Initiated <ul style="list-style-type: none"> <li>• CID: 2, B</li> <li>• LCS: initiated</li> </ul>	None

Telephony Activity	Event (Party A)	Event (Party B)	Event (Party C)
1) Party B dials and rings Party C.	None	Delivered <ul style="list-style-type: none"> <li>CID: 2, NP</li> <li>Alerting DID: NK</li> <li>Calling DID: NK</li> <li>Originally called DID: NK</li> <li>Last redirecting DID: NR</li> <li>LCS: connected</li> </ul>	Delivered <ul style="list-style-type: none"> <li>CID: 2, NP</li> <li>Alerting DID: NK</li> <li>Calling DID: NK</li> <li>Originally called DID: NK</li> <li>Last redirecting DID: NR</li> <li>LCS: alerting</li> <li>Private consultation held with party: A</li> </ul>
1) Party B transfers by going on-hook.	Transferred <ul style="list-style-type: none"> <li>Primary old CID: 1, B</li> <li>Transferring DID: B</li> <li>Transferred DID: NK</li> <li>Transferred CID: (3, A); (3, NP)</li> <li>LCS: connected</li> </ul>	Transferred <ul style="list-style-type: none"> <li>Primary old CID: 1, NP</li> <li>Secondary old CID: 2, B</li> <li>Transferring DID: NK</li> <li>Transferred DID: C</li> <li>Transferred CIDs: (3, C); (3, A)</li> <li>LCS: null</li> </ul>	Transferred <ul style="list-style-type: none"> <li>Primary old CID: 2, B</li> <li>Transferring DID: B</li> <li>Transferred DID: C</li> <li>Transferred CIDs: (3, C); (3, NP)</li> <li>LCS: alerting</li> </ul>
1) Party C answers.	Established CID: 3, NP Answering DID: NK Calling DID: NK Originally called DID: NK Last redirecting DID: NR LCS: connected	None	Established <ul style="list-style-type: none"> <li>CID: 3, NP</li> <li>Answering DID: NK</li> <li>Calling DID: NK</li> <li>Originally called DID: NK</li> <li>Last redirecting DID: NR</li> <li>LCS: connected</li> </ul>

#### 5.19.8.4 Conference Initiation by Conference Master with presentation restricted devices

Party A calls Party B and is connected (call ID 1). Party B presses the consultation key and calls Party C (call ID 2). Party A is held. Party B and C are connected.

The parties in this call scenario are

- Party A: held party (presentation restricted)
- Party B: consulting (presentation restricted)
- Party C: consulted (presentation restricted)

**Table 333: Conference Initiation by Conference Master**

Telephony Activity	Event (Party A)	Event (Party B)	Event (Party C)
1) Party B initiates a 3-party conference, pressing the consultation/conference key or requesting the Conference Call service.	Conferenced <ul style="list-style-type: none"> <li>Primary old CID: 1, NP</li> <li>Conference controller: NK</li> <li>Added DID: C</li> <li>Conference CIDs: (3,C)</li> <li>LCS: connected</li> </ul>	Conferenced <ul style="list-style-type: none"> <li>Primary old CID: 1, B</li> <li>Secondary old CID: 2, B</li> <li>Conference controller: B</li> <li>Added DID: NK</li> <li>Conference CIDs: (3,A)</li> <li>LCS: connected</li> </ul>	Conferenced <ul style="list-style-type: none"> <li>Primary old CID: 2, NP</li> <li>Conference controller: NK</li> <li>Added DID: C</li> <li>Conference CIDs: (3,A)</li> <li>LCS: connected</li> </ul>

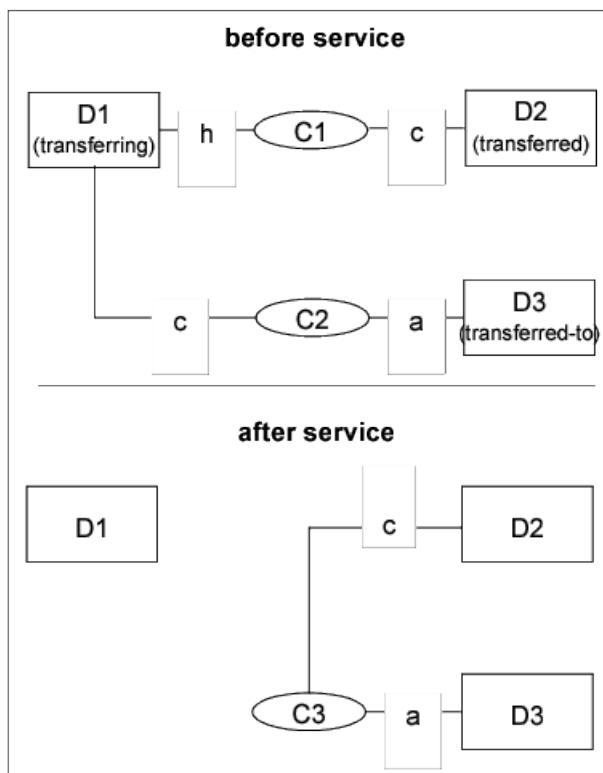
## 5.19.8.5 Blind Transfer with presentation restricted devices (CSTA III)

This service transfers a held party to a consulted party. The transfer service request is issued before the consulted device connects into the new call.

Device D1: presentation restricted

Device D2: presentation restricted

Device D3: presentation restricted



**Figure 115: Blind transfer with presentation restricted devices**

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

**Table 334: Blind Transfer with presentation restricted devices**

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Transfer Call service is invoked on behalf of device D1.	Transfer Call Request						
	• heldConnection	D1C1					
	• activeConnection	D1C2					
1) Acknowledgement.	Transfer Call Response						
	• transferredConnection	D3C3					
1) Calls between D1, D2 and D1, D3 are released. The connections between D2, D1 and D3, D1 are replaced with a single connection between D2 and D3.	Transferred		Transferred		Transferred		The CSTA Transferred event Local View modeling option is provided by the switching function. This means that the primary old call parameters in the Transferred event represent a device oriented view.
	• primaryOldCall	D1C1	• primaryOldCall	D2C1	• primaryOldCall	D3C2	
	• secondaryOldCall	D1C2					
	• transferringDevice	NotKnown	• transferringDevice	D1	• transferringDevice	D1	
	• transferredToDevice	D3	• transferredToDevice	NotKnown	• transferredToDevice	NotKnown	
	• transferredConnections 1. new / old 2. new / old	(D2C3) / (D2C1) (D3C3) / (D3C2)	• transferredConnections 1. new / old 2. new	(C3) / (C1) (D3C3)	• transferredConnections 1. new / old 2. new	(D3C3) / (D3C2) (C3)	
	• localConnectionInfo	null	• localConnectionInfo	connected	• localConnectionInfo	alerting	
	• cause	Transfer	• cause	Transfer	• cause	Transfer	
	• servicesPermitted	none	• servicesPermitted	ClearConn, SendUserInfo	• servicesPermitted	Answer, ClearConn, SendUserInfo	

Remark:

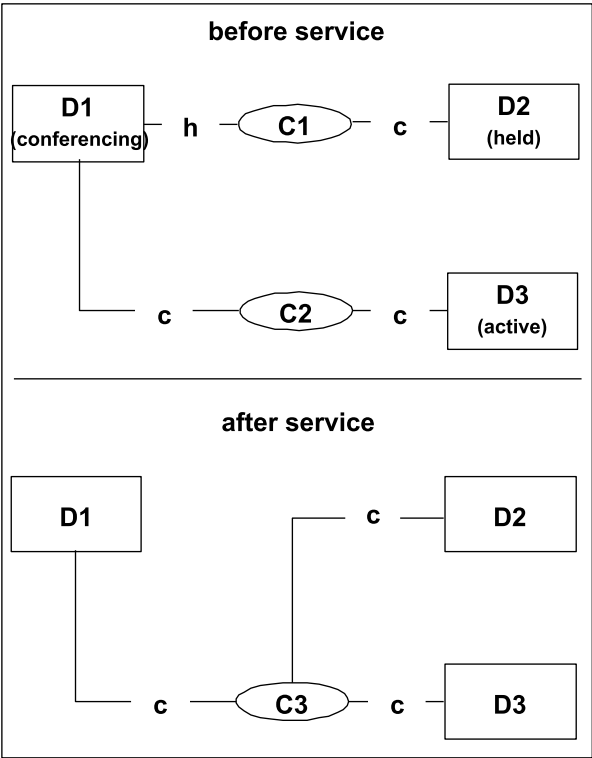
**5.19.8.6 Conference with presentation restricted devices (CSTA III)**

This 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 at the conferencing device.

Device D1: presentation restricted

Device D2: presentation restricted

Device D3: presentation restricted



Conference with presentation restricted device

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

Table 335: Conference with presentation restricted devices

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
Conference Call service is requested on behalf of device D1.	Conference Request						
	• heldConnection	D1C1					
	• activeConnection	D1C2					
Acknowledgement.	Conference Response						
	• conferenced-Connection	D1C3					

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
Conference established.	Conferenced		Conferenced		Conferenced		The addedParty specifies the device ID of the device, that belongs to the active (not held) call of the conference.
	primaryOldCall	D1C1	primaryOldCall	D2C1	primaryOldCall	D3C2	
	secondaryOldCall	D1C2					
	conferencingDevice	NotKnown	conferencingDevice	NotKnown	conferencingDevice	NotKnown	
	Added	NotKnown	Added	D3	Added	NotKnown	Note that the primaryOld-Call and the secondaryOld-Call parameters follows the "local view" modeling option.
	conference-Connections 1. new/old 2. new/old 3. new 4. new	(D1C3)/(D1C1) (D1C3)/(D1C2) (D2C3) (D3C3)	conference-Connections 1. new/old 2. new/old 3. new	(D2C3)/(D2C1) (D3C3)	conference-Connections 1. new/old 2. new/old 3. new	(D1C3)/(D1C2) (D2C3)	
	localConnectionInfo	connected	localConnectionInfo	connected	localConnectionInfo	connected	
	cause	normal	cause	normal	cause	normal	
	servicesPermitted	ClearConn, Consult, Hold, SendUserInfo	servicesPermitted	ClearConn, Consult, Hold, SendUserInfo	servicesPermitted	ClearConn, Consult, Hold, SendUserInfo	

#### Remark:

The manual case is similar to the described event flow.

### 5.19.8.7 Presentation restricted is ignored

This solution was initially developed for the application Procenter. The restriction of the presentation will be partially ignored. It means that the devices with presenstation restriction will be reported in the CSTA III events.

In case of the call leaves the swithcing subdomain (incoming and outgoing calls) the restricted parameters will be reported as "Not Known" although the trunk provides the network party. It means that if a trunk information contains the network party field but its presentation indicator is restricted then CA4000 will not provide this number! The application developer has to take it into consideration when he/she selects this solution.

### 5.19.8.8 Presentation indicator represented by Private Data

The presentation indicator of a device indicates whether the dialling number of a device is allowed or restricted. In case of any restricted presentation reported in the calling or called field of an ACL event CA4000 will provide a private data called Presentation Restricted Device 1 or Presentation Restricted Device 2.

The presentationRestrictedDeviceID1/2 refers to the CSTA calling/called device, that means CA4000 provides restriction information only for the calling, called party. However CA4000 provides restriction information in specific cases for those events (Transferred, Conference, etc), where the CSTA calling/called device do not exist. Due to ACL limitations, CA4000 cannot provide proper restriction information in these cases, but applications can use this additional information as it is.

Remark: The new special extended OpenScape4000 concept of the presentation indicator can provide information about secret devices without any limitation. Please activate the PRESENTATION\_RESTRICTED=private data ALLOW\_ALL\_PRIVATE\_DATA=True in the configuration file and restart the CA4000.

The following scenarios describe the private data representation behaviour. (Old concept!)

### 5.19.8.9 Illustration of the new concept:

In the following scenarios the presentation indicator of Party A and Party B is restricted.

#### 1) Party A calls party B.

Delivered Event:

Calling: A

Called: B

presentationRestrictedDeviceID1: A

presentationRestrictedDeviceID2: B

#### 2) Party A on node1 calls party B on node2.

Delivered Event on monitored party B:

Calling: A

Called: B

presentationRestrictedDeviceID1: A

presentationRestrictedDeviceID2: B

#### 3) Party A on node1 calls party B on node2.

Delivered Event on monitored party A:

Calling: A

Called: B

presentationRestrictedDeviceID1: A

presentationRestrictedDeviceID2: -

### 5.19.8.10 Basic call with presentation restricted devices

This scenario illustrates a call originated through manual device activity.

Device D1: presentation restricted

Device D2: presentation restricted

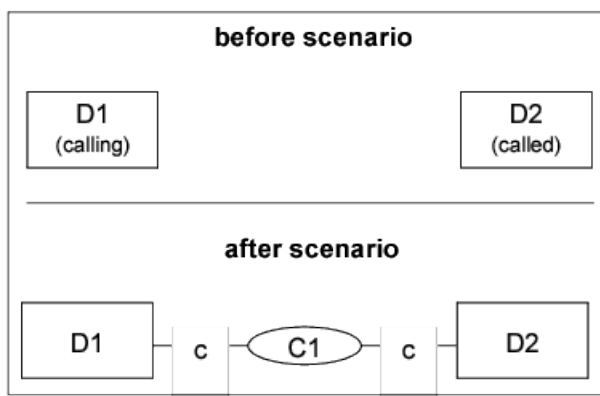


Figure 116: Basic call with presentation restricted devices

Table 336: Basic call with presentation restricted devices

Activity	Monitored Device D1	Monitored Device D2			Comments
1) D1 goes off-hook.	Service Initiated				
	• initiatedConnection	D1C1			
	• initiatingDevice	D1			
	• localConnectionInfo	initiated			
	• cause	normal			
	• servicesPermitted	ClearConn, DialDgt			
	• Presentation Restricted Device1	D1			
1) D1 completes dialling D2.	Digits Dialed				D2's number is 1234
	• diallingConnection	D1C1			
	• diallingDevice	D1			
	• diallingSequence	"1234"			
	• localConnectionInfo	initiated			
	• cause	normal			
	• servicesPermitted	none			
	Originated				
	• originatedConnection	D1C1			
	• callingDevice	D1			
	• calledDevice	D2			
	• localConnectionInfo	connected			
	• cause	normal			

## Call Scenarios

Activity	Monitored Device D1		Monitored Device D2		Comments
	• servicesPermitted	ClearConn			
	• Presentation Restricted Device1	D1			
1) D2 starts ringing.	Delivered		Delivered		
	• connection	D2C1	• connection	D2C1	
	• alertingDevice	D2	• alertingDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	alert	
	• cause	normal	• cause	normal	
	• servicesPermitted	CallBack, ClearConn, SendUserInfo	• servicesPermitted	AnswerCall, ClearConn, De-flect, SendUserInfo	
	• Presentation Restricted Device1	D1	• Presentation Restricted Device1	D1	
1) D2 answers the call.	• Presentation Restricted Device2	D2	• Presentation Restricted Device2	D2	
	Established		Established		
	• establishedConnection	D2C1	• establishedConnection	D2C1	
	• answeringDevice	D2	• answeringDevice	D2	
	• callingDevice	D1	• callingDevice	D1	
	• calledDevice	D2	• calledDevice	D2	
	• lastRedirectionDevice	NS	• lastRedirectionDevice	NS	
	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	normal	• cause	normal	
	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	• servicesPermitted	ClearConn, Consult, Hold, SST, GenDgt, GenTelTones, SendUserInfo	
	• Presentation Restricted Device1	D1	• Presentation Restricted Device1	D1	

Activity	Monitored Device D1		Monitored Device D2		Comments
	• Presentation Restricted Device2	D2	• Presentation Restricted Device2	D2	
1) D2 goes on-hook.	Connection Cleared		Connection Cleared		
	• droppedConnection	D2C1	• droppedConnection	D2C1	
	• releasingDevice	D2	• releasingDevice	D2	
	• localConnectionInfo	connected	• localConnectionInfo	null	
	• cause	normalClr	• cause	normalClr	
	• servicesPermitted	ClearConn	• servicesPermitted	none	
	• Presentation Restricted Device1	D1	• Presentation Restricted Device1	D2	
1) The remaining connection D1C1 goes blocked.	Failed				
	• failedConnection	D1C1			
	• failingDevice	D1			
	• callingDevice	D1			
	• calledDevice	D2			
	• lastRedirectionDevice	NS			
	• localConnectionInfo	fail			
	• cause	blocked			
	• servicesPermitted	ClearConn			
	• Presentation Restricted Device1	D1			
1) D1 goes on-hook.	Connection Cleared				
	• droppedConnection	D1C1			
	• releasingDevice	D1			
	• localConnectionInfo	null			
	• cause	normalClr			
	• servicesPermitted	none			
	• Presentation Restricted Device1	D1			

Remark:

5.19.8.11 Blind Transfer with presentation restricted devices

This service transfers a held party to a consulted party. The transfer service request is issued before the consulted device connects into the new call.

Device D1: presentation restricted

Device D2: presentation restricted

Device D3: presentation restricted

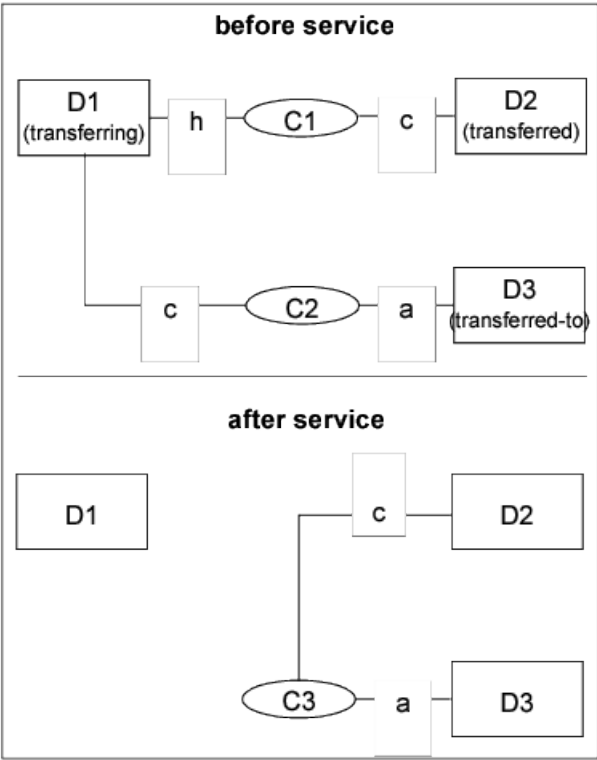


Figure 117: Blind transfer with presentation restricted devices

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

Table 337: Blind Transfer with presentation restricted devices

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	
1) Transfer Call service is invoked on behalf of device D1.	Transfer Call Request					
	• heldConnection	D1C1				
	• activeConnection	D1C2				
1) Acknowledgement.	Transfer Call Response					
	• transferredConnection	D3C3				

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3	
1) Calls between D1, D2 and D1, D3 are released. The connections between D2, D1 and D3, D1 are replaced with a single connection between D2 and D3.	Transferred		Transferred		Transferred	
	• primaryOldCall	D1C1	• primaryOldCall	D2C1	• primaryOldCall	D3C1
	• secondaryOldCall	D1C2				
	• transferringDevice	D1	• transferringDevice	D1	• transferringDevice	D1
	• transferredToDevice	D3	• transferredToDevice	D3	• transferredToDevice	D3
	• transferredConnections 1. new / old 2. new / old	(D2C3) / (D2C1) (D3C3) / (D3C2)	• transferredConnections 1. new / old 2. new	(D2C3) / (D2C1) (D3C3)	• transferredConnections 1. new / old 2. new	(D3C3) / (D3C2)
	• localConnectionInfo	null	• localConnectionInfo	connected	• localConnectionInfo	alert
	• cause	Transfer	• cause	Transfer	• cause	Transfer
	• servicesPermitted	none	• servicesPermitted	ClearConn, SendUserInfo	• servicesPermitted	Answer, Clear, Send
	• Presentation Restricted Device1	D1	• Presentation Restricted Device1	D2	• Presentation Restricted Device1	D2
			• Presentation Restricted Device2	D3	• Presentation Restricted Device2	D3

Remark:

### 5.19.8.12 Conference with presentation restricted devices

This 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 at the conferencing device.

Device D1: presentation restricted

Device D2: presentation restricted

Device D3: presentation restricted

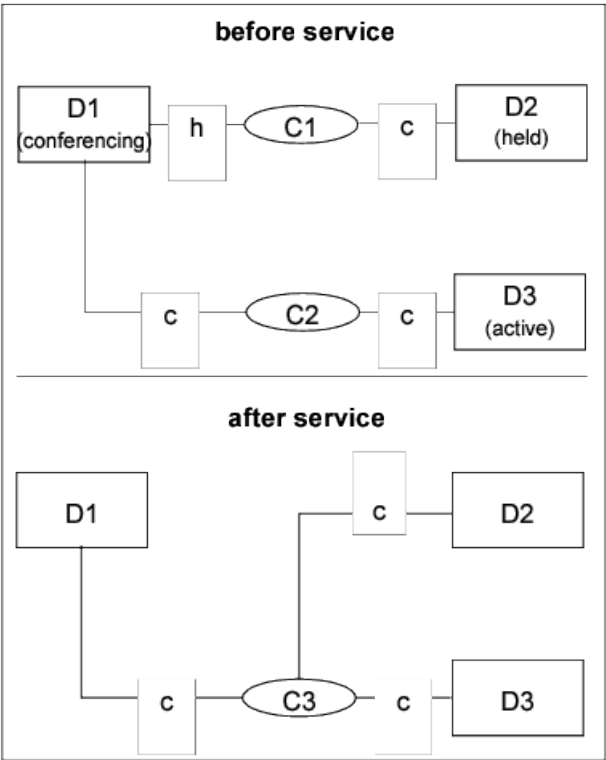


Figure 118: Conference with presentation restricted devices

See [Section 5.13.1, "Successful Consultation Call"](#) for the event flow to get into the "before service" state.

Table 338: Conference with presentation restricted devices

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Conference Call service is requested on behalf of device D1.	Conference Request						
	• heldConnection	D1C1					
	• activeConnection	D1C2					
1) Acknowledgement.	Conference Response						
	• conferencedCon- nection	D1C3					

Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
1) Conference established.	Conferenced		Conferenced		Conferenced		The added-Party specifies the device ID of the device, that belongs to the active (not held) call of the conference.  Note that the primaryOldCall and the secondaryOldCall parameters follows the "local view" modeling option.
	• primaryOldCall	D1C1	• primaryOldCall	D2C1	• primaryOldCall	D3C2	
	• secondaryOldCall	D1C2					
	• conferencingDevice	D1	• conferencing-Device	D1	• conferencingDevice	D1	
	• Added	D3	• Added	D3	• Added	D3	
	• conferenceConnections 1. new/old 2. new/old 3. new 4. new	(D1C3)/(D1C1) (D1C3)/(D1C2) (D2C3) (D3C3)	• conference-Connections 1. new/old 2. new/old 3. new	(D2C3)/(D2C1) (D1C3)/(D1C1) (D3C3)	• conferenceConnections 1. new/old 2. new/old 3. new	(D1C3)/(D1C2) (D3C3)/(D3C2) (D2C3)	
	• localConnectionInfo	connected	• localConnectionInfo	connected	• localConnectionInfo	connected	
	• cause	normal	• cause	normal	• cause	normal	
	• servicesPermitted	ClearConn, Consult, Hold, SendUserInfo	• servicesPermitted	ClearConn, Consult, Hold, SendUserInfo	• servicesPermitted	ClearConn, Consult, Hold, SendUserInfo	
	• Presentation Restricted Device1	D1	• Presentation Restricted Device1	D2	• Presentation Restricted Device1	D1	
	• Presentation Restricted Device2	D3	• Presentation Restricted Device2	D1	• Presentation Restricted Device2	D3	

### Remark:

The manual case is similar to the described event flow.

## 5.19.8.13 Affected events

The following events can be affected by the new private data elements:

Conferenced, Connection Cleared, Delivered, Diverted, Established, Failed, Held, Network Reached, Originated, Queued, Retrieved, Service Initiated, Transferred, Callback

The new private data elements will not be provided in Logical Device Feature events:

Agent Busy, Agent Ready, Agent Not Ready, Agent Working After Call, Agent Logon, Agent Logoff

In the corresponding ACL events there is no ACL calling or called party which could determine the requested private datas.

### 5.19.8.14 Remarks

### 5.19.8.15 Multiple calls

Caller ID blocking/unblocking is supported for multiple calls.

It means that the restriction is always in relation to a call. E.g. if a party goes out to consultation (with call ID 2) and activates caller ID blocking only for the consulted call (call ID 2), in the events for the original call (call ID 1) this party will not have name and number restriction, but for the consultation call (call ID 2) this party will have name and/or number restriction.

### 5.19.8.16 Configuration of presentation indicator

The ways of configuring caller ID blocking for an extension:

### 5.19.8.17 Configure the presentation indicator by AMO

Set

AMO CHA-SBCSU:..., SSTNO=YES (secret station number)

The party will be restricted in the ACL events for all the incoming and outgoing calls (CallingParty and CalledParty).

### 5.19.8.18 Configure the presentation indicator via OptiSet menu

Set Service menu -> More features -> Display suppress on.

The party will be restricted only in the forthcoming call (CallingParty)

## 5.19.9 Connect and Reconnect Timeslot Escape Services

### 5.19.9.1 Connect Timeslot Escape Service

OpenScape 4000 supports the ability to connect a special target device (attached to a trunk) to the voice channel(s) of a source device. The Escape service connect timeslot is used to support this non-standard feature.

Precondition for connect timeslot request: source device is in non-idle state, target device is in talk state (this can be achieved by connecting it to a "Help Party" which can be a virtual device).

Connect Timeslot service supports:

- Joint and Separate mode
- Call and talk oriented timeslot connection

**Separate mode**

At talk oriented mode the source device always has a partner so the timeslot of the source or the partner device is always available.

For call oriented mode the source device may be connected to a tone (e.g. dial tone). In this case if a connect timeslot service with listen channel is started, then the target device will be connected to the same tone. If the service is started with the talk channel, then the target device will be connected to silence. If the automatic reconnect timeslot mechanism detects that the source device got a talking partner, then the timeslot will be connected as in the talk oriented case.

**Joint mode**

If a connect timeslot service with joint mode will be started, then the listen channel and the talk channel of the source device will be mixed using a conference circuit. For joint mode for each source device a conference circuit with two input channels and an output channel will be reserved and used. The output of the conference circuit will be connected to the target device. The input channels of the conference circuit depend on the state of the source device.

**Call and talk oriented Connect Timeslot**

If the connect timeslot service has been started, then the state of the source device will be checked. The service can be started if the source device is in a proper state. The proper state depends on the type of the service (call or talk oriented).

Precondition for using the call oriented ConnectTimeslot request is the talk state of the target device and the non-idle state of the source device.

With the talk oriented connect timeslot service it is only possible to connect the timeslot of a source device to a target device, if the source device is in talk state. Trunks are also supported as source devices. Note that joint mode and talk oriented connection are not work together neither for the listen path nor the talk path.

**5.19.9.2 Reconnect Timeslot Escape Service**

Reconnect Timeslot Escape service stops the connection to the voice channel(s) of the source device.

## 6 Device Identifier Formats

The possible types of Device Identifier formats are:

- Diallable Digits ( DD ) - sequence of characters to be dialled to reach a device Format: dd dd is a string of dialling commands/digits. The following is the table of supported dialling digits/commands:

**Table 339: Supported dialling digits/commands**

0-9	These characters represent 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 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.

- Switching Function Representation ( SFR ) - sequence of characters , that is used to reference devices within a switching sub-domain.

Format: **N< DN >NM;tag=value;tag=value**

The syntax is broken down as follows:

- **N**: the "N" character at the beginning of the Device Identifier string (which is 2 to 22 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".
- **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.
- **;** (semicolon) is the separator character that precedes a tag name;
- **tag** is the specified name of the Name String tag (Tags supported by OpenScape 4000 are defined in table X)
- **=value** is the assigned value of the tag

Field Name	Description
displayNumber	Indicates an optimal dialable form of the Directory  Number (e.g. extension number, public number with route access)

Field Name	Description
	code, etc.). This notation is typically a dialable and displayable number used in call information displays and call logs. This field is present if the station has a PIN activated or it is a HuntGroup member with specific display type or an Agent logged on.
ond	Indicates the sequence of dialable characters used to reach a One Number Service Device (OND) associated with the user's ONS-published number (represented in the Directory Number field).

- **Device Number ( DeN )**- a non-dialable integer representation of a Device Identifier. There are various type of devices with overlapping device number values that must be represented through CSTA device number data types. An encoding scheme is used to identify the device type of a device number value. This scheme uses the most significant byte ( MSB ) of a 4-byte integer to identify the device type.

Device	Device Identifier Format	Device Number Range	Device Number Device Type in MSB	Device Number Device Type mask in HEX
Extension	Diallable Digits / SFR	NA	NA	NA
ACD agent	Diallable Digits / SFR	NA	NA	NA
Attendant Console	Diallable Digits / SFR	NA	NA	NA
ACD Group Number	Device Number	1-255	1	0x01000000
ACD RCG Number	Device Number	1-1020	2	0x02000000
Announcement	Device Number	1-64	3	0x03000000
General Attendant	Device Number	0-999999	4	0x04000000
Hunt Group	Device Number	0-999999	5	0x05000000
Music	Device Number	1-32	6	0x06000000
Parking Slot	Device Number	0-9	7	0x07000000
Trunks	Device Number	0-999999	8	0x08000000
PhoneMail trunk	Device Number	0-9999	8	0x08000000
Server Group	Device Number	0-999	9	0x09000000

## Device Identifier Formats

Device	Device Identifier Format	Device Number Range	Device Number Device Type in MSB	Device Number Device Type mask in HEX
Special device	Device Number	0	10	0x0a000000

---

**NOTICE:** In general, OpenScape 4000 groups are represented by logical devices, because they do not have physical appearances. These groups have assigned at least one dialable number and a (unique) device-number. To monitor the group, the device-number must be used. The device-number issued by OpenScape 4000 is masked as specified in the table above. It is not possible to monitor the dialable number of a group.

---

## 7 Appendix A - Generic Display Protocol

The goal of this protocol definition is to provide a device / subdevice-independent and extensible format of I/O Data. It only applies to the IOData field of Fast-Data-Request and Send-Data-Request services in case of SubAddress = phone (0).

This format (detailed below) will be only taken into account if the I/O Data begins with octal H'FF, otherwise the content of the IOData field will be interpreted as plain text.

The ASN.1 encoding of the content of the IOData parameter in ACL I/O-Service messages is as follows:

- SiemensIO is the definition of the content of the ioData parameter used in the Fast-Data-Request and Send-Data-Request ACL services

```
SiemensIO ::= CHOICE {

    displayContent                [0] IMPLICIT
    DisplayIO

}
DisplayIO ::= SEQUENCE OF DisplayIOData
DisplayIOData ::= [4] IMPLICIT SEQUENCE {

    row                [0] IMPLICIT INTEGER (1..100)
    OPTIONAL, -- (*1)

    column             [1] IMPLICIT INTEGER (1..100)
    OPTIONAL, -- (*1)

    characterAttribute [2] IMPLICIT CharacterAttribute
    OPTIONAL, -- (*2)

    ioContent          [3] IOContent,
    (*3)

    update             [4] IMPLICIT BOOLEAN
    DEFAULT FALSE -- (*4)
```

### Notes:

- (\*1) Start position of content; don't need to be provided; if missing, value 1 is assumed.
- (\*2) the attributes for the display characters (bold, flash, underline, etc.); don't need to be provided; if the display does not support these attributes, it ignores them.
- (\*3) text or symbol
- (\*4) TRUE, if only part of the display is to be overwritten; FALSE, if the display is to be completely erased, before the new content is written; don't need to be provided; if missing, FALSE is assumed; if the display does not support update, it ignores it.

## Appendix A - Generic Display Protocol

**Table 340: Supported Parameters in DisplayIOData**

Parameter in DisplayIOData IE	OPTISET	CMI
row	not supported	supported
column	not supported	supported
characterAttribute	not supported	supported
ioContent	supported	supported
update	not supported	not supported

```

CharacterAttribute ::= BIT STRING {
    bold                (0),
    inverse             (1),
    flash               (2),
    underline           (3),
    italic              (4),
    reserved6           (5),
    reserved7           (6),
    reserved8           (7)
} (SIZE (8))

```

**Table 341: Encoding of the content octet for the CharacterAttribute IE**

Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
reserved8	reserved7	reserved6	italic	underline	flash	inverse	bold

**Table 342: Supported CharacterAttribute Values**

CharacterAttribute Values	OPTISET	CMI
bold	not supported	not supported
inverse	not supported	supported
flash	not supported	not supported
underline	not supported	not supported
italic	not supported	not supported

```

IOContent ::= CHOICE {

```

```

text          [0] IMPLICIT IA5String
              (SIZE(1..227)),          -- (*1)

symbol        [1] IMPLICIT SET (SIZE(1..6)) OF
              SymbolID,                -- (*2)

special       [2] IMPLICIT OCTET STRING (SIZE(1..227))  --
              (*3) }
SymbolID ::= INTEGER (0..255)

```

**Notes:**

- (\*1) ASCII text
- (\*2) softkey (see table 5)
- (\*3) proprietary character string

**Table 343: Supported IOContent choices**

Choices in IOContent	OPTISET	CMI
text	supported	supported
symbol	not supported	supported
special	not supported	not supported

**Table 344: Possible SymbolID values for CMI devices**

SymbolID for CMI	OPTISET
(0)	Softkeys are unchanged.
(1)	No softkeys will be offered.
(2)	Only the softkey OK will be offered.
(3)	Softkey OK and ESC will be offered.
(4)	Every softkeys with OK at the end will be offered.
(5)	Every softkeys with Optionmenu at the end will be offered.

**Implementation Notes:**

- In ACL the IOData IE is defined as an octet string. This proposal defines the contents of this octet string. It does not replace the octet string.
- This ASN.1-proposal is the abstract and private format of an unstructured data field in an ACL message.
- To realize extensibility of this format, it is necessary to ignore unknown fields in the message. This has to be taken into account when implementing the ASN.1-Interpreter for message analysis. The usual behavior of an ASN.1-Interpreter would be to abort message analysis when encountering an unknown field. Therefore a compiler-generated ASN.1-Interpreter cannot be employed.
- When encoding the I/O Data it has to be assured that the complete length of the I/O Data does not exceed the limit of 240 byte.

- The behavior when reaching the end of the line, is defined as follows:
- If the string exceeds the capacity of the line, the exceeding characters are printed in the following line. If this occurs in the last line of the display, the string is truncated.
- If two strings overlap, the second overwrites the first

### Example:

The following example shows the encoding of the string "Hello world!" with character attributes bold and underline, starting in line 2 on position 4 of the display.

```

8e 1e                                IOData IE
    ff (*1)
    a0 1b                            SiemensIO = display = SEQUENCE OF
DisplayIOData
    a4 19                            DisplayIOData = SEQUENCE
    80 01 02                          row =
2                                     (*2)
    81 01 04                          column =
4                                     (*2)
    82 01 09                          characterAttribute = bold and
underline (*2)
    a3 0e                            ioContent
    80 0c 48 65 6c 6c 6f 20 77 6f 72 6c 64 21      text
  
```

### Notes:

(\*1) First byte of I/O Data field; Escape Character

(\*2) The Optiset device does not support neither positioning nor character attributes.

## 8 Appendix B - Representation of keys in the IOData field

This chapter describes the octal representation of pressed keys in the IOData field of the Send-Data-Request (S->C) service during I/O sessions. The representation depends on the class of the key and the device type, following the given rules:

- Functional keys (see [Table 342 "Functional Keys"](#))
- Key representation: 24 xx nn, where xx is the hexadecimal code for identifying the appropriate key and nn is the logical key number (see [Table 346 "Special Keys"](#)). In this case the I/O Data contains only this key information, and the send data event is sent immediately, i.e. buffering is not supported.
- ISO alphanumeric keys (see [Table 343 "ISO Alphanumeric Keys"](#))
- The pushed keys will be buffered depending on the send mode. The NumberOfCharsToCollect specifies the number of characters to collect before sending collected characters on the data path. If its value equals to 1 the overlapped sending mode is used. Buffering is supported exclusively for the characters 0 to 9, \* and #. For all the other alphanumeric keys buffering is not supported.
- Anate Special keys (see [Table 344 "Anate Special Keys"](#))
- Key representation: 25 xx, where xx is hexadecimal code for identifying the appropriate key. In this case the I/O Data contains only this key information, and the send data event is sent immediately, i.e. buffering is not supported.
- Special keys (see [Table 345 "Special Keys"](#))
- The value of NumberOfCharsToCollect is irrelevant, the message is sent immediately.
- NonVoice key
- The NonVoice may cause start, stop or resume of an I/O session. It will not be sent to the application as I/O Data information in a send data event.
- Other keys
- All other keys will not be sent to the application, only a "Not Possible" message may be sent to the device display.

The following tables show the octal data representation for the specified keys. The device type columns shows if the key is supported for the given device type.

**Table 345: Functional Keys**

Functional Key	Octal Data	Anate	Digite	Digite in voice mode	CMI
CancelKey	24 3F nn*	n/a	supported	not supported	n/a
ConsultationKey	24 40 nn*	n/a	supported	not supported	n/a
CallTransferKey	24 41 nn*	n/a	supported	not supported	n/a
ClearKey	24 42 nn*	n/a	supported	not supported	n/a
CheckKey	24 44 nn*	n/a	supported	not supported	n/a
StartKey	24 46 nn*	n/a	supported	not supported	n/a
NumberRedialKey	24 48 nn*	n/a	supported	not supported	n/a

## Appendix B - Representation of keys in the IOData field

Functional Key	Octal Data	Anate	Digite	Digite in voice mode	CMI
NameKey	24 49 nn*	n/a	supported	not supported	n/a
DirectCallKey	24 4A nn*	n/a	supported	not supported	n/a
MailBoxKey	24 4D nn*	n/a	supported	not supported	n/a
CallBackKey	24 4E nn*	n/a	supported	not supported	n/a
ParkKey	24 4F nn*	n/a	supported	not supported	n/a
CallForwKey	24 55 nn*	n/a	supported	not supported	n/a
UnProgrammedKey	24 62 nn*	n/a	supported	not supported	n/a
* nn means the logical number of the pushed button (see <a href="#">Table 346 "Special Keys" on page 659</a> )					

**Table 346: ISO Alphanumeric Keys**

Alphanumeric Key	Octal Data	Anate	Digite	Digite in voice mode	CMI
1	31	supported	supported	supported	supported
2	32	supported	supported	supported	supported
3	33	supported	supported	supported	supported
4	34	supported	supported	supported	supported
5	35	supported	supported	supported	supported
6	36	supported	supported	supported	supported
7	37	supported	supported	supported	supported
8	38	supported	supported	supported	supported
9	39	supported	supported	supported	supported
0	30	supported	supported	supported	supported
*	2A	supported	supported	supported	supported
#	23	supported	supported	supported	supported
a	61	n/a	supported	not supported	n/a
b	62	n/a	supported	not supported	n/a
c	63	n/a	supported	not supported	n/a
d	64	n/a	supported	not supported	n/a
e	65	n/a	supported	not supported	n/a
f	66	n/a	supported	not supported	n/a

## Appendix B - Representation of keys in the IOData field

Alphanumeric Key	Octal Data	Anate	Digite	Digite in voice mode	CMI
g	67	n/a	supported	not supported	n/a
h	68	n/a	supported	not supported	n/a
i	69	n/a	supported	not supported	n/a
j	6A	n/a	supported	not supported	n/a
k	6B	n/a	supported	not supported	n/a
l	6C	n/a	supported	not supported	n/a
m	6D	n/a	supported	not supported	n/a
n	6E	n/a	supported	not supported	n/a
o	6F	n/a	supported	not supported	n/a
p	70	n/a	supported	not supported	n/a
q	71	n/a	supported	not supported	n/a
r	72	n/a	supported	not supported	n/a
s	73	n/a	supported	not supported	n/a
t	74	n/a	supported	not supported	n/a
u	75	n/a	supported	not supported	n/a
v	76	n/a	supported	not supported	n/a
w	77	n/a	supported	not supported	n/a
x	78	n/a	supported	not supported	n/a
y	79	n/a	supported	not supported	n/a
z	7A	n/a	supported	not supported	n/a
A	41	n/a	supported	not supported	n/a
B	42	n/a	supported	not supported	n/a
C	43	n/a	supported	not supported	n/a
D	44	n/a	supported	not supported	n/a
E	45	n/a	supported	not supported	n/a
F	46	n/a	supported	not supported	n/a
G	47	n/a	supported	not supported	n/a
H	48	n/a	supported	not supported	n/a
I	49	n/a	supported	not supported	n/a
J	4A	n/a	supported	not supported	n/a

## Appendix B - Representation of keys in the IOData field

Alphanumeric Key	Octal Data	Anate	Digite	Digite in voice mode	CMI
K	4B	n/a	supported	not supported	n/a
L	4C	n/a	supported	not supported	n/a
M	4D	n/a	supported	not supported	n/a
N	4E	n/a	supported	not supported	n/a
O	4F	n/a	supported	not supported	n/a
P	50	n/a	supported	not supported	n/a
Q	51	n/a	supported	not supported	n/a
R	52	n/a	supported	not supported	n/a
S	53	n/a	supported	not supported	n/a
T	54	n/a	supported	not supported	n/a
U	55	n/a	supported	not supported	n/a
V	56	n/a	supported	not supported	n/a
W	57	n/a	supported	not supported	n/a
X	58	n/a	supported	not supported	n/a
Y	59	n/a	supported	not supported	n/a
Z	5A	n/a	supported	not supported	n/a

**Table 347: Anate Special Keys**

Anate Special Key	Octal Data	Anate	Digite	Digite in voice mode	CMI
"A"	25 41	supported	n/a	n/a	n/a
"B"	25 42	supported	n/a	n/a	n/a
"C"	25 43	supported	n/a	n/a	n/a
"D"	25 44	supported	n/a	n/a	n/a

**Table 348: Special Keys**

Special Keys	Octal Data	Anate	Digite	Digite in voice mode	CMI
OK Key	F3	n/a	supported	not supported	supported
Arrow Left	F4	n/a	supported	not supported	supported
Arrow Right	F5	n/a	supported	not supported	supported
Plus	F6	n/a	supported	not supported	n/a

Special Keys	Octal Data	Anate	Digite	Digite in voice mode	CMI
Minus	F7	n/a	supported	not supported	n/a
Escape	F7	n/a	n/a	n/a	supported

#### Example:

```

Device Type: OPTISET
NumberOfCharsToCollect = 2
pressed keys: "Q", "1", "2", "Arrow Left", "MailboxKey"
IOData fields of the four Send-Data-Request messages (in
hex):
send data 1: "51"
send data 2: "31 32"
send data 3: "F4"
send data 4: "24 4D nn*"
*) nn is the logical key number (see table 5 below)

```

**Table 349: Special Keys**

Logical Key Number	Usage
1 .. 19	Keys on the phone itself
20	Not Used
21 .. 36	Keys on the first add-on key module
37 .. 40	Not Used
41 .. 56	Keys on the second add-on key module
57 .. 60	Not Used
61 .. 76	Keys on the third add-on key module
77 .. 80	Not Used
81 .. 96	Keys on the fourth add-on key module
97 .. 127	Not Used

## 9 Appendix C - Private Data

In ECMA-285 the optional parameter extensions (of type CSTACommonArguments) can be used to encode switching function specific private data.

It contains a CSTAPrivateData type of the following definition:

```
CSTAPrivateData ::= CHOICE
{  string      OCTET STRING,
   private     NULL}
```

The standard allows to replace NULL with another valid ASN.1 type. OpenScape 4000 V10 replaces NULL with the following structure CSTASiemensPrivateData.

**Table 350: CSTA Private Data Parameters**

Parameter Name	Contents	M/C/O	Comments
manufacturer	Octets	M	Must contain the defined string "Siemens CAP"
category	Choice Structure	M	<p>Specifies the service/event category the private data parameters correspond to. This shall be one of the following choices:</p> <ul style="list-style-type: none"> <li>capExchangeServ (CapExchangeServPrivParams) - Capability Exchange Services</li> <li>systemServ (SystemStatusServPrivParams) - System Services</li> <li>monitoringServ (MonitoringServPrivParams) - Monitoring Services</li> <li>snapshotServ (SnapshotServPrivParams) - Snapshot Services</li> <li>callControlServ (CallControlServPrivParams) - Call Control Services</li> <li>callControlEvts (CallControlEvtsPrivParams) - Call Control Events</li> <li>callAssociatedServ (NULL) - Call Associated Services</li> <li>callAssociatedEvts (BasicEvtsPrivParams) - Call Associated Events</li> <li>mediaServ (NULL) - Media Services</li> <li>mediaEvts (NULL) - Media Events</li> <li>routeingServ (NULL) - Routeing Services</li> </ul>

Parameter Name	Contents	M/C/O	Comments
			<ul style="list-style-type: none"> <li>• physDevServ (NULL) - Physical Device Features Services</li> <li>• physDevEvts (BasicEvtsPrivParams) - Physical Device Features Events</li> <li>• logicalServ (NULL) - Logical Device Feature Services</li> <li>• logicalEvts (BasicEvtsPrivParams) - Logical Device Feature Events</li> <li>• deviceMaintEvts (BasicEvtsPrivParams) - Device Maintenance Events</li> <li>• iOServicesServ (IOServicesServPrivParams) - I/O Services</li> <li>• dataCollectionServ (NULL) - Data Collection Services</li> <li>• voiceUnitServ (NULL) - Voice Unit Services</li> <li>• voiceUnitEvts (NULL) - Voice Unit Events</li> <li>• cdrServ (NULL) - Call Detail Record Services</li> <li>• vendorSpecificServ (VendorSpecificServPrivParams) - Vendor Specific Services</li> <li>• vendorSpecificEvts (VendorSpecificEvtsPrivParams) - Vendor Specific Events</li> </ul>

---

**NOTICE:** The xsd and the ASN.1 descriptor of the private data presented here are released alongside with the current documentation.

---

## 9.1 CapExchangeServPrivParams

The Capability Exchange Services can contain the following private data structure.

**Table 351: CapExchangeServPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
trunkGroup	Value	O	Identifies the group of a trunk

## 9.2 SystemStatusServPrivParams

The System Status Services can contain the following private data structure.

**Table 352: SystemStatusServPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
applicationName	Characters	O	Name of the application using the heartbeat mechanism.
routeDestinationWhenLossOfHeartbeat	DeviceID	O	Destination to which the calls will be redirected in case of application or link failure.  Permitted destinations: <ul style="list-style-type: none"> <li>dialingNumber</li> <li>deviceNumber of AccRouteControlGroup, GeneralAttendant, HuntGroup</li> </ul>
switchRestartType	Enumerated	O	This parameter will be sent after a link brokenage or a switch restart.  Possible values: <ul style="list-style-type: none"> <li>switchRestartTypeSoftSymplex (0)</li> <li>switchRestartTypeSoftDuplex (1)</li> <li>switchRestartTypeHard (2)</li> <li>switchRestartTypeServerOnly (3)</li> <li>switchRestartTypeNone (4) (only link break)</li> <li>switchRestartTypeReload (5)</li> </ul>

## 9.3 MonitoringServPrivParams

The Monitoring Services can contain the following private data structure.

**Table 353: MonitoringServPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
privateEvents	Bit String	O	privateEvents can contain a mobileUserStatusEvent, see in <a href="#">Table 357 "MobileUserStatusEvent Parameters" on page 673</a>

## 9.4 SnapshotServPrivParams

The Snapshot Services can contain the following private data structure.

**Table 354: SnapshotServPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
numberOfQueuedCalls	Value	O	Number of queued calls if the snapshot is for an ACD Group number.

Parameter Name	Contents	M/C/O	Comments
mobileUserDirectoryNumber	Characters (64)	O	Directory number of a "mobile user" identifying himself at another phone

## 9.5 CallControlEvtsPrivParams

The Call Control Events can contain the following private data structure.

**Table 355: CallControlEvtsPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
executiveDevice	CallingDeviceID	O	Device ID for OpenScape 4000 CHESE functionality
acdDnis	CalledDeviceID	O	Internal DNIS for the monitor of an RCG; provided whenever ACL provides it
privateEventCause	Enumerated	O	The following private event-causes are supported: a) singleStepCallTransfer (value = 0)
queueNumberGA	Characters (6)	O	GA-Queue-Number for a call. It is provided with the following events: a) the first Delivered-Event at an Attendant Console b) the first NW-Reached-Event in case GA distributes the call to an external party. c) the first Queued-Event at the GA  The number provided by ACL is dependent upon switch-configuration. Please note: this Private Data is only sent to the monitor of the GA
presentationRestrictedDeviceID1	CallingDeviceID	O	Restricted device ID of calling party
presentationRestrictedDeviceID2	CalledDeviceID	O	Restricted device ID of called party
relatedCallLinkageData	CallLinkageData	O	Related call linkage data in case of survivability
presentationRestrictedDeviceID3	SubjectDeviceID	O	Restricted device ID of held party
presentationRestrictedDeviceID4	SubjectDeviceID	O	Restricted device ID of redirecting party
presentationRestrictedDeviceID5	SubjectDeviceID	O	Restricted device ID of redirection party

## Appendix C - Private Data

Parameter Name	Contents	M/C/O	Comments
presentationRestrictedDeviceID6	SubjectDeviceID	O	Restricted device ID of redirected party
presentationRestrictedDeviceID7	SubjectDeviceID	O	Restricted device ID of releasing party
presentationRestrictedDeviceID8	SubjectDeviceID	O	Restricted device ID of added party
presentationRestrictedDeviceID9	SubjectDeviceID	O	Restricted device ID of adding party
presentationRestrictedDeviceID10	SubjectDeviceID	O	Restricted device ID of ACD supervisor extension
presentationRestrictedDeviceID11	SubjectDeviceID	O	Restricted group number
presentationRestrictedDeviceID12	SubjectDeviceID	O	Restricted device ID of originally called party
presentationRestrictedDeviceID13	SubjectDeviceID	O	Restricted device ID of other party
presentationRestrictedDeviceID14	SubjectDeviceID	O	Restricted device ID of other redirecting party
presentationRestrictedDeviceID15	SubjectDeviceID	O	Restricted device ID of released party list member 1
presentationRestrictedDeviceID16	SubjectDeviceID	O	Restricted device ID of released party list member 2
presentationRestrictedDeviceID17	SubjectDeviceID	O	Restricted device ID of released party list member 3
presentationRestrictedDeviceID18	SubjectDeviceID	O	Restricted device ID of released party list member 4
presentationRestrictedDeviceID19	SubjectDeviceID	O	Restricted device ID of released party list member 5
presentationRestrictedDeviceID20	SubjectDeviceID	O	Restricted device ID of released party list member 6
presentationRestrictedDeviceID21	SubjectDeviceID	O	Restricted device ID of conference member list 1
presentationRestrictedDeviceID22	SubjectDeviceID	O	Restricted device ID of conference member list 2
presentationRestrictedDeviceID23	SubjectDeviceID	O	Restricted device ID of conference member list 3
presentationRestrictedDeviceID24	SubjectDeviceID	O	Restricted device ID of conference member list 4
presentationRestrictedDeviceID25	SubjectDeviceID	O	Restricted device ID of conference member list 5
presentationRestrictedDeviceID26	SubjectDeviceID	O	Restricted device ID of conference member list 6

Parameter Name	Contents	M/C/O	Comments
physicalAnsweringDeviceID	Characters (12)	O	Used in a work around for voice recording of the physical answering device in a key system
userToUserInformation	Characters (129)	O	User to user information associated to a call in octetString format
newHGDestination	SubjectDeviceID	O	Indicates the next alerting hunt group member
seamlessHandover	Boolean	O	Indicates if Seamless Handover feature is involved

## 9.6 IOServicesServPrivParams

The I/O Services can contain the following private data structure.

**Table 356: IOServicesServPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
displayMode	Enumerated	O	Specifies if the new display characters are displayed until a system-defined time-out period (temporary display mode) or if they should be displayed until the user resets the display via the check key or non-voice key (fixed display mode). <ul style="list-style-type: none"> <li>temporaryMode (0)</li> <li>fixedMode (1)</li> </ul>
audibleIndication	Enumerated	O	Specifies what type of audible indication should be given to a device. <ul style="list-style-type: none"> <li>silentIndication (0)</li> <li>beepIndication (1)</li> </ul>
applicationId	Characters (3)	O	Specifies the application identification number. This can be the number the user dials to initiate an I/O session
mobileUserDirectoryNumber	Characters (64)	O	Directory Number of a "mobile user" identifying himself at another phone
localIdMode	Boolean	O	Indicates whether the switch has to start the Local ID Procedure in the phone or not.

## 9.7 VendorSpecificServPrivParams

The Vendor Specific Services can contain the following private data choice structure.

Table 357: VendorSpecificServPrivParams Parameters

Parameter Name	Contents	M/C/O	Comments
conenctTimeslotArguments	Structure	C	<p>This request is sent by an application to connect the specified timeslot (listen/talk) of a source device to the listen timeslot of a destination device.</p> <p>Specifies the parameters of a timeslot connection request. They are the following:</p> <ul style="list-style-type: none"> <li>sourceConnection (Conenction ID, M) - specifies the listening connection</li> <li>sourceDirection (Enumerated, M) - specifies the direction of the listening. Possible values are listenDirection (0) talkDirection (1) jointDirection (2)</li> <li>destinationConnection (Connection ID, M) - specifies the listened connection</li> <li>tslConnectionType (Enumerated, O) -specifies the connection type. Possible values are talkOriented (0) callOriented (1)</li> </ul>
connectTimeslotResult	NULL	C	The response contains no information
reconnectTimeslotArgument	Structure	C	<p>This request is sent by an application to restore the timeslot connections in the destination device to the state they were before the execution of the Connect Timeslot service. This means that after the execution of the Reconnect Timeslot service, the destination device will listen to the same device it was listening to before the execution of the Connect Timeslot service.</p> <p>Specifies the parameters of a Reconnect timeslot request. They are the following:</p> <ul style="list-style-type: none"> <li>destinationConenction (ConnectionID, M) - specifies the previously listened connection, that is about to be conencted again</li> </ul>
reconnectTimeslotResult	NULL	C	The response contains no information

Parameter Name	Contents	M/C/O	Comments
setLowerClassOfServiceArgument	Structure	C	Specifies the parameters of a Set Lower COS request. They are the following: <ul style="list-style-type: none"> <li>extensionNumber (Characters (64), M) - specifies the target device</li> <li>lowerCosOn (Boolean) - specifies if the service is to be activated</li> </ul>
setLowerClassOfServiceResult	NULL	C	The response contains no information
getLowerClassOfServiceArgument	Structure	C	Specifies the parameters of a Get LowerCOS request. They are the following: <ul style="list-style-type: none"> <li>extensionNumber(Characters (64), M) - specifies the target device</li> </ul>
getLowerClassOfServiceResult	Structure	C	Specifies the parameters in a Get LowerCOS response. They are the following: <ul style="list-style-type: none"> <li>lowerCosOn (Boolean) - specifies if the service is to be activated</li> </ul>
subaddressArgument	Structure	C	Specifies the parameters of a Subaddress request. They are the following: <ul style="list-style-type: none"> <li>callingPartySubaddress (Characters (22), O)</li> <li>calledPartySubaddress (Characters (22), O)</li> <li>callFacilities (CallFacilities) -- see in <a href="#">Table 355 "CallFacility" on page 672</a></li> </ul>
subaddressResult	NULL	C	The response contains no information
reroutePreventionArgument	Structure	C	Specifies the parameters of a Reroute Prevention Request. They are the following: <ul style="list-style-type: none"> <li>reroutePrevention (Boolean, M) - specifies if the feature is on or off</li> <li>deviceId (DeviceID, M) - specifies the target device</li> </ul>
reroutePreventionResult	NULL	C	The response contains no information
bargeInConferenceCallArgument	Structure	C	Specifies the connections to be connected into a Barge In Conference: <ul style="list-style-type: none"> <li>activeConnectionId1 (ConnectionID, M)</li> <li>activeConnectionId2 (ConnectionID, M)</li> </ul>

## Appendix C - Private Data

### VendorSpecificEvtsPrivParams

Parameter Name	Contents	M/C/O	Comments
bargeInConferenceCallResult	Structure	C	Contains the connection ID of the resulting conference: <ul style="list-style-type: none"> <li>bargeInConnectionId (ConnectionID, M)</li> </ul>
serviceNumberToDisplay-Argument	Structure	C	Specifies the parameters of a Number To Display Request. They are the following: <ul style="list-style-type: none"> <li>numberToDisplay (Characters (22), O)</li> <li>nameToDisplay (Characters (31), O)</li> </ul> Supported charcters are a..z A..Z 0..9 -(minus) (space)
userToUserInformation	Character (129)	C	User to user information provided in a Make Call Request in octetString format

## 9.7.1 CallFacilities

CallFacilities is a List of Structure that can contain one or more CallFacility structures:

**Table 358: CallFacility**

Parameter Name	Contents	M/C/O	Comments
CallFacility	Enumerated	M	The values can be: <ul style="list-style-type: none"> <li>isdnClearChannel (0)</li> <li>noToneAtCallingPty (1)</li> <li>seamlessHandover (2)</li> <li>autoAnswer (3)</li> </ul>

## 9.8 VendorSpecificEvtsPrivParams

The Vendor Specific Events are defined by the following private data structure.

**Table 359: VendorSpecificEvtsPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
VendorSpecificEvtsPrivParams	Choice Structure	M	Specifies the private event. This shall be one of the following choices <ul style="list-style-type: none"> <li>mobileUserStatusEvent (MobileUserStatusEvent)</li> </ul>

## 9.8.1 MobileUserStatusEvent

The MobileUserStatusEvent can contain the following private data structure.

**Table 360: MobileUserStatusEvent Parameters**

Parameter Name	Contents	M/C/O	Comments
physicalDevice	DeviceID	M	Specifies the device at which the "mobile user" has logged in.
mobileUserStatusOn	Boolean	M	Specifies if the Mobile User feature is activated.
mobileUserDirectoryNumber	Characters (64)	O	Specifies the directory number of the "mobile user" who has logged in.

## 9.9 CallControlServPrivParams

The Call Control Service can contain the following private data choice structure.

**Table 361: CallControlService Parameters**

Parameter Name	Contents	M/C/O	Comments
deflectCallArgument	Structure	C	Private parameter in a Deflect Call Service: <ul style="list-style-type: none"> <li>divertedDevice (DeviceID, O)</li> </ul>
deviceList	List of DeviceIDs	C	List of preferred devcies for UC application

## 9.10 BasicEvtsPrivParams

The Call Associated Events, Physical Device Events, Logical Events and Device Maintain Events are following the Basic Events parameter structure.

## Appendix C - Private Data

**Table 362: BasicEventsPrivParams Parameters**

Parameter Name	Contents	M/C/O	Comments
presentationRestrictedDeviceID1	CallingDeviceID	O	Restricted device ID of calling party
presentationRestrictedDeviceID2	CalledDeviceID	O	Restricted device ID of called party
presentationRestrictedDeviceID3	SubjectDeviceID	O	Restricted device ID of held party
presentationRestrictedDeviceID4	SubjectDeviceID	O	Restricted device ID of redirecting party
presentationRestrictedDeviceID5	SubjectDeviceID	O	Restricted device ID of redirection party
presentationRestrictedDeviceID6	SubjectDeviceID	O	Restricted device ID of redirected party
presentationRestrictedDeviceID7	SubjectDeviceID	O	Restricted device ID of releasing party
presentationRestrictedDeviceID8	SubjectDeviceID	O	Restricted device ID of added party
presentationRestrictedDeviceID9	SubjectDeviceID	O	Restricted device ID of adding party
presentationRestrictedDeviceID10	SubjectDeviceID	O	Restricted device ID of ACD supervisor extension
presentationRestrictedDeviceID11	SubjectDeviceID	O	Restricted group number
presentationRestrictedDeviceID12	SubjectDeviceID	O	Restricted device ID of originally called party
presentationRestrictedDeviceID13	SubjectDeviceID	O	Restricted device ID of other party
presentationRestrictedDeviceID14	SubjectDeviceID	O	Restricted device ID of other redirecting party
presentationRestrictedDeviceID15	SubjectDeviceID	O	Restricted device ID of released party list member 1
presentationRestrictedDeviceID16	SubjectDeviceID	O	Restricted device ID of released party list member 2
presentationRestrictedDeviceID17	SubjectDeviceID	O	Restricted device ID of released party list member 3
presentationRestrictedDeviceID18	SubjectDeviceID	O	Restricted device ID of released party list member 4
presentationRestrictedDeviceID19	SubjectDeviceID	O	Restricted device ID of released party list member 5
presentationRestrictedDeviceID20	SubjectDeviceID	O	Restricted device ID of released party list member 6
presentationRestrictedDeviceID21	SubjectDeviceID	O	Restricted device ID of conference member list 1
presentationRestrictedDeviceID22	SubjectDeviceID	O	Restricted device ID of conference member list 2
presentationRestrictedDeviceID23	SubjectDeviceID	O	Restricted device ID of conference member list 3

Parameter Name	Contents	M/C/O	Comments
presentationRestrictedDeviceID24	SubjectDeviceID	O	Restricted device ID of conference member list 4
presentationRestrictedDeviceID25	SubjectDeviceID	O	Restricted device ID of conference member list 5
presentationRestrictedDeviceID26	SubjectDeviceID	O	Restricted device ID of conference member list 6

## 9.11 LogicalServPrivParams

The Logical Service can contain the following private data choice structure.

**Table 363: LogicalServPriv Parameters**

Parameter Name	Contents	M/C/O	Comments
dndResume <sup>24</sup>	Value	O	Specifies the number of minutes the DND feature will stay active
staticOndActive <sup>25</sup>	Boolean	O	The flag specifies if StaticOND feature is active or not
staticOndDN 2	Characters	O	Specifies the directory number of preferred device

<sup>24</sup> Supported from OpenScape 4000 V7R2

<sup>25</sup> Supported from OpenScape 4000 V8R1. Static OND is a feature used by Open Scape Contact Center, but all applications can receive these parameters as private data. To turn this feature off, set DISABLE\_STATIC\_OND flag to YES in Connectivity Adaptor Advanced Configuration.

## 10 Appendix D - Support of uaCSTA

Starting with OpenScape 4000 V10R1, to support the full feature set of UC-clients (Fusion4Office and UC WebRTC Client) uaCSTA is available.

OpenScape 4000 supports the uaCSTA Requests listed below:

- Answer Call Request
- Clear Connection Request – not used
- Consultation Call Request
- Hold Call Request
- Make Call Request - only with Auto Answer enabled
- Retrieve Call Request

---

**NOTICE:** uaCSTA Events are not supported.

---

The features which are not supported yet are listed below:

- Generate Digits Request
- Get Speaker Volume
- Set Speaker Volume
- Get Microphone Mute
- Set Microphone Mute

In case of OpenScape 4000, uaCSTA processing is activated by default for uaCSTA enabled SIP devices. To enable/disable uaCSTA on SIP devices, there are corresponding options in the DLS/ WBM/ administration menu of the SIP device.

---

**NOTICE:** STMI boards are not uaCSTA capable. If a SIP device with uaCSTA capabilities registers on a classic STMI board, uaCSTA is not enabled.

---

### **CSTA Alternate and Reconnect for uaCSTA enabled SIP devices**

CSTA Alternate Request for uaCSTA enabled SIP device is processed as uaCSTA Hold Request followed by uaCSTA Retrieve Request.

CSTA Reconnect Request for uaCSTA enabled SIP device is processed as uaCSTA Clear Connection Request followed by uaCSTA Retrieve Request.

### **Restrictions:**

- WebRTC and Fusion WebClient behaves differently than Fusion Toaster:
  - With WebRTC and Fusion WebClient, the user cannot set a second call on hold (therefore, there is no need to retrieve it). The user can alternate between calls or reconnect to calls.
  - With Fusion Toaster the user can set second call on hold via SIP and retrieve it via SIP.
- If uaCSTA enabled SIP hard phones (e.g. CP700 SIP) are monitored by an application and the user is involved in a consultation/second call scenario,

second alternate/toggle via CSTA interface does not work. It is possible to alternate/toggle using the phone menu.

---

**NOTICE:** In case of this restriction, consultation must be invoked via uaCSTA or from phone menu.

---

### **CSTA Alternate/Reconnect for non-uaCSTA SIP devices**

If non-uaCSTA devices (e.g. WL3) are monitored by an application and the user is involved in a consultation/second call scenario, alternate/toggle via CSTA interface does not work. It is possible to alternate/toggle via phone menu.

---

**NOTICE:** In case of this restriction, consultation must be invoked from phone menu.

---

## Part 3: Shared-Bridged Appearance CSTA Interface Specification

This part describes the keyset or multiple appearance CSTA interface for OpenScape 4000 switching functions. This specification is an extension of the OpenScape 4000 CSTA Interface Specification.

### 11 Shared-bridged appearances modeling

ECMA CSTA standard 269 Annex A describes Shared-Bridged Appearance model that will be provide by OpenScape 4000. Each CSTA device is represented by its attributes 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.

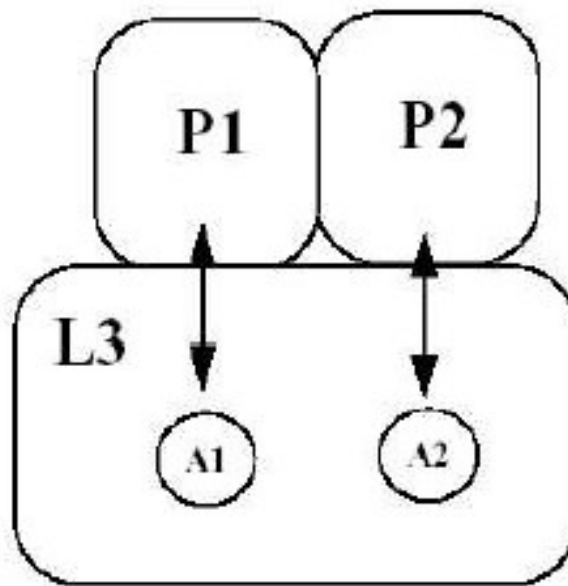
ECMA defines device elements as follows:

Physical element and Logical element: In the case of the OS4K keyset configured there is a logical element which has appearances on the physical elements.

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. These appearances may be configured as primary line and secondary lines.

Changes in the number of addressable appearances for a logical element are reflected by the capabilities exchange services. Please consult OS4K documentation for complete description of keyset configuration.

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. In the switching function the keyset feature is identical to the bridged device configuration. It means that there is one logical element which has appearances on physical elements. The figure below illustrates the bridged device configuration.



**Figure 119: Bridged Device Configuration**

The corresponding keyset setup is the following. P1 is a physical device configured as keyset party. The logical element L3 has the A1 appearance associated with P1. It is called the primary line on P1. P1 can have secondary lines (e.g. logical element L4) as well. On physical device P2 the logical element L3 has the appearance A2. It is called the secondary line on P2.

When a call arrives at the logical element, bridged call appearances are simultaneously selected. 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.
- They end the call.
- They are permanently cleared from the call through some feature/service.
- They move the call away from the device and none of the appearances are connected into the call.

Figure A-6 illustrates this behavior of shared-bridged appearances. In this example, the logical element is L and it has three addressable shared-bridged appearances (A1, A2, and 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.

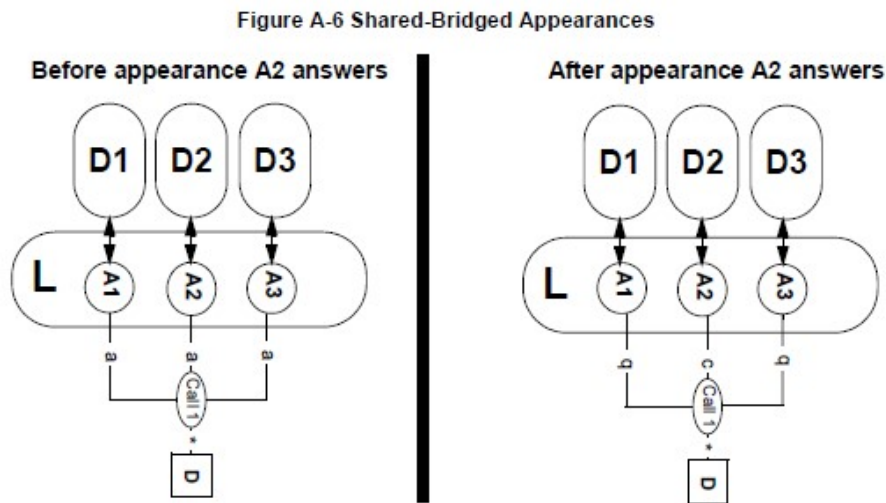
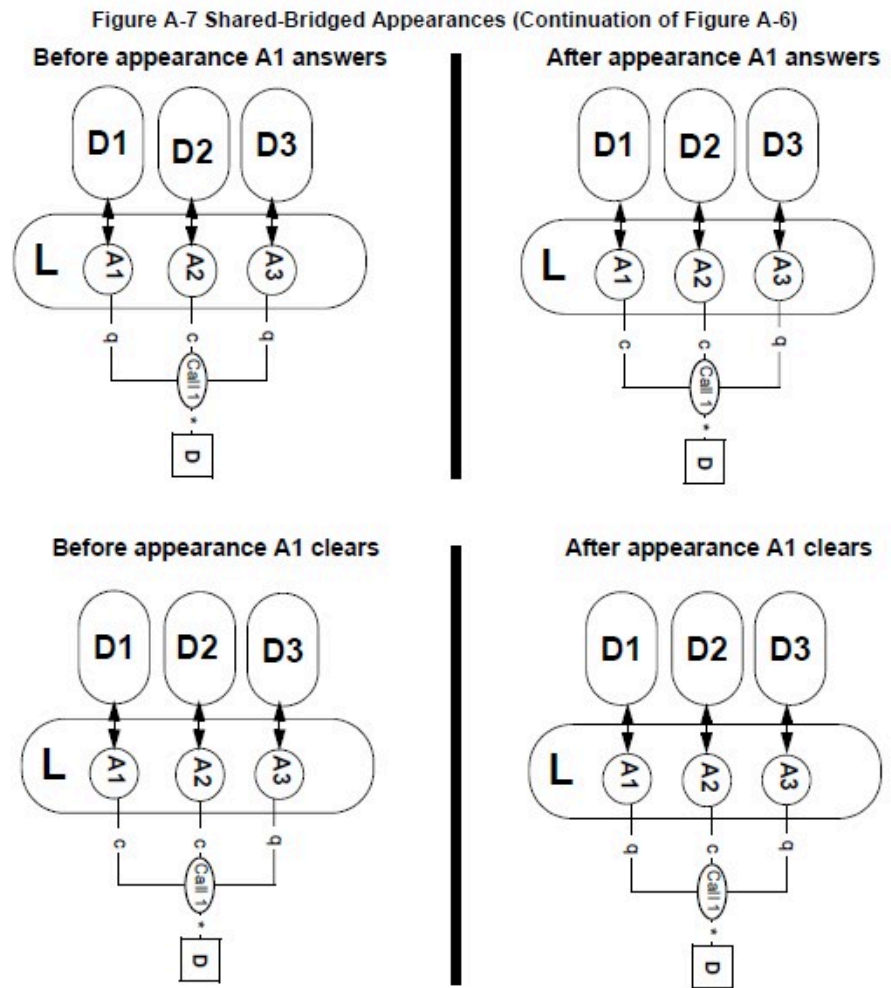


Figure 120: A-6 Shared-Bridged Appearances

The significant behavior 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 -- See Note 1 below.). 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 A-7 illustrates this behavior 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.

**NOTICE:** OS4K will present Conferenced and Bridged Events to report other appearances joining the call and Bridged Events when they go inactive (Bridged/Queued) on the call.



**Figure 121: A-7 Shared-Bridged Appearances (Continuation of Figure A-6)**

When the call is moved away from an appearance or an appearance places the call on hold, the shared-bridged appearance type may become two types:

- 1) **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 A-8 illustrates this behavior 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 A-8 Independent-Shared-Bridged Appearances

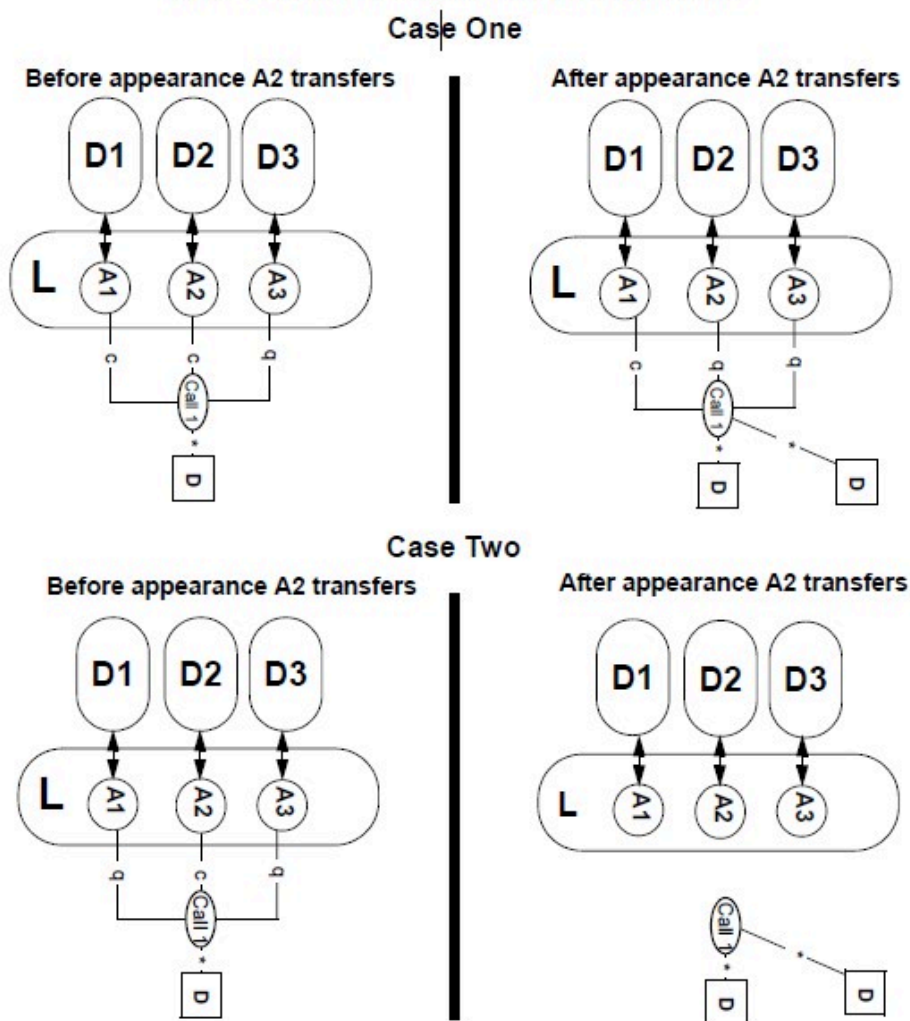


Figure 122: A-8 Independent-Shared-Bridged Appearances

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 A-9 illustrates this behavior 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.

**NOTICE:** OS4K does not support any scenarios where the independent model would be distinguished. Interdependent model will be reported on the OS4K's CSTA interface.

Figure A-9 Independent-Shared-Bridged Appearances (Continuation of Figure A-8)  
Case One

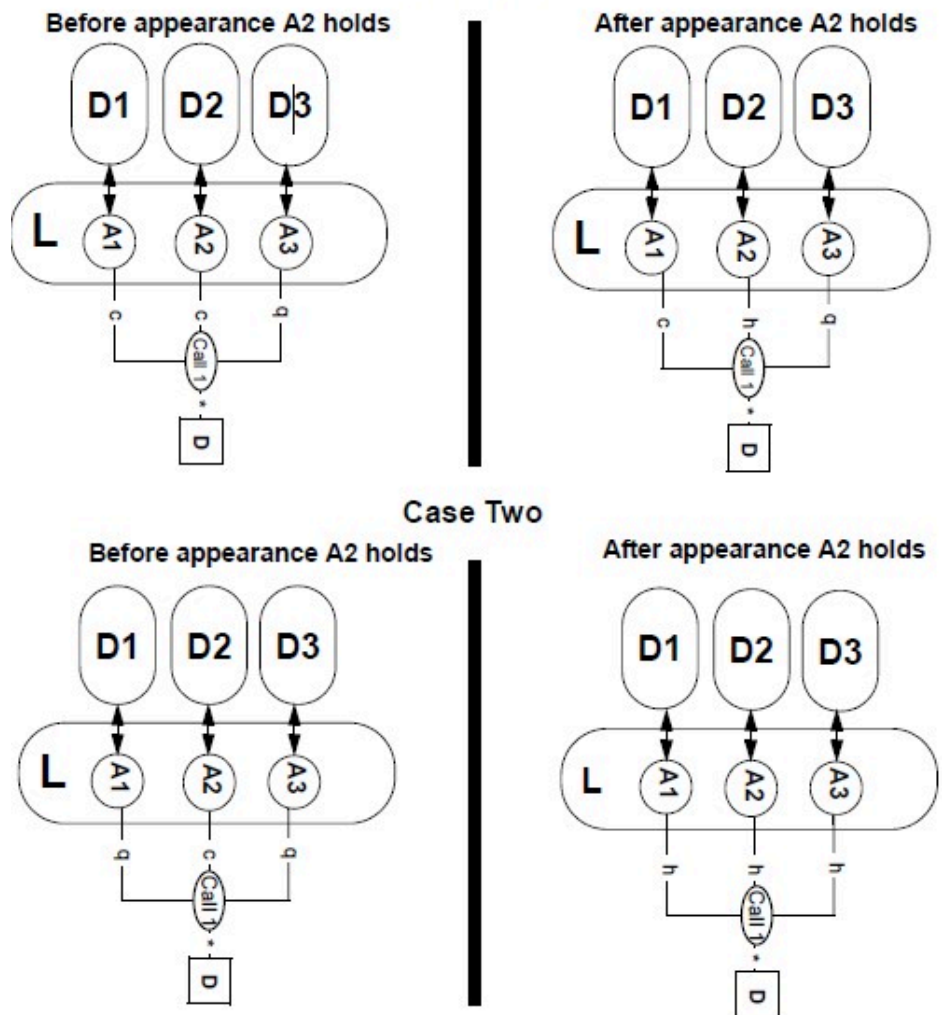


Figure 123: A-9 Independent-Shared-Bridged Appearances (Continuation of Figure A-8)

- 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 A-8 in Case 2 illustrates this behavior 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.

## 11.1 Device Identifier and DeviceID Presentation

OpenScape Voice and OpenScape 4000 Device Identifier and DeviceID presentation for shared-bridge appearances are the same.

The SFR format as defined by ECMA Standard 269 Clause 10.1.2 is used for this purpose and is illustrated below:

Table 364: XML DeviceID Example

SFR Format: N<DN/EXT>
Where: N - Indicates SFR format < - Start/end angle brackets are presented when the SFR NM field is present
DN - Is the Subscriber Directory Number (logical element) / - separator indicating the physical DeviceID will follow EXT - Is the Physical Device added to the logical element to uniquely identify the bridged appearance
> - end DN NM - Subscriber's name and other elements separated by ";" (e.g., display number)

Table 365: XML DeviceID Example - continued

XML DeviceID Example: DN=12200, EXT=12202
<establishedConnection> <callID>7</callID> <deviceID>N&lt;12200/12202&gt;Scoate Fum</deviceID> </establishedConnection>

NOTES:

- 1) When presenting a shared-bridged appearance if the Logical and Physical elements are the same then the DN and EXT part of the deviceID shall be presented.
  - If the event being presented applies to all appearances of the Logical Device then only the DN may presented by the SF. For example, Call Forwarding and Busy cases.
  - OS4K does not provide E164 for bridged appearance. Bridged appearance consists of extension number as configured with SBCSU.

11.2 Supported Phones

All devices can be configured using AMO-SBCSU. This means the following:

- 1) All supported digital system telephones (excluding the ones prepared to be a "secretary" in an Executive-Secretary group or members of an Executive-Secretary group)
  - Supported IP telephones (excluding SIP)
  - CMI devices can be configured as primary lines of a Personal Device group

11.3 Supported CSTA Services

The new shared-bridged appearance capability raises the issue of addressable appearances. 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. There are services where this statement can be applied and there are services where the addressability is not applicable

or not supported by OpenScape 4000 (OS4K). A checkmark (✓) means the service is supported.

The following table shows the usage of addressable/not addressable appearances in CSTA III Services for OpenScape 4000 (OS4K).

The terminology used in the table below have the following meanings:

- **Subscriber Directory Number (DN)** - The fully qualified or extension number assigned in the switching function to identify a subscriber and assign services and features, such as CSTA and shared line appearances.
- **Primary Line** - if only the Subscriber Directory Number (DN) is used to specify the party in the request, the active line appearance on which the service will be performed, must be the primary line, otherwise the service will be refused via a Negative Acknowledge.
- **Physical Device** - if only the Subscriber DN is used to specify the party in the request, the service will be executed on the Physical Device. Physical Device is the device on which the Primary Line is defined for a certain Logical Device. For example, in case of Set the number in the Subscriber DN represents the Physical Device on which the service has to be executed.
- **Logical Device** - if only the Subscriber DN is used to specify the party in the request, the service will be executed on the Logical Device. Logical Device includes all the appearances; which may mean the Primary Line plus the Secondary Lines. For example, a Monitor Start may only be requested with a monitorObject that contains only the Subscriber DN. The device monitoring will then be active for the entire Logical Device, which means events are sent for all associated appearances. A subscriber DN may be assigned as a Phantom Line. A Phantom Line is a Logical Device that does not appear as a primary line on any Physical Device. A Phantom Line that appears on only one Logical Device is referred to as an Exclusive Phantom Line.
- **Line Appearance** - if both the Subscriber DN and the Physical Device are used to specify the party in the request, the service will be executed on behalf of the specified line appearance (Primary, Secondary or Phantom Line). The Subscriber DN will represent the Logical Device and the PhysicalDevice IE will represent the Physical Device. This is the classical addressable appearance.

**Table 366: Supported CSTA Services**

CSTA Service Request	OS4K		Subscriber DN	Subscriber DN/Physical Device EXT
	ASN1	XML		
CAPABILITIES EXCHANGE SERVICES				
get logical device Information	#		Logical Device	Rejected
get physical device information	#		Physical Device	Rejected
MONITORING SERVICES				
change monitor filter	#		Logical Device	Rejected
monitor start	#	#	Logical Device	Rejected
monitor stop	#	#	Logical Device	Rejected
SNAPSHOT SERVICES				

## Shared-bridged appearances modeling

CSTA Service Request	OS4K		Subscriber DN	Subscriber DN/Physical Device EXT
	ASN1	XML		
<b>CAPABILITIES EXCHANGE SERVICES</b>				
snapshot call	#	#	Primary Line	Rejected
snapshot device	#	#	Logical Device	Rejected
<b>CALL CONTROL SERVICES</b>				
accept call	#	#	Primary Line	Rejected
alternate call	#	#	Primary Line	Line Appearance
answer call (2)	#	#	Primary Line	Line Appearance
call back call-related	#	#	Primary Line	Line Appearance
call back message call-related				
camp on call				
clear connection	#	#	Primary Line	Line Appearance
conference call	#	#	Primary Line	Line Appearance
consultation call	#	#	Logical Device	Rejected
connected-timeslot-request	#	#	Primary Line	Line Appearance
deflect call	#	#	Logical Device	Rejected
dial digits	#	#	Primary Line	Line Appearance
directed pickup call			Primary Line	Line Appearance
group pickup call			Primary Line	Line Appearance
hold call	#	#	Primary Line	Line Appearance
intrude call			Primary Line	Line Appearance
join call			Primary Line	Rejected
make call	#	#	Primary Line	Line Appearance
make predictive call	#	#	Primary Line	Rejected
park call				
reconnect call	#	#	Primary Line	Line Appearance
reconnect-timeslot-request	#	#	Primary Line	Line Appearance
retrieve call	#	#	Primary Line	Line Appearance
send user information	#		Not supported	Not supported
single step transfer	#	#	Logical Device	Rejected

CSTA Service Request	OS4K		Subscriber DN	Subscriber DN/Physical Device EXT
	ASN1	XML		
<b>CAPABILITIES EXCHANGE SERVICES</b>				
single step conference			Primary Line	Line Appearance
transfer call	#	#	Primary Line	Line Appearance
<b>CALL ASSOCIATED FEATURES</b>				
associate data			Not applicable	Not applicable
change connection information			Primary Line	Line Appearance
generate digits	#	#	Primary Line	Line Appearance
generate telephony tones	#	#	Physical Device	Rejected
<b>ROUTING SERVICES</b>	(1)			
<b>Registration Services</b>				
route register			Logical Device	Rejected
route register abort			Logical Device	Rejected
route register cancel			Logical Device	Rejected
<b>Services</b>				
reroute			Logical Device	Rejected
route end			Logical Device	Rejected
route reject			Logical Device	Rejected
route request			Logical Device	Rejected
route select			Logical Device	Rejected
<b>PHYSICAL DEVICE FEATURES</b>				
get microphone mute	#	#	Physical Device	Line Appearance
get speaker volume			Physical Device	Rejected
get message waiting indicator	#		Physical Device	Rejected
set message waiting indicator	#	#	Physical Device	Rejected
set microphone mute	#	#	Physical Device	Line Appearance
set speaker volume			Physical Device	Rejected
<b>LOGICAL DEVICE FEATURES</b>				
call back non-call-related			Logical Device	Line Appearance
cancel call back	#	#	Logical Device	Line Appearance

## Shared-bridged appearances modeling

CSTA Service Request	OS4K		Subscriber DN	Subscriber DN/Physical Device EXT
	ASN1	XML		
CAPABILITIES EXCHANGE SERVICES				
get agent state	#	#	Logical Device	Rejected
get do not disturb	#	#	Logical Device	Rejected
get forwarding	#	#	Logical Device	Rejected
set agent state	#	#	Logical Device	Rejected
set do not disturb	#	#	Logical Device	Rejected
set forwarding	#	#	Logical Device	Rejected
I/O SERVICES				
Registration Services				
I/O register (C#S)	#	#	Physical Device	Rejected
I/O register abort (S#C)			Physical Device	Rejected
I/O register cancel (C#S)	#	#	Physical Device	Rejected
I/O Services				
data path resumed (S#C)	#	#	Physical Device	Rejected
data path suspended (S#C)	#	#	Physical Device	Rejected
fast data(C#S)	#	#	Physical Device	Rejected
fast data(S#C)	#	#	Physical Device	Rejected
resume data path (C#S)			Physical Device	Rejected
send broadcast data (C#S)			Physical Device	Rejected
send data (C#S)	#	#	Physical Device	Rejected
send data (S#C)	#	#	Physical Device	Rejected
send multicast data (C#S)			Physical Device	Rejected
start data path (S#C)	#		Physical Device	Rejected
start data path (C#S)	#	#	Physical Device	Rejected
stop data path (S#C)	#	#	Physical Device	Rejected
stop data path (C#S)		#	Physical Device	Rejected
suspend data path (S#C)			Physical Device	Rejected
suspend data path (C#S)			Physical Device	Rejected

(1) Routing services are support by OpenScape 4000 only for RCGs. Keyset operation is a feature for extensions (AMO-SBCSU).

(2) For answer call to work "hands-free" the SIP phone's auto-answer feature must be enabled.

## 11.4 Feature Activation

To activate the feature of OpenScape 4000, set APPEARANCE\_LIST\_REQUIRED flag to 1/YES in Connectivity Adaptor Advanced Configuration.

## 11.5 CALL SCENARIOS

### 11.5.1 Logical Device Monitoring

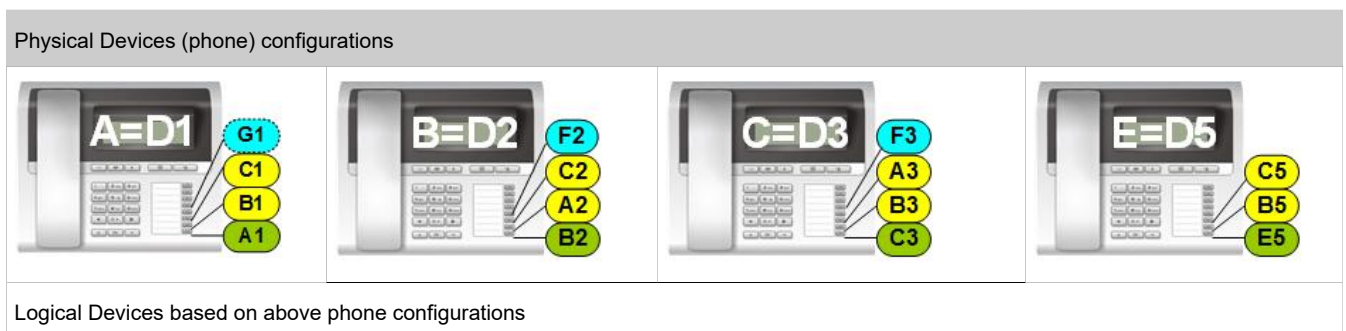
To observe CSTA events related to shared-bridge appearances the Computing Function shall invoke the Monitor Start service on the Logical Device elements. The resulting event flow provides the Computing Function with enough information on the elements to track and control calls and other capabilities.

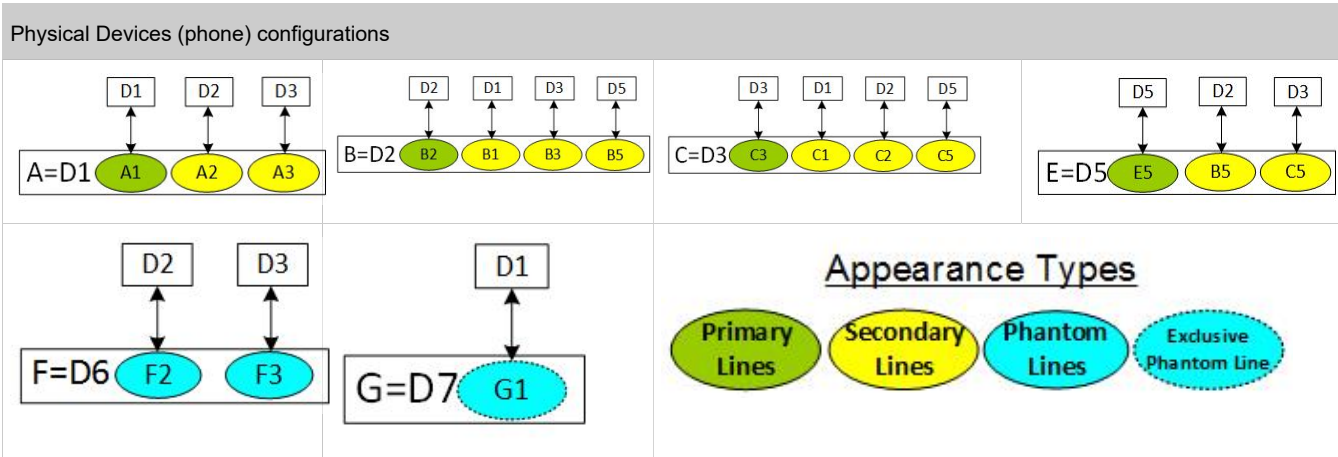
### 11.5.2 Discovery Services

The Get Logical Device Information and Get Physical Device Information services are used by the Computing Function to discover the shared appearance configuration. The discover examples provided in this section show 4 phones configured as keysets. D1, D2, D3 and D5 tags are used to identify the Physical Devices. In the information below the color coded highlighting identifies Primary Lines, Secondary Lines, and Phantom Lines.

**NOTICE:** The logical and physical configuration examples shown in Figure 128 below are used throughout this specification chapter for the CSTA service and event flow examples.

#### Logical and Physical Configurations





11.5.2.1 Example Configuration

The table below illustrates the relationship of Logical to Physical to Subscriber DN for the above configuration. Configuration in OS4K is realized by means of SBCSU, TAPRO and KCSU AMOs.

Table 367: Relationship of Logical to Physical to Subscriber DN

Logical Device	Physical Device	Subscriber DN
A	D1	31001
B	D2	31002
C	D3	31003
E	D5	31005
F	--	31006
G	--	31007

**NOTICE:** Phantom lines are Logical Device numbers and are not assigned as the Prime Line on any phone.

**NOTICE:** It is also possible to configure phones with one or more private lines. Private lines appear on only one physical phone, they are not shared by other physical devices. When subscribers DNs for these lines are monitored then a standard non-appearance CSTA event flow shall be presented.

11.5.2.2 Get Logical Device Information

The device parameter in the Get Logical Device Information request must only be the Logical Device identifier (i.e., Subscriber DN).

Based on the physical device configuration shown in the previous section the following parameters shall be presented in the Get Logical Device Response:

- deviceCategory = station
- nameDeviceType = station
- hasPhysicalElement = see table below

**Table 368: Logical Device Information**

Logical Device	value
31001	true
31002	true
31003	true
31005	true
31006	false
31007	false

- appearanceAddressable = true
- appearanceType = OS4K: interdependentSharedBridged
- appearanceList - not applicable for shared-bridged appearances.
- otherPhysicalDeviceList

**Table 369: Other Physical Device List**

Logical Device	deviceID
31001	31002
	31003
31002	31001
	31003
	31005
31003	31001
	31002
	31005
31005	31002
	31003
31006	31002
	31003
31007	31001

### 11.5.2.3 Get Physical Device Information

The device parameter in the Get Physical Device Information request must only be the Logical Device identifier (i.e., Subscriber DN).

The following parameters are presented in the Get Physical Device Response:

- **deviceCategory** = station
- **namedDeviceTypes** = station
- **deviceModelName** = (as is, e.g. **OpenStage 40**)
- **hasLogicalElements** = true
- **otherLogicalDeviceList** - see table below

**Table 370: Other Logical Device List**

Device	deviceID
31001	31002
	31003
31002	31001
	31003
	31005
31003	31001
	31002
	31005
31005	31002
	31003

If the specific device in the Get Physical Device Information request is a Phantom Line then the response is presented as follows:

- **deviceCategory** = station
- **namedDeviceTypes** = station
- **hasLogicalElements** = false
- **otherLogicalDeviceList** - empty
- **deviceModelName** = Phantom Device

## 11.5.3 Snapshot Services

### 11.5.3.1 Snapshot Device

The snapshotObject in the Snapshot Device service request is the Logical Device == Subscriber DN.

The following discovery services assume the follow connection status.

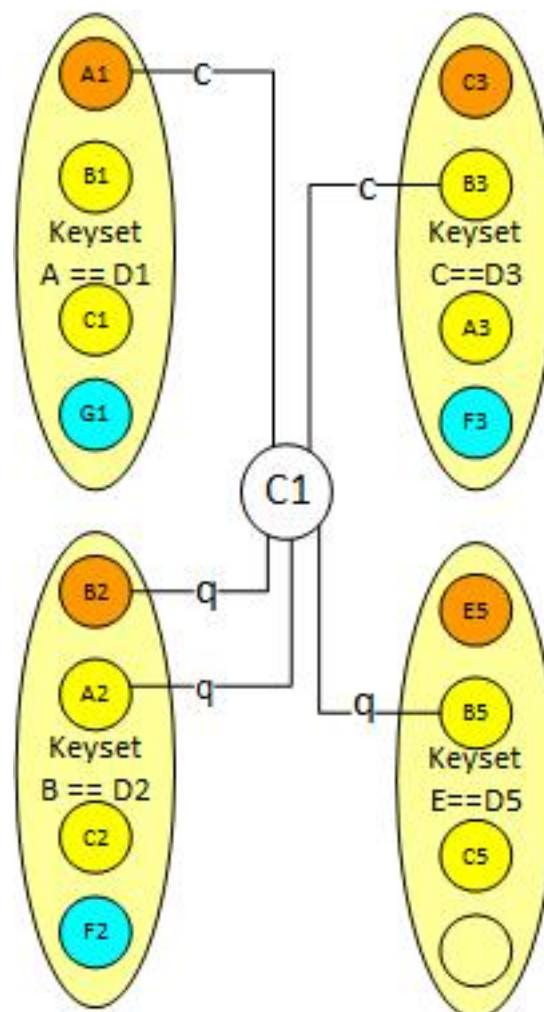
A1 has called B and B3 has answered the call as illustrated in Figure 129. To determine the status of other appearances of A and B that are not connected the following rules shall be applied:

**Shared-Bridged appearances of A and B enter the Bridged/Queued state under the following conditions:**

- The appearance of line A or B is not already connected at the physical device.
- The appearance of line A or B is connected but neither is connected at the host device.

Using the example illustration below of A and B connected have appearances on that are able to bridge on the current connection under the rules listed above.

- A2 is able to bridge
- B2 and B5 are able to bridge
- B1 cannot bridge since physical device A already has a connection to an appearance of B.
- A3 cannot bridge since physical device C already has a connection to an appearance of A.



**Figure 124: Example Shared-Bridged Appearance Call**

Case 1: The following described the Snapshot Device response data presented for Logical Devices A and B based on the example call topology. The servicesPermitted are as determined by the switching function based on the current state.

### 1) 1. SnapshotDevice; snapshotObject = A

#### SnapshotDeviceResponse (impacts)

- connectionIdentifier = A/D1 C1
- endpoint = A/D1
- localCallState
  - o) compoundCallState
    - localConnectionState = connected (localConnectionInfo for A/D1)
    - localConnectionState = connected (localConnectionInfo for B/D3)
- connectionIdentifier = A/D2 C1
- endpoint = A/D2
- localCallState
  - o) compoundCallState
    - localConnectionState = queued (localConnectionState for A/D2)
    - localConnectionState = connected (localConnectionState for B/D3)

### 2) SnapshotDevice; snapshotObject = B

#### SnapshotDeviceResponse (impacts)

- connectionIdentifier = B/D2 C1
- endpoint = B/D2
- localCallState
  - o) compoundCallState
    - localConnectionState = queued (localConnectionState for B/D2)
    - localConnectionState = connected (localConnectionState for A/D1)
- connectionIdentifier = B/D3 C1
- endpoint = B/D3
- localCallState
  - o) compoundCallState
    - localConnectionState = connected (localConnectionState for B/D3)
    - localConnectionState = connected (localConnectionState for A/D1)
- connectionIdentifier = B/D5 C1
- endpoint = B/D5
- localCallState
  - o) compoundCallState
    - localConnectionState = queued (localConnectionState for B/D5)
    - localConnectionState = connected (localConnectionState for A/D1)

Case 2: The following illustrate the Snapshot Device response data presented for Logical Devices A and B, were A2 is bridged on the call. The servicesPermitted and mediaServiceInfoList data are as determined by the switching function based on the current state of the

calls. NOTE: That B2 connection longer in the queued state since A2 is now connected with B3 on the host device.

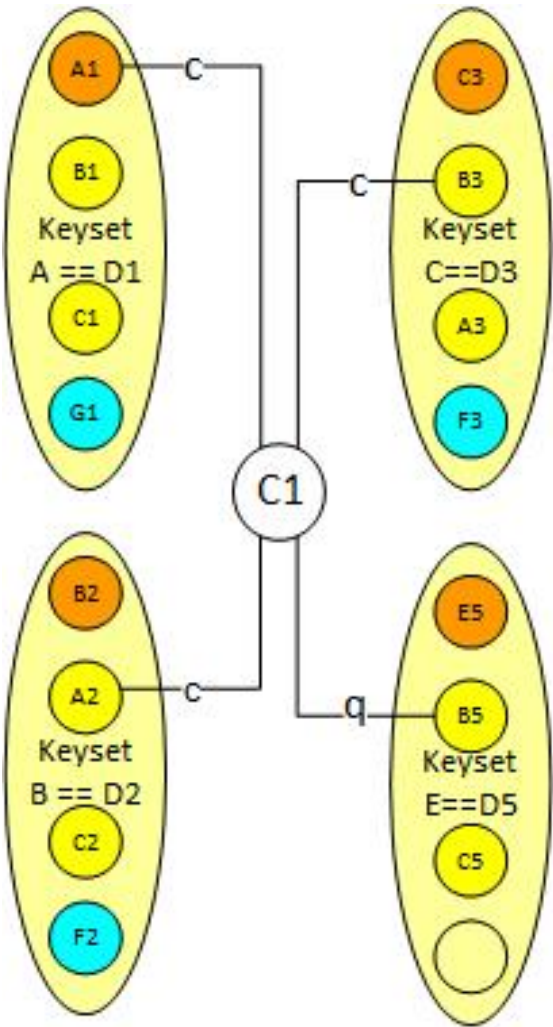


Figure 125: Example A-B bridged call topology for SBA connections

### 3) SnapshotDevice; snapshotObject = A

#### SnapshotDeviceResponse (impacts)

- connectionIdentifier = A/D1 C1
- endpoint = A/D1
- localCallState
  - o) compoundCallState
    - localConnectionState = connected (localConnectionInfo for A/D1)
    - localConnectionState = connected (localConnectionInfo for B/D3)
    - localConnectionState = connected (localConnectionInfo for A/D2)
- connectionIdentifier = A/D2 C1
- endpoint = A/D2
- localCallState
  - o) compoundCallState
    - *localConnectionState = connected (localConnectionState for A/D2)*
    - *localConnectionState = connected (localConnectionState for B/D3)*
    - *localConnectionState = connected (localConnectionState for A/D1)*

### 4) SnapshotDevice; snapshotObject = B

#### SnapshotDeviceResponse (impacts)

- connectionIdentifier = B/D3 C1
- endpoint = B/D3
- localCallState
  - o) compoundCallState
    - localConnectionState = connected (localConnectionState for B/D3)
    - localConnectionState = connected (*localConnectionState for A/D1*)
    - localConnectionState = connected (localConnectionState for A/D2)
- connectionIdentifier = B/D5 C1
- endpoint = B/D5
- localCallState
  - o) compoundCallState
    - localConnectionState = queued (localConnectionState for B/D5)
    - localConnectionState = connected (*localConnectionState for A/D1*)
    - localConnectionState = connected (localConnectionState for A/D2)

## 11.5.3.2 Snapshot Call

The following table illustrates the Snapshot Call responses for each call presented each of the previous Snapshot Device requests for CASE 1. Service Permitted is as presented by the SF.

The following table illustrates the Snapshot Call Response results. The snapshotObject may be A C1 or B C1 or any connection identifier presented in the Snapshot Device Response shown in the previous section (e.g., A/D2 C1, B/D5 C1, etc.).

**Table 371: Snapshot Call Response results for Case 1**

deviceOnCall	callIdentifier	localConnectionInfo
A/D1	C1	connected
A/D2	C1	queued
B/D2	C1	queued
B/D3	C1	connected
B/D5	C1	queued

The following table illustrates the Snapshot Call responses for each call presented each of the previous Snapshot Device requests for CASE 2. Service Permitted and mediaServiceInfoList values are as presented by the SF.

The following table illustrates the Snapshot Call Response results. The snapshotObject may be A C1 or B C1 or any connection identifier presented in the Snapshot Device Response shown in the previous section (e.g., A/D2 C1, B/D5 C1, etc.).

deviceOnCall	callIdentifier	localConnectionInfo
A/D1	C1	connected
A/D2	C1	connected
B/D3	C1	connected
B/D5	C1	queued

**Table 372: Snapshot Call Response results for Case 2**

## 11.5.4 Call Origination

---

**NOTICE:** The Phone and Logical Device Model are the same as described in section 10.5.2 for all call flow examples.

---



---

**NOTICE: Shared-Bridged connections in the Bridged/Queued state only permit the CSTA Answer Call service for the purpose of Bridging onto the call. The CSTA Clear Connection service is not supported in the Bridged/Queued connection state.**

---

### 11.5.4.1 Basic Call answered on secondary line appearance

- A calls B
- B2, B3 and B5 are ringing (i.e., B2= D2/D2, B3=D2/D3, B5=D2/D5.)
- B3 answers call from A1
- A1 and B3 are connected

Shared-bridged appearances modeling

- A2, B2 and B5 are able to bridge onto this call per rules described in [Sect. 10.5.3.1](#)

Table 373: Basic call answered on secondary line appearance

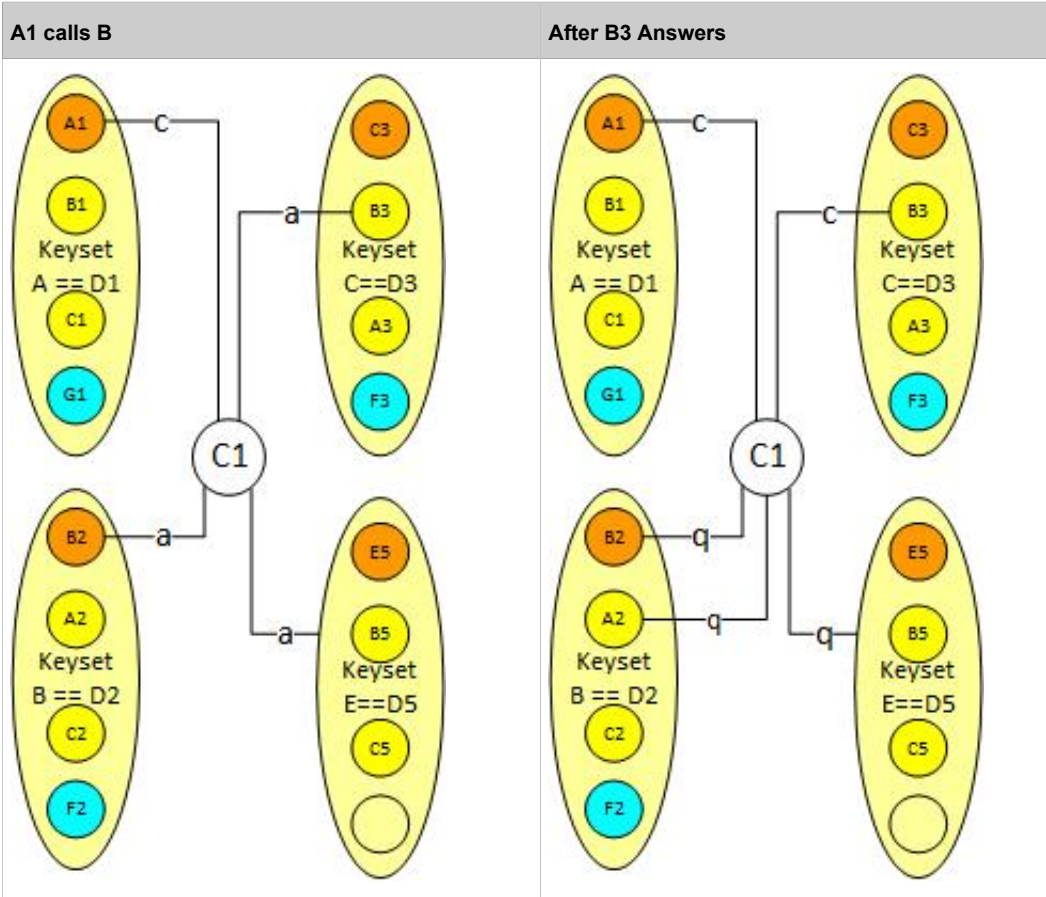


Table 374: Basic call answered on secondary line appearance

Activity	Monitored Device A == D1		Monitored Device B == D2	
1. Application invokes call from A to B via CSTA MakeCall.  Note: The callingDevice identifier may also be presented as A/D1(i.e., D1/D1)	MakeCall	A/D1		
	callingDevice	B		
	calledDirectoryNumber	A/D1 C1		
	MakeCallResponse			
	callingDevice			
	ServiceInitiated	A/D1 C1		
	Connection	A/D1		
	initiatingDevice	Initiated		
	cause	normal		
	localConnectionInfo			

Activity	Monitored Device A == D1		Monitored Device B == D2	
	Digits Dialed	A/D1 C1		
	diallingConnection	A/D1		
	diallingDevice	B		
	diallingSequence	Initiated		
	localConnectionInfo	normal		
	cause			
	Originated	A/D1 C1		
	connection	A/D1		
	callingDevice	B		
	calledDevice	connected		
	localConnectionInfo	normal		
	cause			
2. Appearances B2, B3 and B5 are ringing. NOTE: Other appearances of A do not receive Delivered Event since no action is possible by these appearances until the call is answered by B later in the example flow.	Delivered	B/D2 C1	Delivered	B/D2 C1
	connection	B/D2	connection	B/D2
	alertingDevice	A/D1	alertingDevice	A/D1
	callingDevice	B	callingDevice	B
	calledDevice	notSpecified	calledDevice	notSpecified
	lastRedirectionDevice	normal	lastRedirectionDevice	normal
	cause	connected	cause	alerting
	localConnectionInfo		localConnectionInfo	
	Delivered	B/D3 C1	Delivered	B/D3 C1
	connection	B/D3	connection	B/D3
	alertingDevice	A/D1	alertingDevice	A/D1
	callingDevice	B	callingDevice	B
	calledDevice	notSpecified	calledDevice	notSpecified
	lastRedirectionDevice	normal	lastRedirectionDevice	normal
	cause	connected	cause	alerting
	localConnectionInfo		localConnectionInfo	

## Shared-bridged appearances modeling

Activity	Monitored Device A == D1		Monitored Device B == D2	
	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	B/D5 C1 B/D5 A/D1 B notSpecified normal connected	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	B/D5 C1 B/D5 A/D1 B notSpecified normal alerting
3. Application answers call from A1 on behalf B3			Answer Call callToBeAnswered Answer Call Response	B/D3 C1
4. A1 and B3 are connected  NOTE: Established and Bridged Events on device B monitor may be presented before the Established and Bridged Events on device A monitor.	Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	B/D3 C1 B/D3 A/D1 B notSpecified normal connected	Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	B/D3 C1 B/D3 A/D1 B notSpecified normal connected
5. A2, B2 and B5 are in Bridged/Queued state	BridgedEvent connection bridedAppearance cause localConnectionInfo	B/D2 C1 B/D2 normal connected	BridgedEvent connection bridedAppearance cause localConnectionInfo	B/D2 C1 B/D2 normal queued
	BridgedEvent connection bridedAppearance cause localConnectionInfo	B/D5 C1 B/D5 normal connected	BridgedEvent connection bridedAppearance cause localConnectionInfo	B/D5 C1 B/D5 normal queued
	BridgedEvent connection bridedAppearance cause localConnectionInfo	A/D2 C1 A/D2 normal queued	BridgedEvent connection bridedAppearance cause localConnectionInfo	A/D2 C1 A/D2 normal connected

11.5.4.2 Basic Call to Busy Logical/Physical Device (no call waiting support)

- X calls B
- All Physical/Logical Device B is already busy on the call.
- All Physical/Logical device B is not configured for call waiting.
- Call from X will fail.

Table 375: Basic call to Busy Device

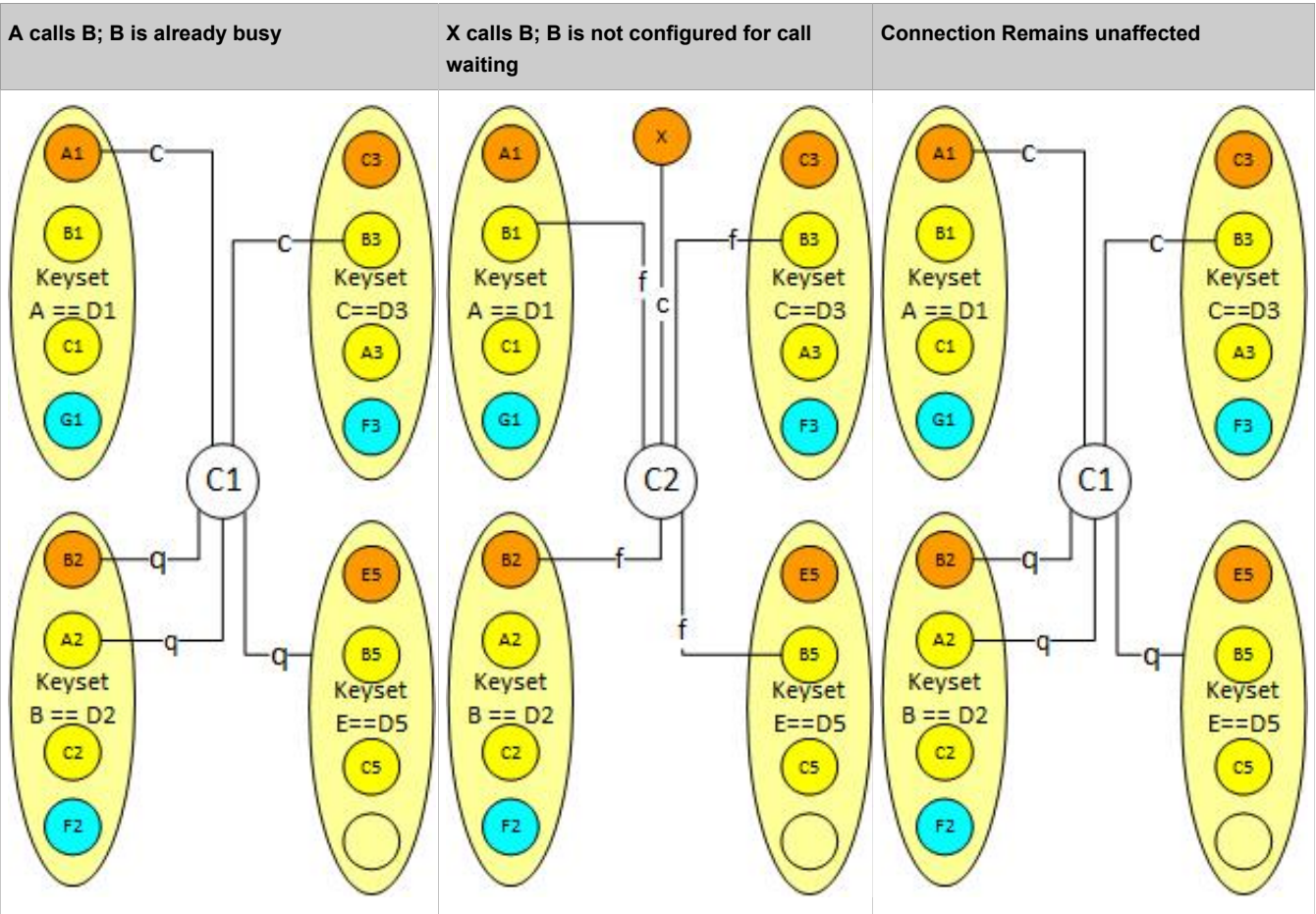


Table 376: Basic call to Busy Device

Activity	Monitored Device X		Monitored Device B	
1. X calls B	ServiceInitiated	X C2		
	Connection	X		
	initiatingDevice	normal		
	cause	initiated		
	localConnectionInfo			

## Shared-bridged appearances modeling

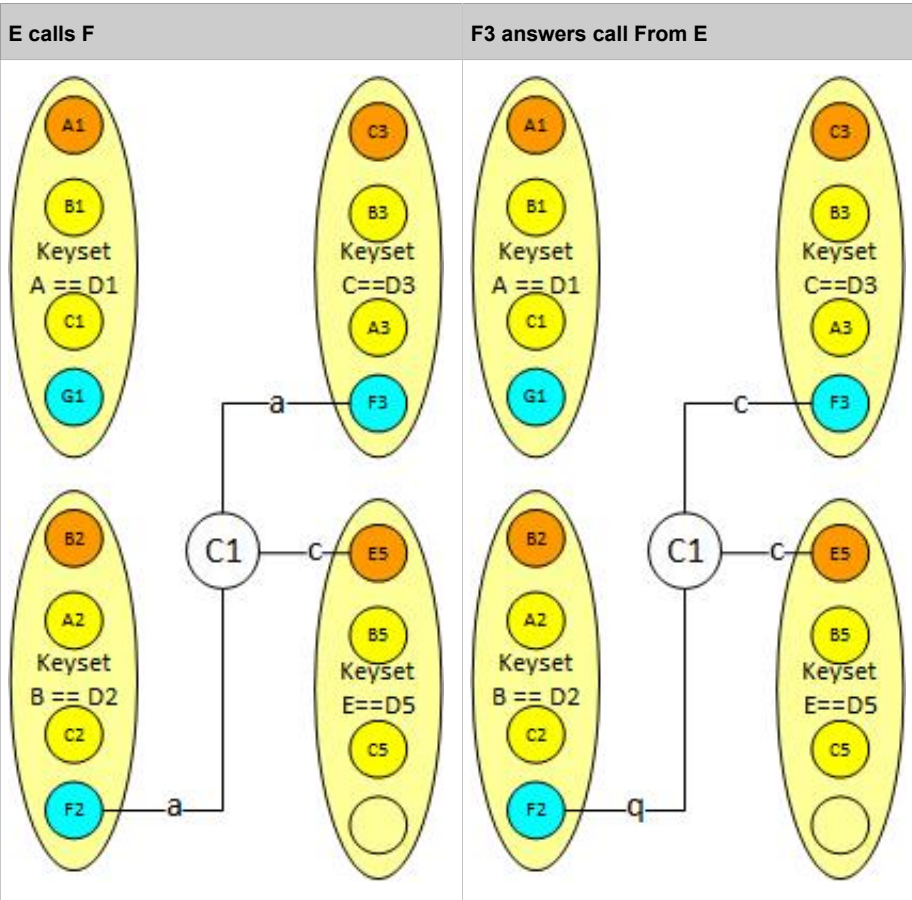
Activity	Monitored Device X		Monitored Device B	
	Digits Dialed	X C2		
	diallingConnection	X		
	diallingDevice	B		
	diallingSequence	initiated		
	localConnectionInfo	normal		
	cause			
	Originated	X C2		
	connection	X		
	callingDevice	B		
	calledDevice	connected		
	localConnectionInfo	normal		
	cause			
2. B is busy	Failed	B C2	Failed	B C2
	connection	B*	connection	B*
	failingDevice	X	failingDevice	X
	callingDevice	B	callingDevice	B
	calledDevice	busy	calledDevice	busy
	cause	connected	cause	fail
	localConnectionInfo		localConnectionInfo	
	ConnectionCleared	B C2**	ConnectionCleared	B C2
	connection	B**	connection	B**
	releasingDevice	normalClearing	releasingDevice	normalClearing
	cause	connected	cause	null
	localConnectionInfo		localConnectionInfo	
3. optional: timer expires:	Failed	X C2		
	connection	X		
	failingDevice	X		
	callingDevice	B		
	calledDevice	blocked		
	cause	fail		
	localConnectionInfo			

Activity	Monitored Device X		Monitored Device B	
4. X goes idle (on hook or because of the timer)	ConnectionCleared connection releasingDevice cause localConnectionInfo	X C2** X normalClearing null		
* The failedConnection is "B C2" and the failedDevice is "B" because the event applies to all appearances of B. The alternative would be to send separate Failed events for each appearance.				
** The droppedConnection is "B C2" and the releasingDevice is "B" because the event applies to all appearances of B. The alternative would be to send separate Connection Cleared events for each appearance.				

11.5.4.3 Basic Call - To Phantom Line

- E calls F
- F3 answers
- All other shared-bridged appearance connections for F are in the Bridged/Queued state and may be bridged onto via the CSTA Answer Call service.

Table 377: Basic Call to Phantom Line



**Table 378: Basic Call to Phantom Line**

Activity	Monitored Device E == D5		Monitored Device F	
1. Application invokes call from E to F via CSTA MakeCall.  Note: The callingDevice identifier may also be presented as A/D1(i.e., D1/D1)	MakeCall callingDevice calledDirectoryNumber MakeCallResponse callingDevice	E F E C1		
2. E calls F	ServiceInitiated connection initiatingDevice cause localConnectionInfo	E/D5 C1 E/D5 newCall intiated		
	Digits Dialed diallingConnection diallingDevice diallingSequence localConnectionInfo cause	E/D5 C1 E/D5 F Initiated normal		
	Originated connection callingDevice calledDevice localConnectionInfo cause	E/D5 C1 E/D5 F connected normal		
3. F2 and F3 are ringing	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	F/D2 C1 F/D2 E/D5 F notSpecified normal connected	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	F/D2 C1 F/D2 E/D5 F notSpecified normal alerting

Activity	Monitored Device E == D5		Monitored Device F	
	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	F/D3 C1 F/D3 E/D5 F notSpecified normal connected	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	F/D3 C1 F/D3 E/D5 F notSpecified normal alerting
4. F3 answers	Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	F/D3 C1 F/D3 E/D5 F notSpecified normal connected	Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	F/D3 C1 F/D3 E/D5 F notSpecified normal connected
5. F2 is queued	BridgedEvent connection bridedAppearance cause localConnectionInfo	F/D2 C1 F/D2 normal connected	BridgedEvent connection bridedAppearance cause localConnectionInfo	F/D2 C1 F/D2 normal queued

## 11.5.5 Call Completion Services

### 11.5.5.1 Call Back - Call Related

- A2 and B2 are currently INUSE on Logical/Physical Device B (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- User of Keyset B invokes a call setup from C2 to E
- E is currently busy so call fails.
- Application invokes callback request on behalf of C2 toward E
- Once E and A2 and B2 go idle callback is initiated by the switching function on behalf of B2 to E. NOTE: The SF always initiates Call Back Recall from the primary line of the device where the Call Back service request was invoked.

Table 379: Call Back

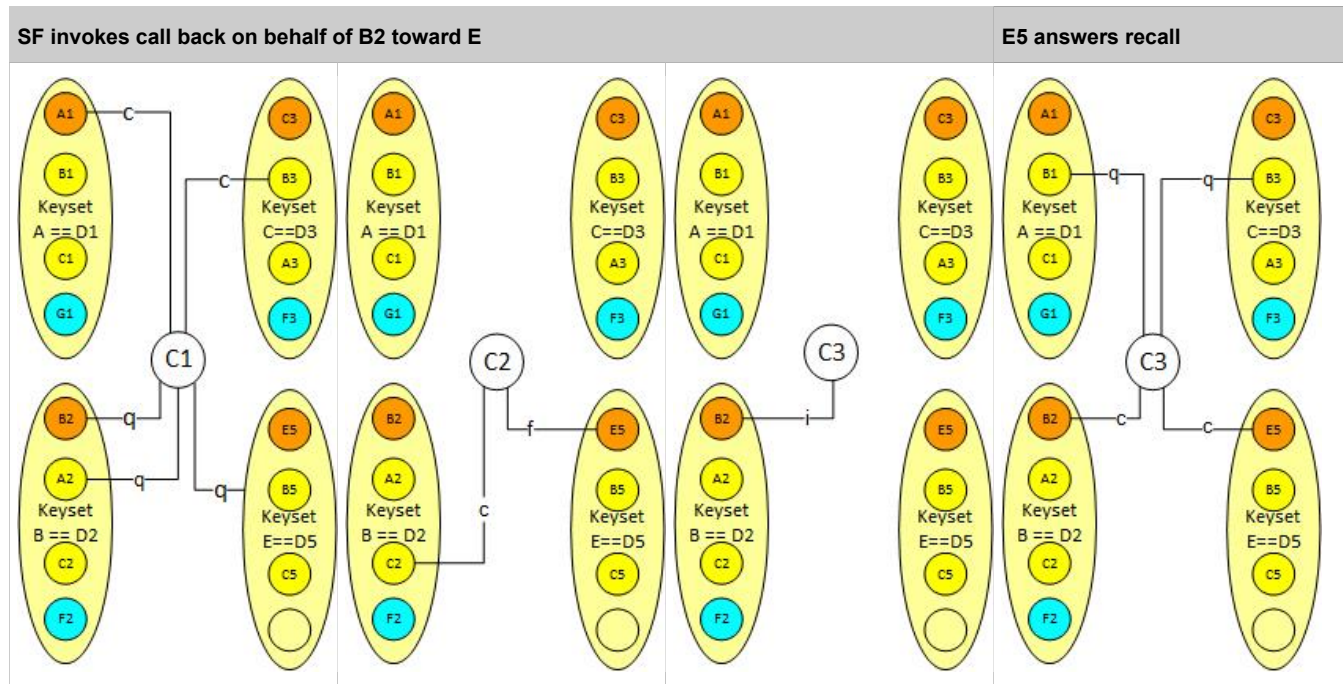


Table 380: Call Back

Activity	Monitored Device C == D3		Monitored Device E == D5	
1. E/D5 is currently busy call back is initiated for connection C/D2 C2	Call Back Request	C/D2 C2		
	callbackConnection	E		
	Call Back Response			
	targetDevice			
	Failed	C/D2 C2		
	connection	C/D2		
	failingDevice	C/D2		
	callingDevice	E		
	calledDevice	blocked		
	cause	fail		
	localConnectionInfo			
	CallbackEvent	B		
	originatingDevice	E		
	targetDevice	true		
	callbackSetCanceled			

2. C2 goes idle	ConnectionCleared droppedConnection releasingDevice localConnectionInfo cause	C/D2 C2 C/D2 null normalClearing		
B2 A2 and E5 go idle through normal clearing flow				

**Table 381: Call Back**

Activity	Monitored Device B == D2		Monitored Device E == D5	
3. SF initiates Callback Recall	Service Initiated initiatedConnection initiatingDevice localConnectionInfo cause	B/D2 C3 B/D2 initiated CallBack		
4. E is reserved for the callback			Failed failedConnection failingDevice callingDevice calledDevice localConnectionInfo cause	E/D5 C3 E/D5 B/D2 E fail blocked
OS4K clears D5 blocking.			ConnectionCleared connection releasingDevice cause localConnectionInfo	E C3 E normalClearing null
B2 answers and the standard call setup flow from B2 to D5 occurs	Originated connection callingDevice calledDevice localConnectionInfo cause	B/D2 C3 B/D2 E connected CallBack		

## Shared-bridged appearances modeling

5. E5 is ringing	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C3 E/D5 B/D2 E notSpecified CallBack connected	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C3 E/D5 B/D2 E notSpecified CallBack alerting
6. call is built up via normal steps				

### 11.5.6 Bridged Call Flows

When the Answer Call service is invoked on a Bridged/Queued connectionID a Conferenced Event is raised reusing the same callID. Subsequent Bridged and Conferenced events are raised as shared call appearance become active or passive on the bridged call. The Conference modeling of Bridged calls shall extend the use of keyOperation cause code to differentiate bridged-call from basic Conference calls.

**Note: in OS4K a bridge may consist of 3 parties only**

#### 11.5.6.1 Bridging onto a Bridged/Queued connection using the Answer Call Service

- A1 and B3 are talking (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- B2 bridges onto the connection with A1 and B3
- A2 is not longer in Bridged/Queued state per rules described in [Sect. 10.5.3.1](#)

Table 382: Bridging using the Answer Call Service

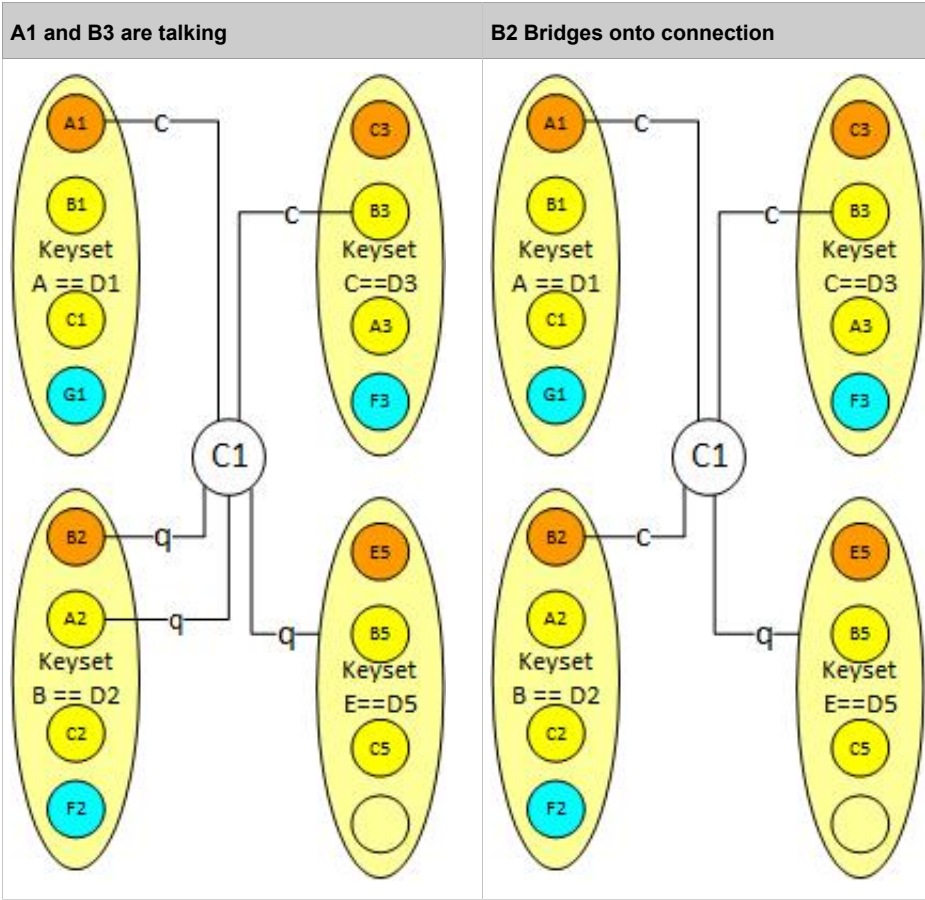


Table 383: Bridging using the Answer Call Service

Activity	Monitored Device A == D2		Monitored Device B == D2	
1. B2 answers bridged call.  NOTE: Refer to section 10.5.4.1 for the setup event flow)			Answer Call callToBeAnswered AnswerCallResponse	B/D2 C1

## Shared-bridged appearances modeling

2. A1, B2 and B3 are connected  E5 remains unaffected in BridgedQ/ueued state section 10.5.4.1	Conferenced	A C1 *	Conferenced	B C1*
	primaryOldCall	-	primaryOldCall	-
	secondaryOldCall	B/D2	secondaryOldCall	B/D2
	conferencingDevice	B/D2	conferencingDevice	B/D2
	addedParty	A/D1 C1	addedParty	A/D1 C1
	connectionlist	B/D2 C1	connectionlist	B/D2 C1
	newConnection	B/D3 C1	newConnection	B/D3 C1
	connectionlist	connected	connectionlist	connected
	newConnection	keyOperation	newConnection	keyOperation
	connectionlist		connectionlist	
	newConnection		newConnection	
	localConnectionInfo		localConnectionInfo	
	cause		cause	

### 11.5.6.2 Available Line Selection via CSTA

- A2 and B2 are currently INUSE on Logical/Physical Device B (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- User of Keyset B wishes to setup a call via CSTA-enabled application to E
- The application sees that C2 and F2 are available on Keyset B and selects C2 to setup the call to E
- E5 answers call from C2.
- C1 and C3 are placed into the Bridged/Queued state
- The first call between A1 and B3 is unaffected.

Table 384: Available Line Selection

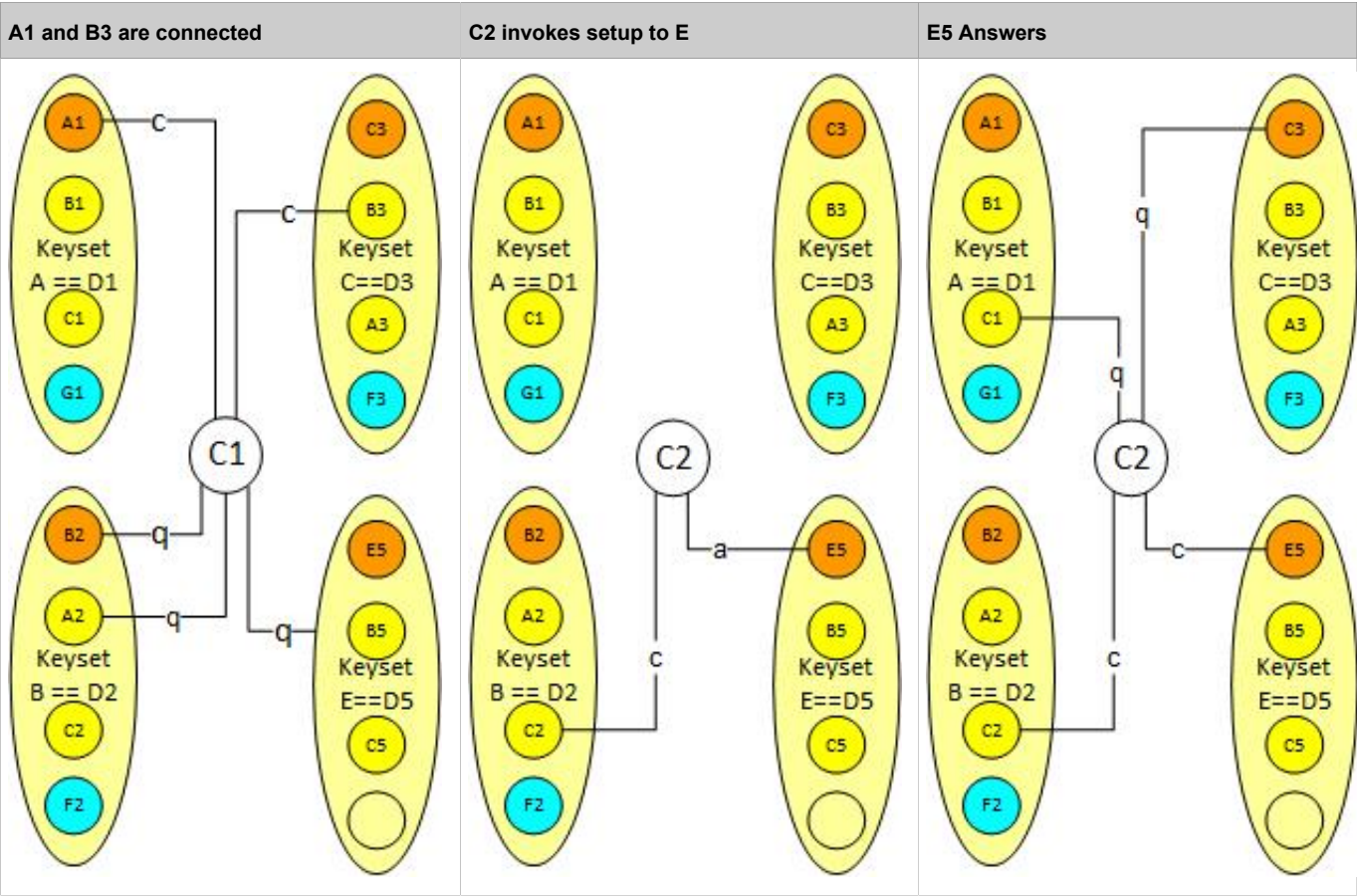


Table 385: Available Line Selection

Activity	Monitored Device C == D3		Monitored Device E == D5	
1. Application invokes call from appearance C2 to E via CSTA MakeCall.  NOTE: Application is responsible to select free line when using CSTA. Snapshot Device provides current status.	MakeCall	C/D2		
	callingDevice	E		
	calledDirectoryNumber	C/D2 C2		
	MakeCallResponse			
	callingDevice			
	ServiceInitiated	C/D2 C2		
	connection	C/D2		
	initiatingDevice	makeCall		
	cause	initiated		
	localConnectionInfo			

## Shared-bridged appearances modeling

Activity	Monitored Device C == D3		Monitored Device E == D5	
	Digits Dialed	C/D2 C2		
	diallingConnection	C/D2		
	diallingDevice	E		
	diallingSequence	Initiated		
	localConnectionInfo	normal		
	cause			
	Originated	C/D2 C2		
	connection	C/D2 C2		
	callingDevice	E		
	calledDevice	connected		
	localConnectionInfo	normal		
	cause			
2. E5 is ringing	<b>Delivered</b>	E/D5 C2	Delivered	E/D5 C2
	connection	E/D5	connection	E/D5
	alertingDevice	C/D2	alertingDevice	C/D2
	callingDevice	E	callingDevice	E
	calledDevice	notSpecified	calledDevice	notSpecified
	lastRedirectionDevice	normal	lastRedirectionDevice	normal
	cause	connected	cause	alerting
	localConnectionInfo		localConnectionInfo	
3. Application answers call from C2 on behalf E			Answer Call callToBeAnswered	E/D5 C2
			Answer Call Response	
4. C2 and E5 are connected	Established	E/D5 C2	Established	E/D5 C2
	connection	E/D5	connection	E/D5
	answeringDevice	C/D2	answeringDevice	C/D2
	callingDevice	E	callingDevice	E
	calledDevice	notSpecified normal	calledDevice	notSpecified
	lastRedirectionDevice	connected	lastRedirectionDevice	normal
	cause		cause	connected
	localConnectionInfo		localConnectionInfo	

Activity	Monitored Device C == D3		Monitored Device E == D5	
5. C1 and C3 are placed into the Bridged/ Queued state	BridgedEvent	C/D3 C2	BridgedEvent	C/D3 C2
	connection	C/D3	connection	C/D3
	bridgedAppearance	normal	bridgedAppearance	normal
	cause	queued	cause	connected
	localConnectionInfo		localConnectionInfo	
	BridgedEvent	C/D1 C2	BridgedEvent	C/D1 C2
	connection	C/D1	connection	C/D1
	bridgedAppearance	normal	bridgedAppearance	normal
	cause	queued	cause	connected
	localConnectionInfo		localConnectionInfo	

### 11.5.6.3 Releasing from Bridge Call using Clear Connection Service

When the Answer Call service is invoked on a Bridged/Queued connectionID a Conferenced Event is raised reusing the same callID. Subsequent Bridged and Conferenced events are raised as shared call appearance become active or passive on the bridged call. The Conference modeling of Bridged calls shall extend the use of keyOperation cause code to differentiate bridged-call from basic Conference calls.

- A1, B2 and B3 are connected on a Bridged call
- B3 releases from the bridged call
- A3 and B3 enter Bridged/Queued state per rules described in [Sect. 10.5.3.1](#)

Table 386: Releasing from Bridged Call

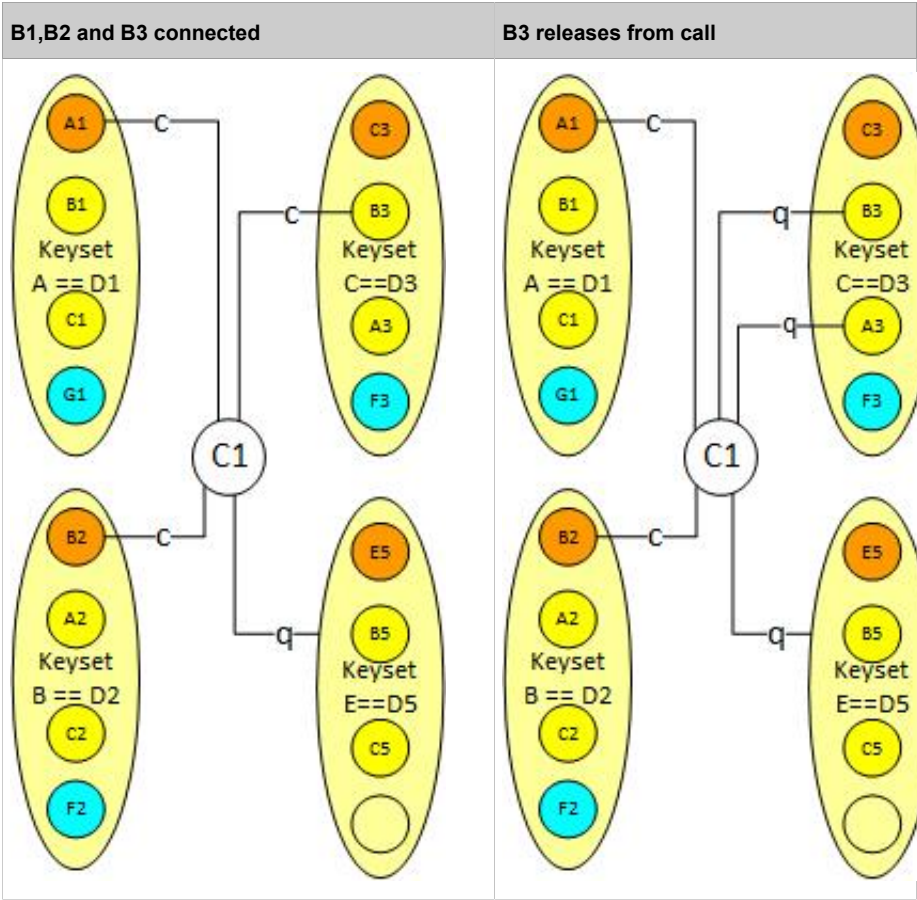


Table 387: Releasing from Bridged Cal

Activity	Monitored Device A		Monitored Device B	
1. Application releases B3 from Bridged call, leaving A1 and B2 connected.			ClearConnection connectionToBeCleared	B/D3 C1
	ConnectionCleared connection releasingDevice localConnectionInfo cause	B/D3 C1 B/D3 connected normalClearing	ConnectionCleared connection releasingDevice localConnectionInfo cause	B/D3 C1 B/D3 null normalClearing
2. Adjust Services Permitted for A/D1 and B/D3	CallInformationEvent connection device servicesPermitted	A/D1 C1 A/D1 SF dependent		

Activity	Monitored Device A		Monitored Device B	
			CallInformationEvent connection device servicesPermitted	B/D2 C1 B/D2 SF dependent
3. D5 remains in bridged/queued state. A3 and B3 enter bridged/queued state	BridgedEvent	B/D3 C1	BridgedEvent	B/D3 C1
	connection	B/D3	connection	B/D3
	bringedAppearance	normal	bringedAppearance	normal
	cause	connected	cause	queued
	localConnectionInfo		localConnectionInfo	
	BridgedEvent	B/D5 C1	BridgedEvent	B/D5 C1
	connection	B/D5	connection	B/D5
	bringedAppearance	normal	bringedAppearance	normal
	cause	connected	cause	queued
	localConnectionInfo		localConnectionInfo	
	BridgedEvent	A/D3 C1	BridgedEvent	A/D3 C1
	connection	A/D3	connection	A/D3
	bringedAppearance	normal	bringedAppearance	normal
	cause	queued	cause	connected
	localConnectionInfo		localConnectionInfo	

## 11.5.7 Terminating a Bridged Call

### 11.5.7.1 Device releases from a bridged connection

- A1 and B3 are talking (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- When one of the remaining two connections is cleared by the application or phone the entire call is released from all appearances.

Table 388: Device release

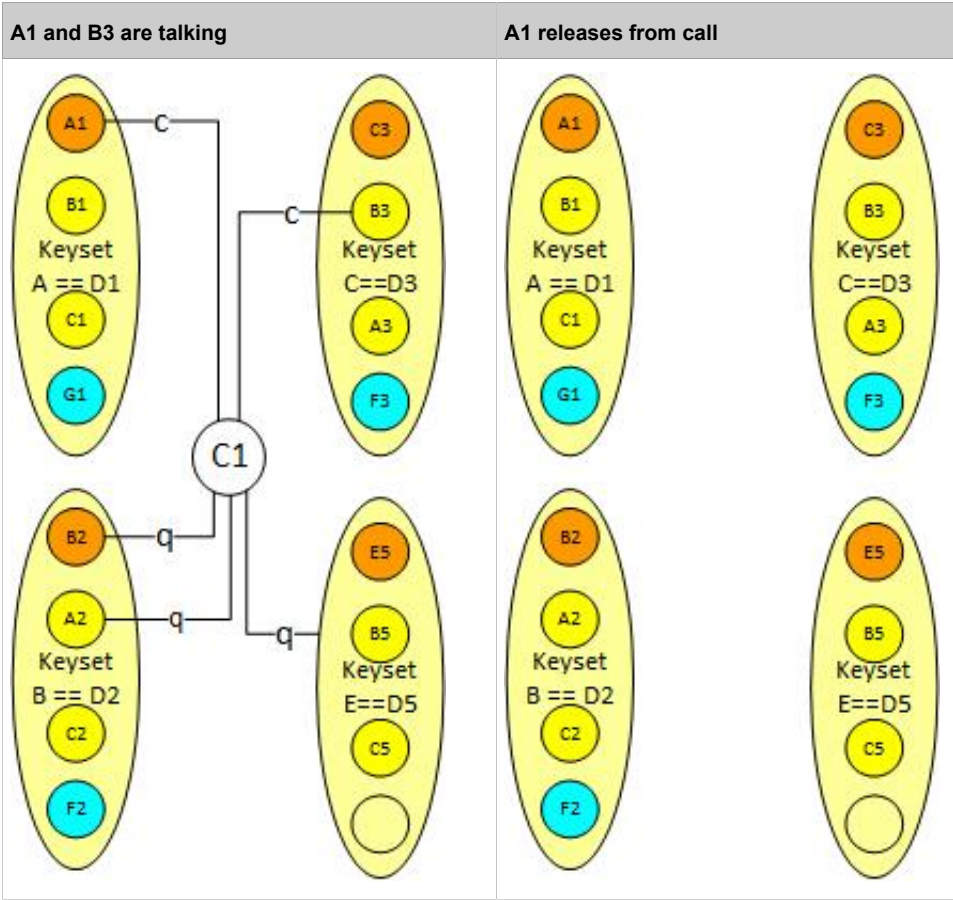


Table 389: Device release

Activity	Monitored Device A		Monitored Device B	
1. Application releases the active connection on A1	ClearConnection Request	A/D1 C1		
	connectionToBeCleared			
	ClearConnection Response			
2. All call appearances are cleared. See notes.	Failed	A/D1 C1	Failed	A/D1 C1
NOTE: Connection is cleared through blocked state => additional Failed (LCS failed) evt on B	connection	A/D1	connection	A/D1
	failingDevice	A/D1	failingDevice	A/D1
	callingDevice	B	callingDevice	B
	calledDevice	blocked	calledDevice	blocked
	cause	fail	cause	fail
	localConnectionInfo		localConnectionInfo	

Activity	Monitored Device A		Monitored Device B	
	ConnectionCleared	A/D1 C1**	ConnectionCleared	A/D1 C1**
	connection	A/D1 **	connection	A/D1 **
	releasingDevice	null	releasingDevice	connected
	localConnectionInfo	normalClearing	localConnectionInfo	normalClearing
	cause		cause	
			Failed	B/D3 C1
			connection	B/D3
			failingDevice	A/D1
			callingDevice	B
			calledDevice	blocked
			cause	fail
			localConnectionInfo	
			ConnectionCleared	B/D3 C1*
			connection	B/D3 *
			releasingDevice	null
			localConnectionInfo	normalClearing
			cause	
<p>* The droppedConnection is "B C1" and the releasingDevice is "B" because the event applies to all appearances of B. The alternative would be to send separate Connection Cleared events for each appearance.</p>				
<p>** The droppedConnection is "A C1" and the releasingDevice is "A" because the event applies to all appearances of A. The alternative would be to send separate Connection Cleared events for each appearance.</p>				

## 11.5.8 Hold and Retrieve Call Flows

### 11.5.8.1 Placing a Shared-Bridged appearance on Hold using CSTA Hold Call service

- A1 and B3 are talking
- B3 places shared-bridged appearance connections on hold

Table 390: Placing a Shared-Bridged appearance on Hold

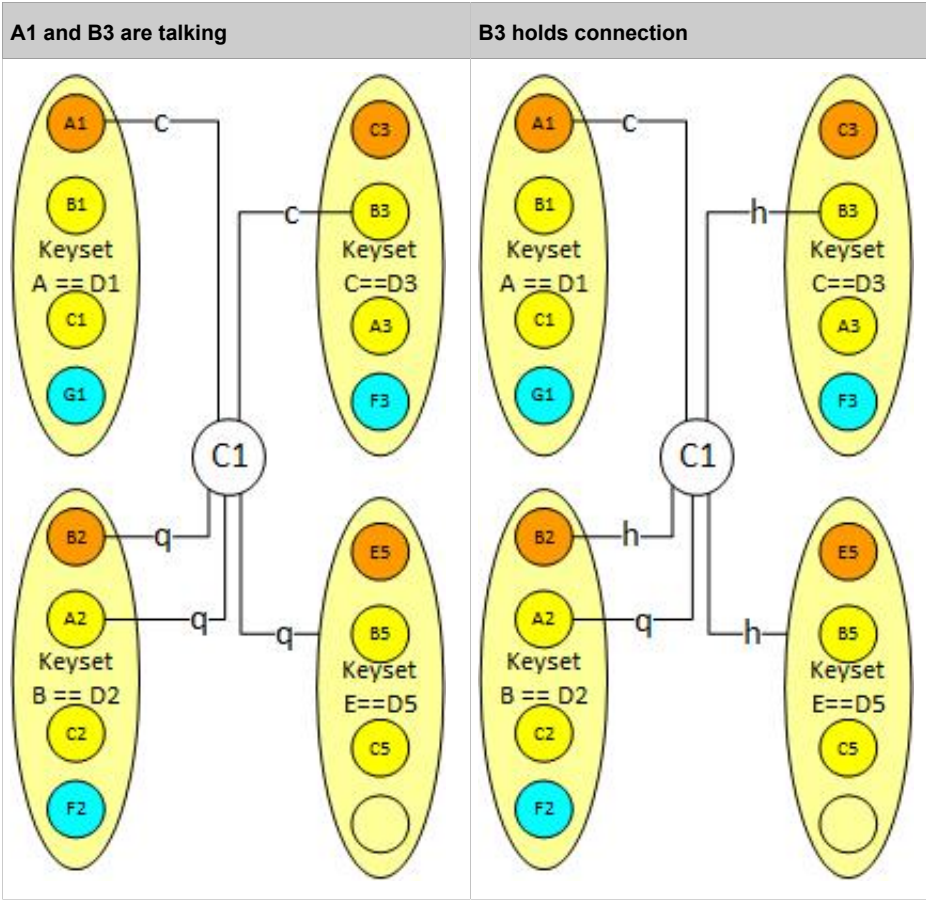


Table 391: Placing a Shared-Bridged appearance on Hold

Activity	Monitored Device A		Monitored Device B (ONS)	
1. Application invokes hold on behalf of B3			HoldCall connectionToBeHeld HoldCall Response	B/D3 C1
2. All shared-bridged appearances are placed on hold.  Note: Any held SBA connection may be retrieved. A2 can not bridge while B is holding the call.	HeldEvent	B/D3 C1	HeldEvent	B/D3 C1
	connection	B/D3	connection	B/D3
	holdingDevice	normal	holdingDevice	normal
	cause	connected	cause	hold
	localConnectionInfo		localConnectionInfo	
	HeldEvent	B/D2 C1	HeldEvent	B/D2 C1
	connection	B/D3	connection	B/D3
	holdingDevice	normal	holdingDevice	normal
	cause	connected	cause	hold
	localConnectionInfo		localConnectionInfo	

Activity	Monitored Device A		Monitored Device B (ONS)	
	HeldEvent	B/D5 C1	HeldEvent	B/D5 C1
	connection	B/D3	connection	B/D3
	holdingDevice	normal	holdingDevice	normal
	cause	connected	cause	hold
	localConnectionInfo		localConnectionInfo	
	CallInformationEvent	A/D2 C1		
	connection	A/D2		
	device	no services		
	servicesPermitted			

### 11.5.8.2 Retrieving a Shared-Bridged appearance on Hold using CSTA Retrieve Call service

- A1 is connected to Music on Hold while all shared-bridged appearances of B have been placed on hold as shown in [Sect. 10.5.8.1](#).
- B2 retrieves its shared-bridged appearance connection on hold
- A3 and B3 enter bridged/queued state. A2 is cleared.

Table 392: Retrieving a Shared-Bridged appearance on Hold

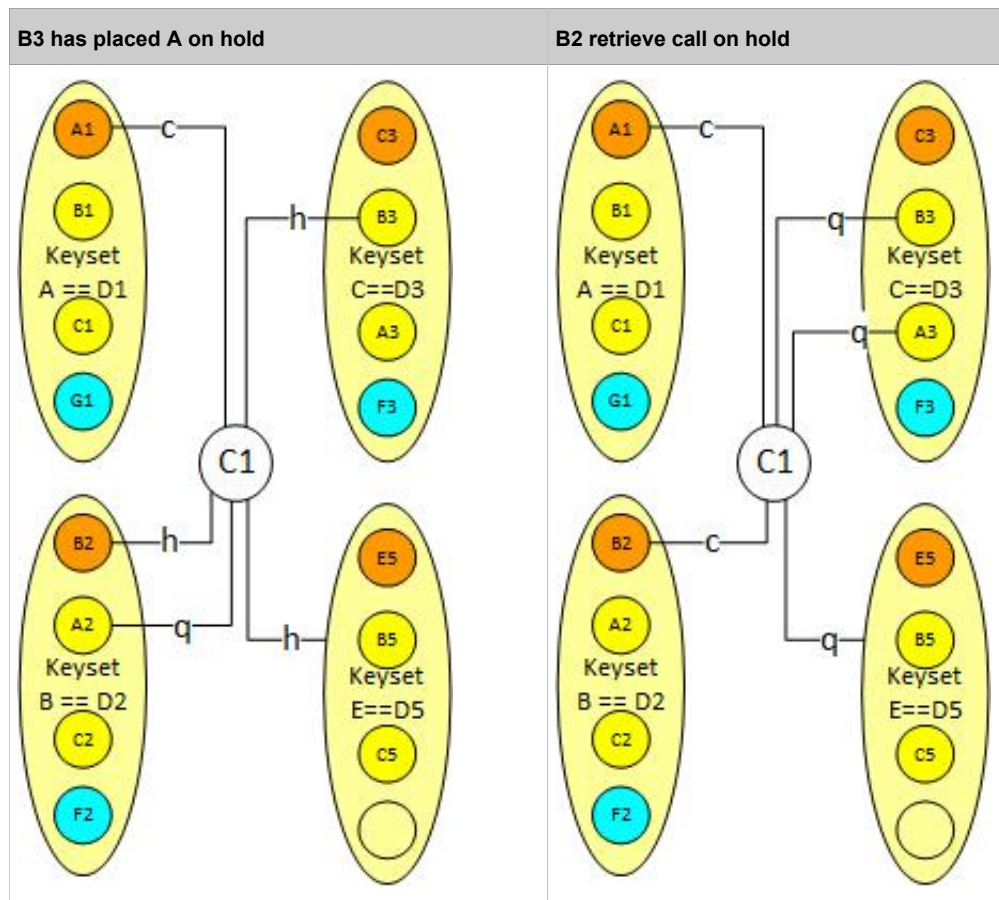


Table 393: Retrieving a Shared-Bridged appearance on Hold

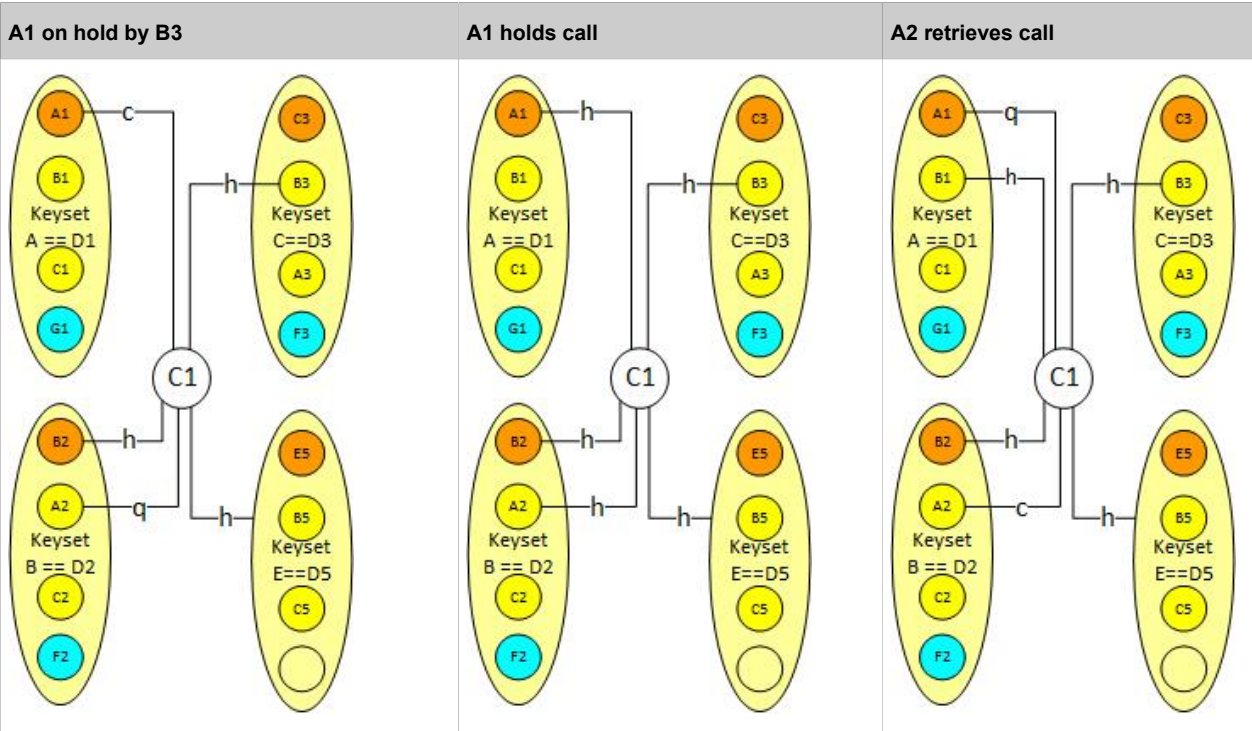
Activity	Monitored Device A		Monitored Device B (ONS)	
1. Application invokes Retrieve on behalf of B2			<b>RetrieveCall</b> callToBeRetrieved <b>RetrieveCall</b> <b>Response</b>	B/D2 C1
2. A1 and B2 are connected.	<b>Retrieved</b> connection retrievingDevice cause <b>localConnectionInfo</b>	B/D2 C1 B/D2 normal connected	<b>Retrieved</b> connection retrievingDevice cause <b>localConnectionInfo</b>	B/D2 C1 B/D2 normal connected
3. All other shared-bridged call appearances are placed in Bridged/Queued state.	<b>BridgedEvent</b> connection bridgedAppearance cause <b>localConnectionInfo</b>	B/D3 C1 B/D3 normal connected	<b>BridgedEvent</b> connection bridgedAppearance cause <b>localConnectionInfo</b>	B/D3 C1 B/D3 normal queued

Activity	Monitored Device A		Monitored Device B (ONS)	
	<b>BridgedEvent</b>	B/D5 C1	<b>BridgedEvent</b>	B/D5 C1
	connection	B/D5	connection	B/D5
	bridgedAppearance	normal	bridgedAppearance	normal
	cause	connected	cause	queued
	<b>localConnectionInfo</b>		<b>localConnectionInfo</b>	
4. A2 connection is not cleared, A3 is signaled and set to Bridged/Queued state.	<b>BridgedEvent</b>	A/D3 C1	<b>BridgedEvent</b>	A/D3 C1
	connection	A/D3	connection	A/D3
	bridgedAppearance	normal	bridgedAppearance	normal
	cause	queued	cause	connected
	<b>localConnectionInfo</b>		<b>localConnectionInfo</b>	

11.5.8.3 Placing a Shared-Bridged appearance on Hold using CSTA Hold Call service while on-hold

- A1 has been placed on hold by B3
- A1 call on hold (hold-on-hold)
- A2 retrieves held call

Table 394: Placing a Shared-Bridged appearance on Hold



**Table 395: Placing a Shared-Bridged appearance on Hold**

Activity	Monitored Device A		Monitored Device B (ONS)	
1. Application invokes hold call on behalf of A1	<b>HoldCall</b> connectionToBeHeld <b>HoldCall Response</b>	A/D1 C1		
2. If shared-bridged appearances are placed on hold.	<b>HeldEvent</b> connection holdingDevice cause localConnectionInfo	A/D1 C1 A/D1 normal hold	<b>HeldEvent</b> connection holdingDevice cause localConnectionInfo	A/D1 C1 A/D1 normal hold
	<b>HeldEvent</b> connection holdingDevice cause localConnectionInfo	A/D2 C1 A/D1 normal hold	<b>HeldEvent</b> connection holdingDevice cause localConnectionInfo	A/D2 C1 A/D1 normal hold
3. Application invokes retrieve call on behalf of A2	<b>RetrieveCall</b> connectionToBeRetrieved <b>RetrieveCall Response</b>	A/D2 C1		
4. A2 is now connected	<b>Retrieved</b> connection retrievingDevice cause localConnectionInfo	A/D2 C1 A/D2 normal connected	<b>RetrievedEvent</b> connection retrievingDevice cause localConnectionInfo	A/D2 C1 A/D2 normal hold
5. A1 is placed in the Bridged/Queued state	<b>BridgedEvent</b> connection bridgedAppearance cause localConnectionInfo	A/D1 C1 A/D1 normal queued	<b>BridgedEvent</b> connection bridgedAppearance cause localConnectionInfo	A/D1 C1 A/D1 normal hold
6. B/D1 is placed on in queued and then hold state	<b>BridgedEvent</b> connection bridgedAppearance cause localConnectionInfo	B/D1 C1 B/D1 normal connected	<b>BridgedEvent</b> connection bridgedAppearance cause localConnectionInfo	B/D1 C1 B/D1 normal queued

Activity	Monitored Device A		Monitored Device B (ONS)	
	<b>BridgedEvent</b>	B/D5 C1	<b>BridgedEvent</b>	B/D5 C1
	connection	B/D5	connection	B/D5
	bridgedAppearance	normal	bridgedAppearance	normal
	cause	connected	cause	queued
	localConnectionInfo		localConnectionInfo	
	<b>HeldEvent</b>	B/D5 C1	<b>HeldEvent</b>	B/D5 C1
	connection	A/D1	connection	A/D1
	holdingDevice	normal	holdingDevice	normal
	cause	connected	cause	hold
	localConnectionInfo		localConnectionInfo	
	<b>HeldEvent</b>	B/D1 C1	<b>HeldEvent</b>	B/D1 C1
	connection	A/D1	connection	A/D1
	holdingDevice	normal	holdingDevice	normal
	cause	connected	cause	hold
	localConnectionInfo		localConnectionInfo	
	<b>CallInformationEvent</b>	A/D1 C1		
	connection	A/D1		
	device	none		
	servicesPermitted			

## 11.5.9 Call Forwarding and Call Diversion

General note: Diverted event by default is provided only to the diverting device. There is an option to make it available also for the partner. Delivered events to called side will be provided after the delivered events for called side.

### 11.5.9.1 Call Forward No Answer - Logical Device

- A1 calls B (B Call Forwards on No Answer to C)
- All shared-bridged appearances of B are ringing
- B's no answer timer expires and calls divert to C
- C5 answers call forwarded from B

Table 396: Call Forward

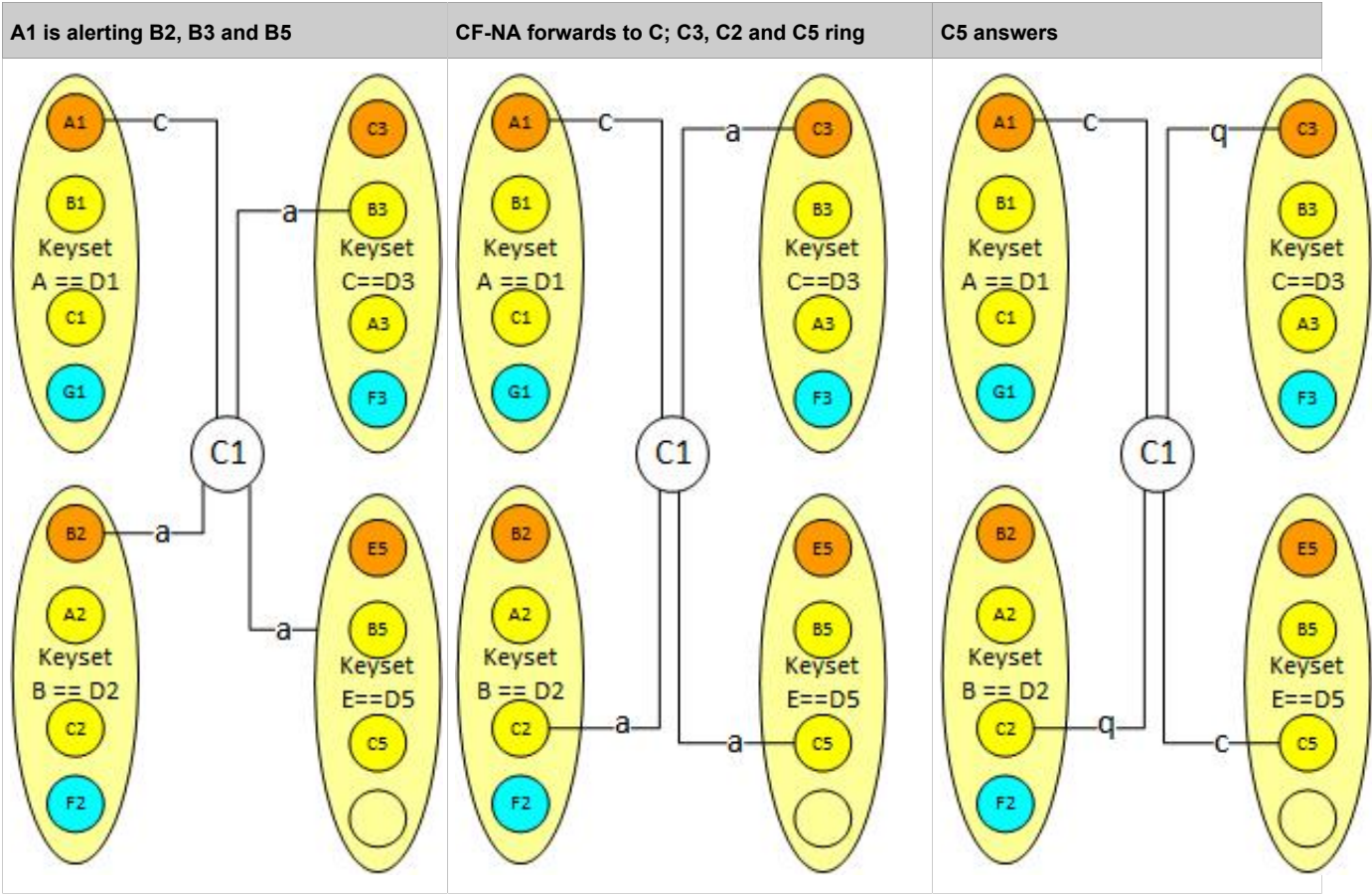


Table 397: Call Forward

Activity	Monitored Device A == D1		Monitored Device B (ONS)		Monitored Device C	
1. B's No Answer Timer expires and call is diverted to C			<b>Diverted</b>	B C1		
			connection	B		
			divertingDevice	C		
			newDestination	A/D1		
			callingDevice	B		
			calledDevice	notSpecified		
			lastRedirection-Device	forwardNoAnswer		
			cause	connected		
			localConnectionInf			

Activity	Monitored Device A == D1		Monitored Device B (ONS)		Monitored Device C	
2. C3, C2 and C5 ring	Delivered	C/D3 C1			Delivered	C/D3 C1
	connection	C/D3			connection	C/D3
	alertingDevice	A/D1			alertingDevice	A/D1
	callingDevice	B			callingDevice	B
	calledDevice	B			calledDevice	B
	lastRedirection-Device	forwardNoAnswer			lastRedirection-Device	forwardNoAnswer
	cause	connected			cause	alerting
	localConnectionInfo				localConnectionInfo	
	Delivered	C/D2 C1			Delivered	C/D2 C1
	connection	C/D2			connection	C/D2
	alertingDevice	A/D1			alertingDevice	A/D1
	callingDevice	B			callingDevice	B
	calledDevice	B			calledDevice	B
	lastRedirection-Device	forwardNoAnswer			lastRedirection-Device	forwardNoAnswer
	cause	connected			cause	alerting
	localConnectionInfo				localConnectionInfo	
	Delivered	C/D5 C1			Delivered	C/D5 C1
	connection	C/D5			connection	C/D5
	alertingDevice	A/D1			alertingDevice	A/D1
	callingDevice	B			callingDevice	B
	calledDevice	B			calledDevice	B
	lastRedirection-Device	forwardNoAnswer			lastRedirection-Device	forwardNoAnswer
	cause	connected			cause	alerting
	localConnectionInfo				localConnectionInfo	
3. C5 answers	Established	C/D5 C1			Established	C/D5 C1
	connection	C/D5			connection	C/D5
	answeringDevice	A/D1			answeringDevice	A/D1
	callingDevice	B			callingDevice	B
	calledDevice	notSpecified			calledDevice	notSpecified
	lastRedirection-Device	normal			lastRedirection-Device	normal
	cause	connected			cause	connected
	localConnectionInfo				localConnectionInfo	

## Shared-bridged appearances modeling

Activity	Monitored Device A == D1		Monitored Device B (ONS)		Monitored Device C	
4. A2, C2 and C3 in bridged/ queued state	BridgedEvent	C/D2 C1			BridgedEvent	C/D2 C1
	connection	C/D2			connection	C/D2
	bridedAppearance	normal			bridedAppearance	normal
	cause	connected			cause	queued
	localConnectionInfo				localConnectionInfo	
	BridgedEvent	C/D3 C1			BridgedEvent	C/D3 C1
	connection	C/D3			connection	C/D3
	bridedAppearance	normal			bridedAppearance	normal
	cause	connected			cause	queued
	localConnectionInfo				localConnectionInfo	
	BridgedEvent	A/D3 C1			BridgedEvent	A/D3 C1
	connection	A/D3			connection	A/D3
	bridedAppearance	normal			bridedAppearance	normal
	cause	queued			cause	connected
	localConnectionInfo				localConnectionInfo	
	BridgedEvent	A/D2 C1			BridgedEvent	A/D2 C1
	connection	A/D2			connection	A/D2
	bridedAppearance	normal			bridedAppearance	normal
	cause	queued			cause	connected
	localConnectionInfo				localConnectionInfo	

### 11.5.9.2 Deflect Call - From Hunt Group to logical device

- Call from X1 is queued on Hunt Group P1
- Application Deflect Call to B
- All shared-bridged appearances of B are ringing
- B3 answers call

Table 398: Deflect Call

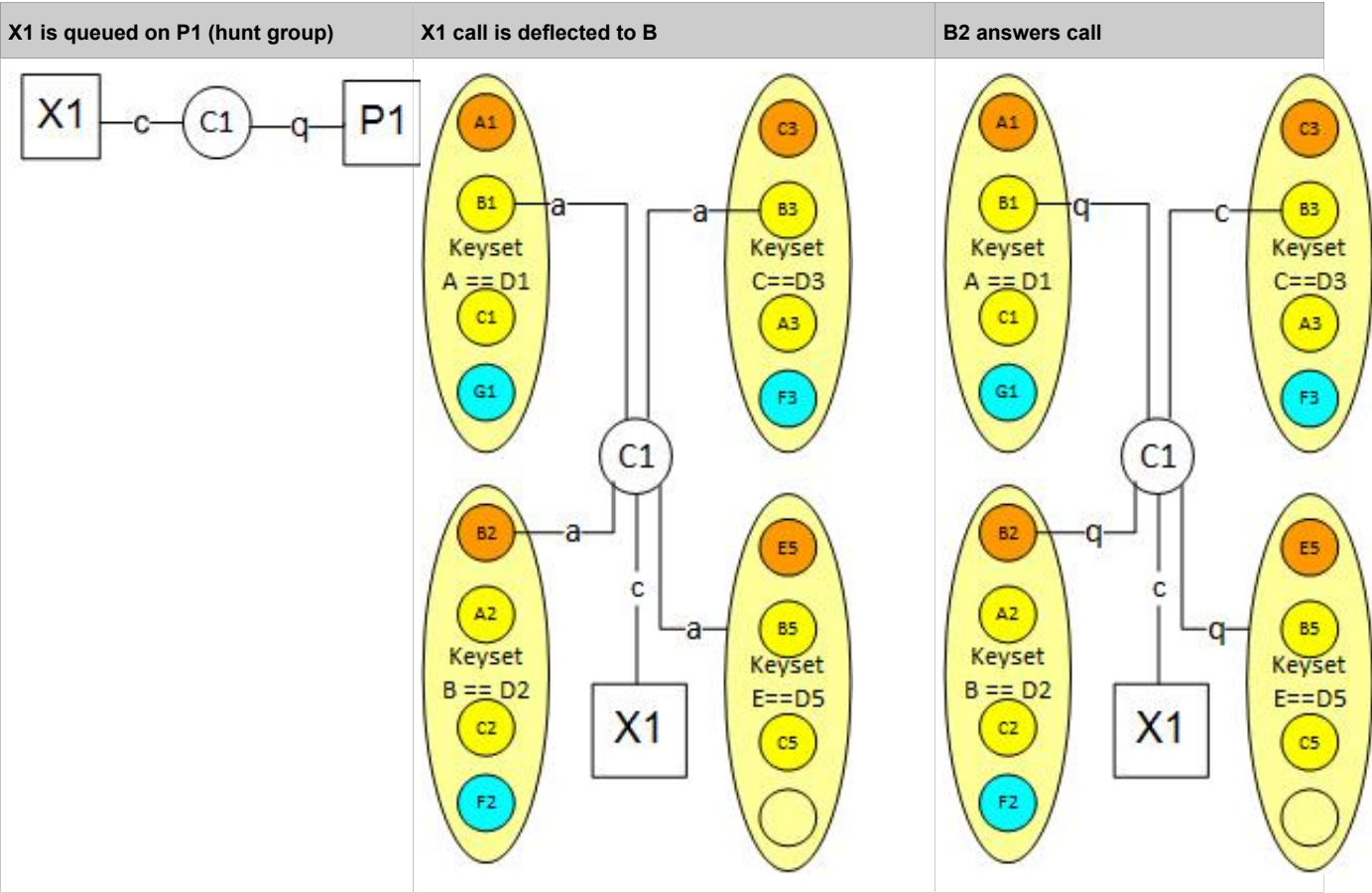


Table 399: Deflect Call

Activity	P1 (Hunt Group)		Monitored Device B == D2	
1. Application moves X1's call from Hunt Group P1 to B	Deflect Call	P1 C1		
	callToBeDiverted	B		
	newDestination			
	DeflectCallResponse			
	Diverted	P1 C1		
	connection	P1		
	divertingDevice	B		
	newDestination	X1		
	callingDevice	P1		
	calledDevice	notSpecified		
	lastRedirectionDevice	connected		
	localConnectionInfo	redirected		
	cause			

## Shared-bridged appearances modeling

Activity	P1 (Hunt Group)		Monitored Device B == D2	
2. All Appearances of B are ringing.			Delivered	B/D1 C1
			connection	B/D1
			alertingDevice	X1
			callingDevice	P1
			calledDevice	P1
			lastRedirectionDevice	redirected
			cause	alerting
			localConnectionInfo	
			Delivered	B/D2 C1
			connection	B/D2
			alertingDevice	X1
			callingDevice	P1
			calledDevice	P1
			lastRedirectionDevice	redirected
			cause	alerting
			localConnectionInfo	
		Delivered	B/D3 C1	
		connection	B/D3	
		alertingDevice	X1	
		callingDevice	P1	
		calledDevice	P1	
		lastRedirectionDevice	redirected	
		cause	alerting	
		localConnectionInfo		
		Delivered	B/D5 C1	
		connection	B/D5	
		alertingDevice	X1	
		callingDevice	P1	
		calledDevice	P1	
		lastRedirectionDevice	redirected	
		cause	alerting	
		localConnectionInfo		

Activity	P1 (Hunt Group)		Monitored Device B == D2	
3. Application answers call from A1 on behalf B3			Answer Call callToBeAnswered Answer Call Response	B/D3 C1
4. A1 and B3 are connected  <i>NOTE: Established and Bridged Events on device B monitor may be presented before the Established and Bridged Events on device A monitor.</i>			Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	B/D3 C1 B/D3 X1 P1 notSpecified normal connected
5. A2, B2 and B5 are in Bridged/Queued state			BridgedEvent connection bridedAppearance cause localConnectionInfo	B/D1 C1 B/D1 normal queued
			BridgedEvent connection bridedAppearance cause localConnectionInfo	B/D2 C1 B/D2 normal queued
			BridgedEvent connection bridedAppearance cause localConnectionInfo	B/D5 C1 B/D5 normal queued

## 11.5.10 Transfer

### 11.5.10.1 Semi-Attended/Blind Transfer

- A1 and B3 are talking (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- B3 consults to E
- B3 transfers A1 to E during ringing

Table 400: Semi-Attended/Blind Transfer

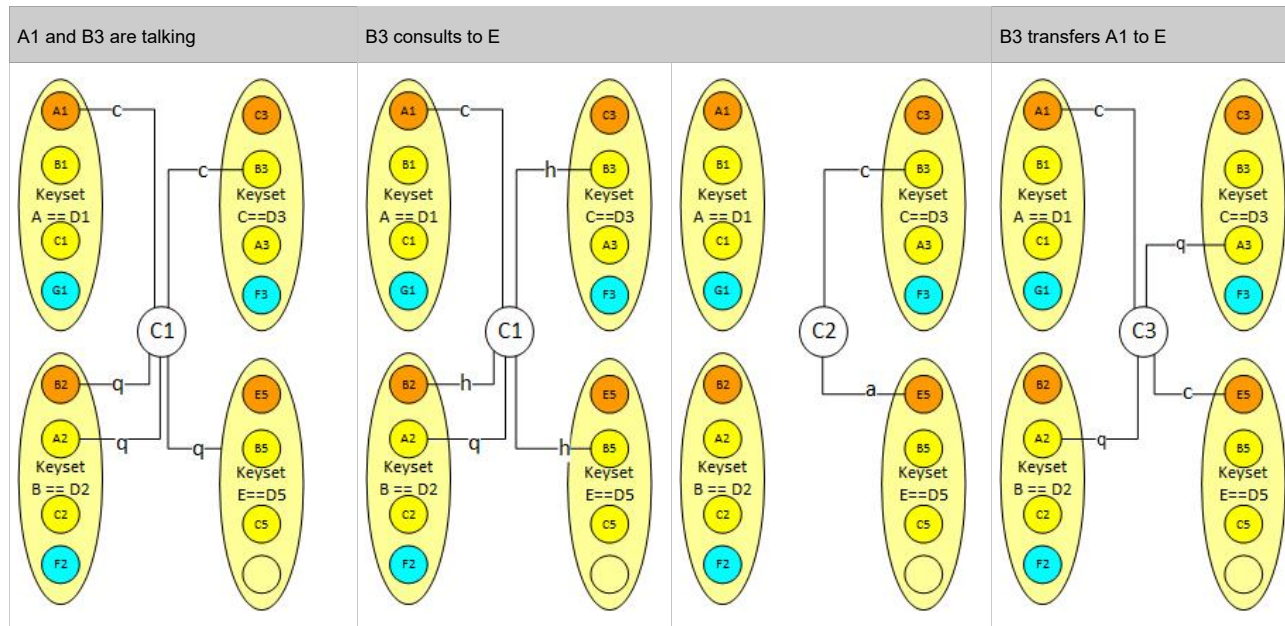


Table 401: Semi-Attended/Blind Transfer

Activity	Monitored Device A	Monitored Device B	Monitored Device E
1. Application invokes consultation to E on behalf of B3		ConsultationCall Request  existingCall  consultedDevice  ConsultationCall Response  initiatedCall	B/D3 C1  E  B/D3 C2
	Held connection holdingDevice cause localConnectionInfo	B/D3 C1  B/D3 consultation connected  localConnectionInfo	Held connection B/D3 C1  B/D3 consultation hold  localConnectionInfo
	Held connection holdingDevice cause localConnectionInfo	B/D2 C1  B/D3 consultation connected  localConnectionInfo	<b>Held</b> connection B/D2 C1  B/D3 consultation hold  localConnectionInfo

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
	Held connection holdingDevice cause localConnectionInfo	B/D5 C1 B/D3 consultation connected	Held connection holdingDevice cause localConnectionInfo	B/D5 C1 B/D3 consultation hold		
2. Update A/ D2 services permitted.	CallInformationEvent connection device servicesPermitted	A/D2 C1 A/D2 none				
3. B3 call setup to D5  NOTE: held + service initiated for holding are sent first			ServiceInitiated connection initiatingDevice cause localConnectionInfo	B/D3 C2 B/D3 consultation initiated		
			Digits Dialed diallingConnection diallingDevice diallingSequence localConnectionInfo cause	B/D3 C2 B/D3 E Initiated Consultation		
			Originated connection callingDevice calledDevice localConnectionInfo cause	B/D3 C2 B/D3 E connected consultation		
			Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C2 E/D5 B/D3 E notSpecified normal connected	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C2 E/D5 B/D3 E notSpecified normal alerting

## Shared-bridged appearances modeling

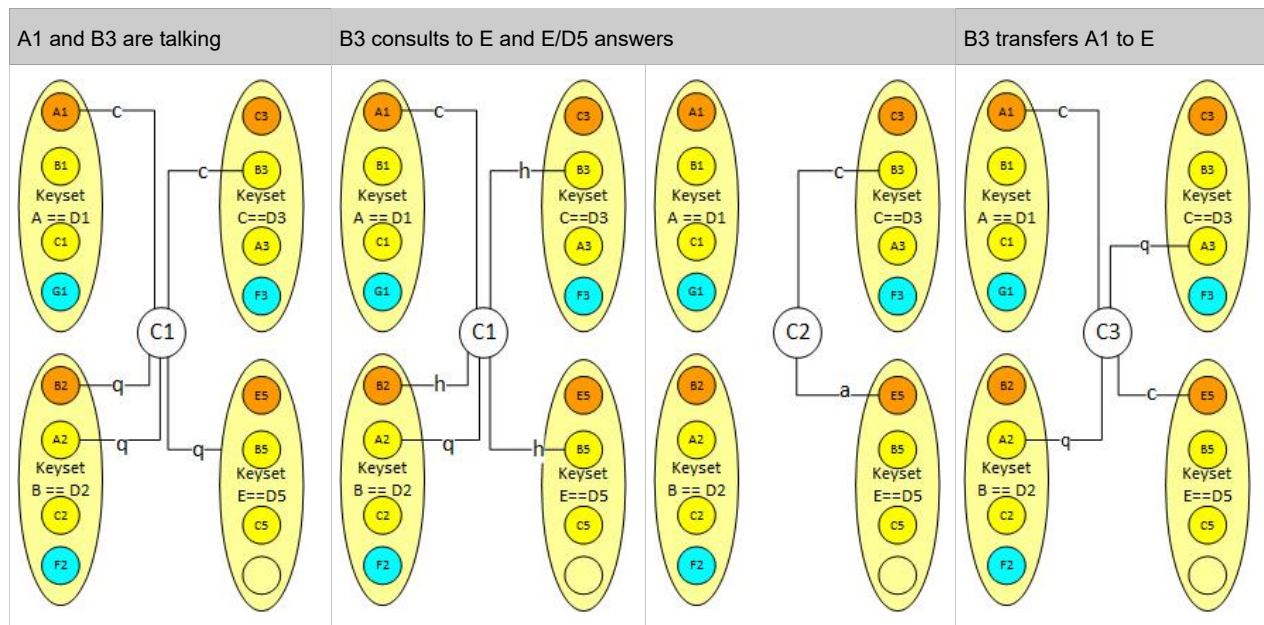
Activity	Monitored Device A		Monitored Device B		Monitored Device E	
<p>4. Application invokes transfer of A1 to E</p> <p>NOTE: transferred connection list with new and old call IDs</p> <p>*Physical device is not reported for E in this event</p>			Transfer Call Request	B/D3 C1		
			heldCall	B/D3 C2		
			activeCall	E C3		
			Transfer Call Response			
			transferredCall			
	Transferred	A/D1 C1	Transferred	B/D3 C1	Transferred	E/D5 C2
	primaryOldCall	Not Present	primaryOldCall	B/D3 C2	primaryOldCall	Not Present
	secondaryOldCall	B/D3	secondaryOldCall	B/D3	secondaryOldCall	B/D3
	transferringDevice	E/D5	transferringDevice	E *	transferringDevice	E/D5
	transferredToDevice	A/D1 C3	transferredToDevice	A/D1 C3	transferredToDevice	A/D1 C3
	connection list	A/D1 C1	connection list	A/D1 C1	connection list	E/D5 C3
	newConnection	E/D5 C3	newConnection	E * C3	newConnection	E/D5 C2
	oldConnection	connected	oldConnection	E * C1	connection list	alerting
	connection list	transfer	connection list	null	newConnection	transfer
<p>5. Clear A2,B2 and B5 callID C1</p>			newConnection	transfer	oldConnection	
			localConnectionInfo		localConnectionInfo	
			cause		cause	
			ServiceInitiated	B/D3 Cx		
			Connection	B/D3		
			initiatingDevice	normal		
			cause	initiated		
			localConnectionInfo			
			Failed	B/D3 Cx		
			connection	B/D3		
			failingDevice	B/D3		
			callingDevice	NotKnown		
			calledDevice	normal		
			cause	fail		
			localConnectionInfo			

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
			ConnectionCleared connection releasingDevice localConnectionInfo cause	B/D3 Cx B/D3 null normalClearing		
6. D5 answers	Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C3 E/D5 A/D1 E notSpecified normal connected			Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C3 E/D5 A/D1 E notSpecified normal connected
7. A2 and A3 are placed in the Bridged/ Queued state	BridgedEvent connection bridedAppearance cause localConnectionInfo	A/D2 C3 A/D2 normal queued			BridgedEvent connection bridedAppearance cause localConnectionInfo	A/D2 C3 A/D2 normal connected
	BridgedEvent connection bridedAppearance cause localConnectionInfo	A/D3 C3 A/D3 normal queued			BridgedEvent connection bridedAppearance cause localConnectionInfo	A/D3 C3 A/D3 normal connected

### 11.5.10.2 Attended Transfer

- A1 and B3 are talking (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- B3 consults to E
- E5 answer call from B3
- B3 transfers A1 to E5
- E5 Answers

## Shared-bridged appearances modeling

**Table 402: Attended transfer****Table 403: Attended transfer**

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
1. Application invokes consultation to E on behalf of B3			Consultation Call Request	B/D3 C1		
			existingCall	E		
			consultedDevice	B/D3 C2		
			Consultation Call Response			
			initiatedCall			
	Held	B/D3 C1	Held	B/D3 C1		
	connection	B/D3	connection	B/D3		
	holdingDevice	consultation	holdingDevice	consultation		
	cause	connected	cause	hold		
	localConnectionInfo		localConnectionInfo			
	Held	B/D2 C1	Held	B/D2 C1		
	connection	B/D3	connection	B/D3		
	holdingDevice	consultation	holdingDevice	consultation		
	cause	connected	cause	hold		
	localConnectionInfo		localConnectionInfo			

## Shared-bridged appearances modeling

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
	Held	B/D5 C1	Held	B/D5 C1		
	connection	B/D3	connection	B/D3		
	holdingDevice	consultation	holdingDevice	consultation		
	cause	connected	cause	hold		
	localConnectionInfo		localConnectionInfo			
2. Update A/D2 services permitted.	CallInformationEvent	A/D2 C1				
	connection	A/D2				
	device	none				
	servicesPermitted					
3. B3 call setup to D5  <i>Note: held + service initiated for holding are sent first.</i>			ServiceInitiated	B/D3 C2		
			connection	B/D3		
			initiatingDevice	consultation		
			cause	initiated		
			localConnectionInfo			
			Digits Dialed	B/D3 C2		
			diallingConnection	B/D3		
			diallingDevice	E		
			diallingSequence	initiated		
			localConnectionInfo	consultation		
			cause			
			Originated	B/D3 C2		
			connection	B/D3		
			callingDevice	E		
			calledDevice	connected		
			localConnectionInfo	consultation		
			cause			
			Delivered	E/D5 C2	Delivered	E/D5 C2
			connection	E/D5	connection	E/D5
			alertingDevice	B/D3	alertingDevice	B/D3
			callingDevice	E	callingDevice	E
			calledDevice	notSpecified	calledDevice	notSpecified
			lastRedirectionDevice	normal	lastRedirectionDevice	normal
			cause	connected	cause	alerting
			localConnectionInfo		localConnectionInfo	

## Shared-bridged appearances modeling

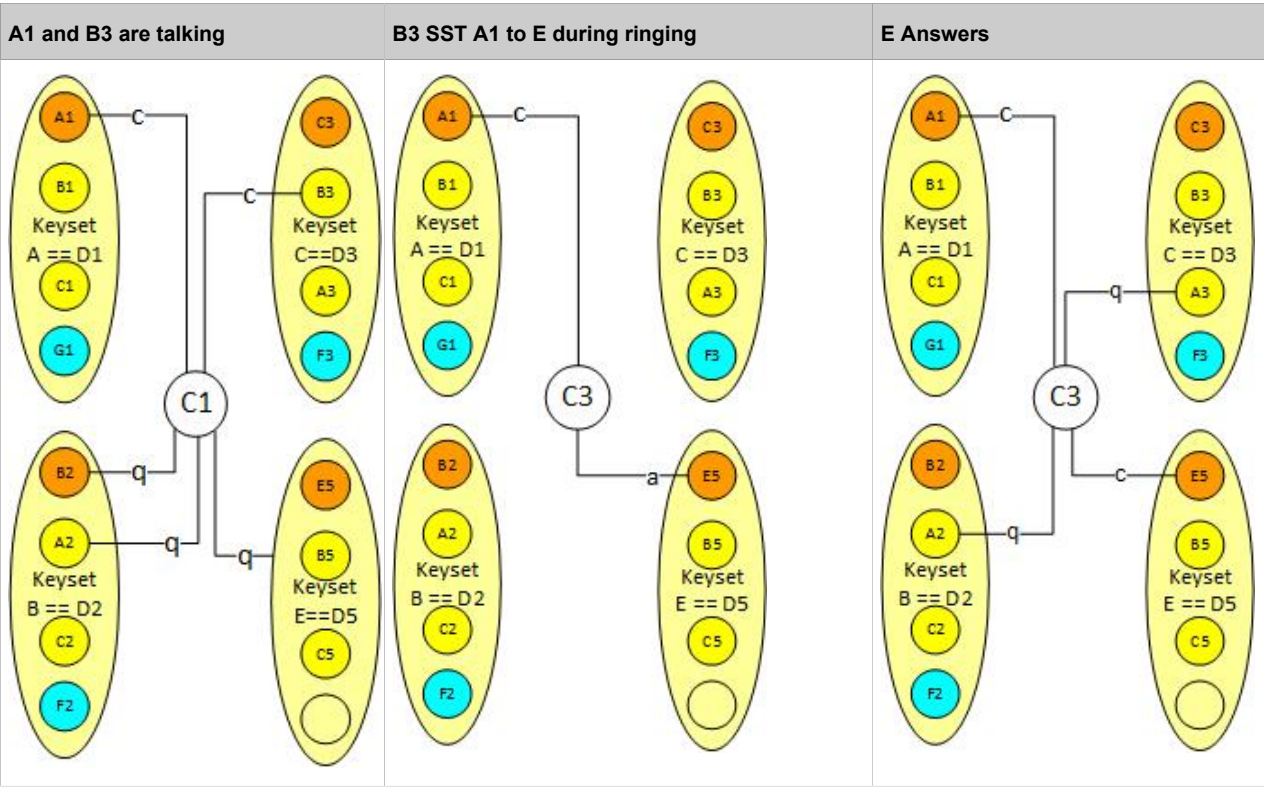
Activity	Monitored Device A		Monitored Device B		Monitored Device E	
4. D5 Answers			Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C2 E/D5 B/D3 E notSpecified normal connected	Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C2 E/D5 B/D3 E notSpecified normal connected
5.Application invokes transfer of A1 to E  <i>Note: transferred connection list with new and old call IDs</i>			Transfer Call Request  heldCall  activeCall  <b>Transfer CallResponse</b>  transferredCall	B/D3 C1  B/D3 C2  E C3		
	Transferred	A/D1 C1	<b>Transferred</b>	B/D3 C1	<b>Transferred</b>	E/D5 C2
	primaryOldCall	Not Present	primaryOldCall	B/D3	primaryOldCall	Not Present
	secondaryOldCall	B/D3	secondaryOldCall	B/D3	secondaryOldCall	B/D3
	transferringDevice	E/D5	transferringDevice	E/D5	transferringDevice	E/D5
	transferredToDevice	A/D1 C3	transferredToDevice	A/D1 C3	transferredToDevice	A/D1 C3
	connection list	A/D1 C1	connection list	A/D1 C1	connection list	E/D5 C3
	newConnection	E/D5 C3	newConnection	E/D5 C3	newConnection	E/D5 C2
	oldConnection	connected	oldConnection	E/D5 C2	connection list	connected
	connection list	transfer	connection list	null	newConnection	transfer
	newConnection		newConnection	transfer	oldConnection	
	localConnectionInfo		oldConnection		localConnectionInfo	
	cause		localConnectionInfo		cause	
	cause		cause			
6.A2 and A3 are placed in the Bridged/Queued state	<b>BridgedEvent</b> connection bridedAppearance cause localConnectionInfo	A/D2 C3  A/D2  transfer  queued			<b>BridgedEvent</b> connection bridedAppearance cause localConnectionInfo	A/D2 C3  A/D2  transfer  connected

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
	<b>BridgedEvent</b>	A/D3 C3			<b>BridgedEvent</b>	A/D3 C3
	connection	A/D3			connection	A/D3
	bridgedAppearance	transfer			bridgedAppearance	transfer
	cause	queued			cause	connected
	localConnectionInfo				localConnectionInfo	

11.5.10.3 Single Step Transfer

- A1 and B3 are talking (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- B3 transfer A1 to E in single step

Table 404: Single Step Transfer



**Table 405: Single Step Transfer**

Activity	Monitored Device A		Monitored Device B (ONS)		Monitored Device E	
1. Application transfer A1 to E on behalf of B3  NOTE: no new call ID in case of SST			SingleStepTransferRequest connection deviceToTransferTo	B/D3 C1 E E C1		
	Transferred primaryOldCall secondaryOldCall transferringDevice transferredToDevice connection list newConnection connection list newConnection localConnectionInfo cause	A/D1 C1 Not Present B/D3 E/D5 A/D1 C1 E/D5 C1 connected singleStepXfer	Transferred primaryOldCall secondaryOldCall transferringDevice transferredToDevice connection list newConnection connection list newConnection localConnectionInfo cause	B/D3 C1 Not Present B/D3 E A/D1 C1 E C1 null singleStepXfer		
2. E/D5 is ringing	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C1 E/D5 A/D1 E notSpecified singleStepXfer connected			Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C1 E/D5 A/D1 E notSpecified singleStepXfer alerting
3. E/D5 answers	Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C1 E/D5 A/D1 E notSpecified normal connected			Established connection answeringDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C1 E/D5 A/D1 E notSpecified normal connected

Activity	Monitored Device A		Monitored Device B (ONS)		Monitored Device E	
4. Unblocks B  <i>NOTE: unblock depends on timer</i>			ServiceInitiated	B/D3 Cx		
			Connection	B/D3		
			initiatingDevice	normal		
			cause	initiated		
			localConnectionInfo			
			ConnectionCleared	B/D3 Cx*		
			droppedConnection	B/D3		
			releasingDevice	null		
			localConnectionInfo	normalClearing		
			cause			
7. A2 and A3 are placed in the Bridged/Queued state	BridgedEvent	A/D2 C3			BridgedEvent	A/D2 C3
	connection	A/D2			connection	A/D2
	bridedAppearance	normal			bridedAppearance	normal
	cause	queued			cause	conneted
	localConnectionInfo				localConnectionInfo	
	BridgedEvent	A/D3 C3			BridgedEvent	A/D3 C3
	connection	A/D3			connection	A/D3
	bridedAppearance	normal			bridedAppearance	normal
	cause	queued			cause	connected
	localConnectionInfo				localConnectionInfo	

#### 11.5.10.4 Semi-Attended/Blind Transfer (with multiple consulted appearances alerting)

- Similar to previous example except E has an additional appearance on J = D8
- A1 and B3 are talking (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- B3 consults to E with appearances E1 and E8
- B3 transfers A1 to E during ringing
- E8 Answers

Table 406: Semi-Attended/Blind Transfer

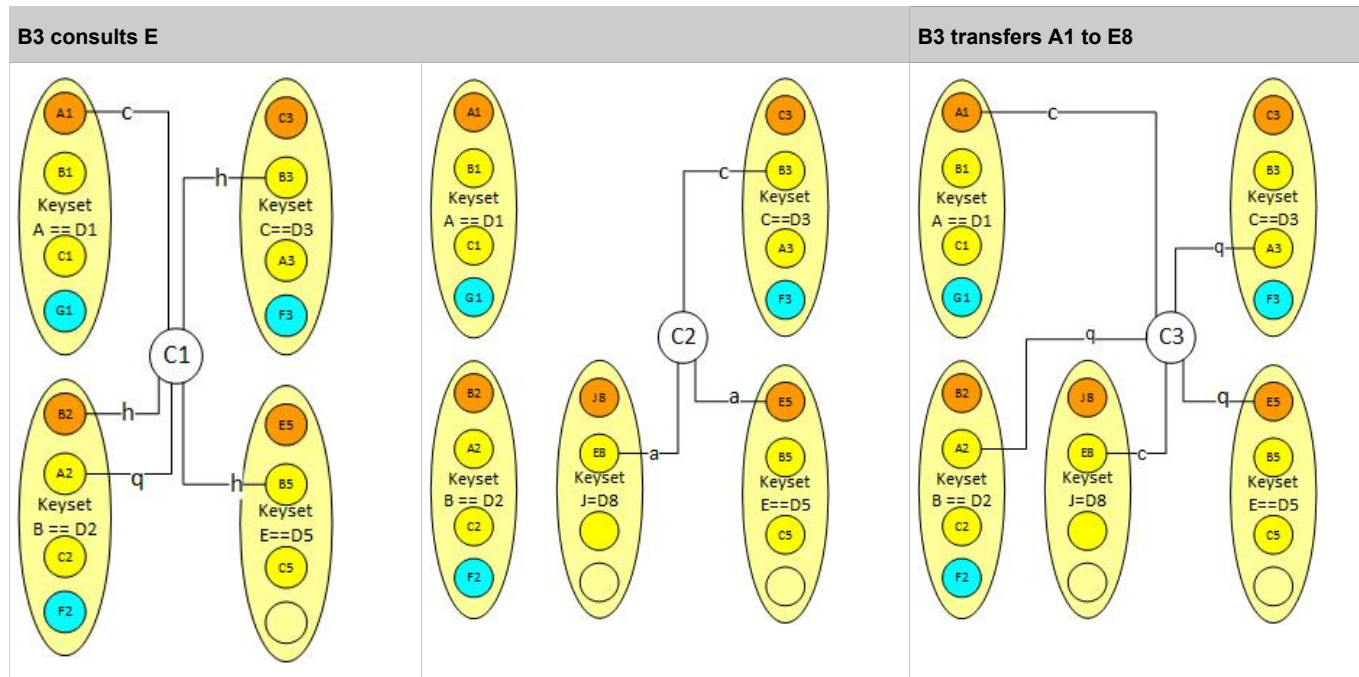


Table 407: Semi-Attended/Blind Transfer

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
1. Application invokes consultation to E on behalf of B3			Consultation Call Request existingCall consultedDevice ConsultationCall Response initiatedCall	B/D3 C1 E B/D3 C2		
Held connection holdingDevice cause localConnectionInfo	B/D3 C1 B/D3 consultation connected		Held connection holdingDevice cause localConnectionInfo	B/D3 C1 B/D3 consultation hold		
Held connection holdingDevice cause localConnectionInfo	B/D2 C1 B/D3 consultation connected		Held connection holdingDevice cause localConnectionInfo	B/D2 C1 B/D3 consultation hold		

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
	Held connection holdingDevice cause localConnectionInfo	B/D5 C1 B/D3 consultation connected	Held connection holdingDevice cause localConnectionInfo	B/D5 C1 B/D3 consultation hold		
2. Update A/ D2 services	CallInformationEvent connection device servicesPermitted	A/D2 C1 A/D2 none				
3. B3 call setup to D5  <i>NOTE: held + service initiated for holding are sent first</i>			ServiceInitiated connection initiatingDevice cause localConnectionInfo	B/D3 C2 B/D3 consultation initiated		
			Digits Dialed diallingConnection diallingDevice diallingSequence localConnectionInfo cause	B/D3 C2 B/D3 E Initiated consultation		
			Originated connection callingDevice calledDevice localConnectionInfo cause	B/D3 C2 B/D3 E connected consultation		
			Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDe- vice cause localConnectionInfo	E/D5 C2 E/D5 B/D3 E notSpecified normal connected	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D5 C2 E/D5 B/D3 E notSpecified normal alerting

## Shared-bridged appearances modeling

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
			Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D8 C2 E/D8 B/D3 E notSpecified normal connected	Delivered connection alertingDevice callingDevice calledDevice lastRedirectionDevice cause localConnectionInfo	E/D8 C2 E/D8 B/D3 E notSpecified normal alerting
4. Application invokes transfer of A1 to E  NOTE: transferred connection list with new and old call IDs			<b>Transfer Call Request</b> heldCall activeCall Transfer Call Response transferredCall	B/D3 C1 B/D3 C2 B/D3 C3		
	Transferred primaryOldCall secondaryOldCall transferringDevice transferredToDevice connection list newConnection oldConnection connection list newConnection localConnectionInfo cause	A/D1 C1 Not Present B/D3 E/D8 A/D1 C3 A/D1 C1 E/D8 C3 connected transfer	Transferred primaryOldCall secondaryOldCall transferringDevice transferredToDevice connection list newConnection oldConnection connection list newConnection oldConnection localConnectionInfo cause	B/D3 C1 B/D3 C2 B/D3 E A/D1 C3 A/D1 C1 E/C3 E/C2 Null transfer	Transferred primaryOldCall secondaryOldCall transferringDevice transferredToDevice connection list newConnection oldConnection localConnectionInfo cause	E/D8 C2 Not Present B/D3 E/D8 A/D1 C3 E/D8 C3 E/D8 C2 alerting transfer

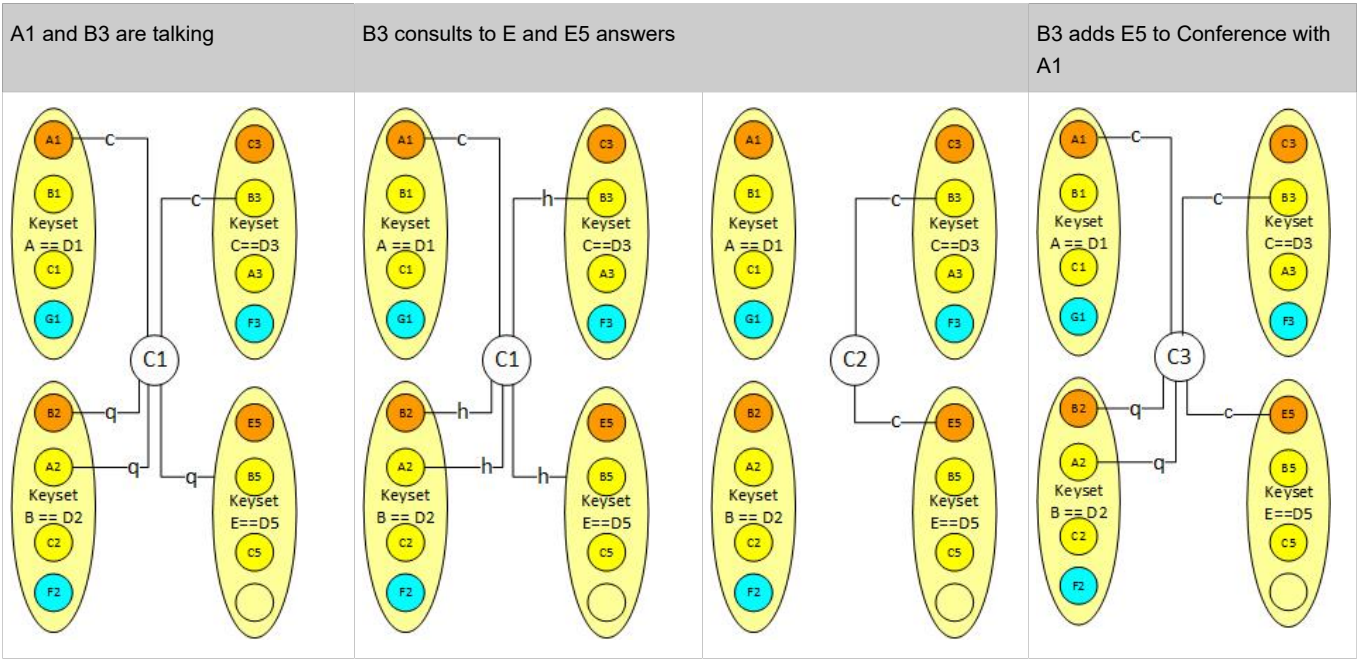
Activity	Monitored Device A		Monitored Device B		Monitored Device E	
	Transferred	A/D1 C1			Transferred	E/D5 C2
	primaryOldCall	Not Present			primaryOldCall	Not Present
	secondaryOldCall	B/D3			secondaryOldCall	B/D3
	transferringDevice	E/D5			transferringDevice	E/D5
	transferredToDevice	A/D1 C3			transferredToDevice	A/D1 C3
	connection list	A/D1 C1			connection list	E/D5 C3
	newConnection	E/D5 C3			newConnection	E/D5 C2
	oldConnection	connected			connection list	alerting
	connection list	transfer			newConnection	transfer
	newConnection				oldConnection	
	localConnectionInfo				localConnectionInfo	
	cause				cause	
5. E/D8 answers	Established	E/D8 C3			Established	E/D8 C3
	connection	E/D8			connection	E/D8
	answeringDevice	A/D1			answeringDevice	A/D1
	callingDevice	E			callingDevice	E
	calledDevice	notSpecified			calledDevice	notSpecified
	lastRedirectionDevice	normal			lastRedirectionDevice	normal
	cause	connected			cause	connected
	localConnectionInfo				localConnectionInfo	
6. A2, A3 and E5 are placed in the Bridged/ Queued state	BridgedEvent	A/D2 C3			BridgedEvent	A/D2 C3
	connection	A/D2			connection	A/D2
	bridedAppearance	normal			bridedAppearance	normal
	cause	queued			cause	connected
	localConnectionInfo				localConnectionInfo	
	BridgedEvent	A/D3 C3			BridgedEvent	A/D3 C3
	connection	A/D3			connection	A/D3
	bridedAppearance	normal			bridedAppearance	normal
	cause	queued			cause	connected
	localConnectionInfo				localConnectionInfo	
	BridgedEvent	E/D5 C3			BridgedEvent	E/D5 C3
	connection	E/D5			connection	E/D5
	bridedAppearance	normal			bridedAppearance	normal
	cause	connected			cause	queued
	localConnectionInfo				localConnectionInfo	

11.5.11 Conference

11.5.11.1 Conference Call Service

- A1 and B3 are talking (Refer to [Sect. 10.5.4.1](#) for the setup event flow)
- B3 consults to E
- E5 answers
- B3 adds E5 to Conference with A1

Table 408: Conference



**Table 409: Conference**

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
1. Application invokes consultation to E on behalf of B3  <i>NOTE: Only Held Event for B3 permits Retrieve Call service</i>			Consultation Call Request  existingCall  consultedDevice  ConsultationCall Response  initiatedCall	B/D3 C1  E  B/D3 C2		
	Held  connection  holdingDevice  cause  localConnectionInfo	B/D3 C1  B/D3  consultation  connected	Held  connection  holdingDevice  cause  localConnectionInfo	B/D3 C1  B/D3  consultation  hold		
	Held  connection  holdingDevice  cause  localConnectionInfo	B/D2 C1  B/D3  consultation  connected	Held  connection  holdingDevice  cause  localConnectionInfo	B/D2 C1  B/D3  consultation  hold		
	Held  connection  holdingDevice  cause  localConnectionInfo	B/D5 C1  B/D3  consultation  connected	Held  connection  holdingDevice  cause  localConnectionInfo	B/D5 C1  B/D3  consultation  hold		
2. Update A/ D2 services permitted.	CallInformationEvent  connection  device  servicesPermitted	A/D2 C1  A/D2  None				
3. B3 setup to D5			ServiceInitiated  connection  initiatingDevice  cause  localConnectionInfo	B/D3 C2  B/D3  consultation  initiated		

## Shared-bridged appearances modeling

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
			Digits Dialed	B/D3 C2		
			diallingConnection	B/D3		
			diallingDevice	E		
			diallingSequence	initiated		
			localConnectionInfo	consultation		
			cause			
			Originated	B/D3 C2		
			connection	B/D3		
			callingDevice	E		
			calledDevice	connected		
			localConnectionInfo	consultation		
			cause			
			Delivered	E/D5 C2	Delivered	E/D5 C2
			connection	E/D5	connection	E/D5
			alertingDevice	B/D3	alertingDevice	B/D3
			callingDevice	E	callingDevice	E
			calledDevice	notSpecified	calledDevice	notSpecified
			lastRedirectionDe- vice	normal	lastRedirectionDevice	normal
			cause	connected	cause	alerting
			localConnectionInfo		localConnectionInfo	
4. E5 answers			Answer Call Request	E/D5 C2		
			CallToBeAnswered			
			Answer Call Response			
			Established	E/D5 C2	Established	E/D5 C2
			connection	E/D5	connection	E/D5
			answeringDevice	B/D3	answeringDevice	B/D3
			callingDevice	E	callingDevice	E
			calledDevice	notSpecified	calledDevice	notSpecified
			lastRedirectionDe- vice	normal	lastRedirectionDevice	normal
			cause	connected	cause	connected
			localConnectionInfo		localConnectionInfo	

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
			BridgedEvent connection bridgedAppearance cause localConnectionInfo	B/D2 B/D2 normal queued	BridgedEvent connection bridgedAppearance cause localConnectionInfo	B/D2 B/D2 normal connected
6. Application invokes conference on behalf of B3.  NOTE: ConenctionList will contain connection IDs with old and new call IDs as well			Conference Call Request  heldCall  activeCall  ConferenceCallRe- sponse conferenceCall	B/D3 C1  B/D3 C2  B/D3 C3		
	Conferenced	A C1*	<b>Conferenced</b>	B C1*	Conferenced	E C2*
	primaryOldCall	Not present	primaryOldCall	B C2*	primaryOldCall	Not present
	secondaryOldCall	B/D3	secondaryOldCall	B/D3	secondaryOldCall	B/D3
	conferencingDevice	E/D5	conferencingDevice	A/D1	conferencingDevice	E/D5
	addedParty	A/D1 C3	addedParty	B/D3 C3	addedParty	B/D3 C3
	connectionlist	A/D1 C1	connectionlist	B/D3 C1	connectionlist	B/D3 C2
	newConnection	B/D3 C3	newConnection	B/D3 C3	newConnection	E/D5 C3
	oldConnection	B/D3 C1	oldConnection	B/D3 C2	oldConnection	E/D5 C2
	connectionlist	E/D5 C3	connectionlist	E/D5 C3	connectionlist	A/D1 C3
	newConnection	connected	newConnection	A/D1 C3	newConnection	connected
	oldConnection	normal	oldConnection	connected	oldConnection	normal
	connectionlist		connectionlist	normal	connectionlist	
	newConnection		newConnection		newConnection	
	localConnectionInfo		connectionlist		localConnectionInfo	
	cause		newConnection localConnectionInfo cause		cause	

## Shared-bridged appearances modeling

Activity	Monitored Device A		Monitored Device B		Monitored Device E	
7. In OS4K a bridge may consist of 3 parties only, therefore BridgedEvent is generated after conference with empty services permitted (no bridging in is possible)	<b>BridgedEvent</b> connection bridgedAppereance cause localConnectionInfo	A/D2 C3 A/D2 normal queued				
	<b>BridgedEvent</b> connection bridgedAppereance cause localConnectionInfo	B/D2 C3 B/D2 normal connected	<b>BridgedEvent</b> connection bridgedAppereance cause localConnectionInfo	B/D2 C3 B/D2 normal queued	<b>BridgedEvent</b> connection bridgedAppereance cause localConnectionInfo	B/D2 C3 B/D2 normal connected

## 12 Glossary

### A

#### **ACD**

Automatic call distribution.

#### **ACD calls**

An ACD call is an incoming call that reaches an ACD number. If an incoming call arrives on a trunk group dedicated to an ACD number, it immediately becomes an ACD call and begins to be routed to the system. However, if an incoming call arrives on a trunk that is not dedicated to an ACD number, the system does not route the call until it does reach an ACD number—for example, if the call is internally transferred.

#### **ACD group**

A group of ACD agents responsible for servicing a particular type of call (such as a brokerage call, a bank loan call or a call to an airline for a reservation). An ACD group can be a single extension, such as customer service. Larger departments can also be divided into many smaller ACD groups. See automatic call distribution.

#### **ACD number**

A dialable number that initiates processing of the call as an ACD call. The ACD number maps the call to a RCG depending on the source of the call (ANI), the destination of the call (DNIS), the day of the week, and the time of day.

#### **ACD RCG**

See RCG.

#### **ACD RCG monitor group**

All ACD RCG are pre-configured in the OpenScape 4000 Communications Server. The reserve number that is reserved for this monitor group is \*888. The Monitor Start using this number as its monitor object "activates" this group monitor, as if the communications server had received individual Monitor Starts for each RCG in the group.

#### **ACD Routing Table (ART)**

Tables that permit the configuration of the call routing. An ACD routing table is a set of instructions that an ACD call follows until an agent is available to answer the call. For example, the caller might first hear a recorded message stating that all agents are still busy, followed by music for a certain number of seconds. If an agent is still not available, the call might be routed to another group of agents or even to an off-site number. Many routing tables can be configured for each ACD group. This permits customized routing to meet each group's requirements.

#### **ACL**

Application connectivity link.

### **ACL Communication Manager**

Software residing in a OpenScape 4000 Communications Server that allows communication between the CallBridge server and ACL applications.

### **agent**

A customer service person who uses an agent workstation to make or receive calls from customers.

### **agent workstation**

A workstation consisting of a telephone connected to the OpenScape 4000 Communications Server and a terminal/PC connected to a host computer or client-server LAN.

### **answer call**

A service that answers a call alerting device.

### **ANI**

Automatic Number Identification.

### **API**

Application Program Interface.

### **ACD Prompt Response Integration (APRI)**

ACD Prompt Response IVR server can obtain ACD queue-related information from the OpenScape 4000 to aid in routing decisions. The collected data is passed through the OpenScape 4000 to attached applications like CallBridge for reporting on caller or call-related information.

### **Application Connectivity Link (ACL)**

The Unify proprietary communications link that connects the OpenScape 4000 to the CallBridge server. The ACL transports request and event messages between the CallBridge server and the OpenScape 4000.

### **Application Program Interface (API)**

The software through which the host computer commands the OpenScape 4000 Communications Server to perform telephony functions such as monitoring the start, stop, or transfer of calls.

### **application vendor**

A company that sells the application programs that run on the host/server computer to which the OpenScape 4000 Communications Server is connected.

### **ART**

See ACD routing table.

### **automated outbound dialing**

A feature that allows a host application to place a call to a customer on behalf of an agent.

**Automatic Call Distribution (ACD)**

A system feature that allows a high volume of incoming calls, arriving on dedicated trunks, to be distributed efficiently based on preset algorithms.

**Automatic Number Identification (ANI)**

A feature that works with the digital public network to identify outside callers to users of the OpenScape 4000 Communications Server. ANI allows agents connected to the OpenScape 4000 to view information about the caller so they can answer the call intelligently. This is useful when calls are regularly received from customers that deserve special treatment, for instance, large accounts. When they call, the ACD system and CallBridge can be used to bring up their records on the answering agent's terminal before the call is answered.

**C****call**

Any connection made between two or more parties, such as the connection made between an inbound trunk and an extension or between two or more extensions.

**call center**

A customer business center where initial access is by telephone. Employees of call centers often have a terminal to access databases for needed information. Modern call centers often embrace other media such as the Internet.

**call handling services**

Services that allow the host application to activate an agent's request for: make call, clear connection, consultation call, transfer call, answer call, and so on.

**clear connection**

A service that releases a specific device from a call.

**Communications Server (CS)**

A computerized switching system providing telephone communications between internal stations, and between internal stations and external telephone networks.

**Computer-Telephony Integration (CTI)**

A generic term describing the integration of computers with telephony equipment.

**conference call**

A call in which three or more devices are connected together.

**connection state**

This is the condition that exists for a party in a transaction that remains off-hook after the other party has gone back on-hook. The party remaining off-hook must go into an idle state before receiving dial tone, or before making another call.

### **consultation call**

(1) A call that allows you to talk privately with a second party while the first party is on hold. (2) A service that places an existing active call at a device on soft hold and initiates a new call from the same device.

### **coordinated voice and data transfer**

A feature that allows simultaneous voice and data transfer when a call is transferred from one agent to another.

### **D**

### **Dialed Number Identification Service (DNIS)**

A network service that provides information on the number dialed. Calls coming in to a call center may contain identifying information. The telephone number customers dial can be used to route calls. If, for example, a company has agents taking calls for life insurance for seniors as well as for a sports magazine subscription, it can help the agents answer these callers differently. If separate telephone numbers are provided for these two products, the ACD system, using DNIS and CallBridge, can inform the agents which type of customer is calling before the agent answers the call.

### **Directory Number (DN)**

A dialable number.

### **E**

### **event stream**

Call tracking information the OpenScape 4000 Communications Server generates and sends messages to the application. Applications use this information to track calls and determine agent availability and to support features such as intelligent answering and coordinated voice and data transfer.

### **extension**

A unique number assigned to a telephone station that is connected to the OpenScape 4000 Communications Server. All connections for any extension number are switched through the OpenScape 4000.

### **F**

### **forwarding**

The feature that redirects an incoming call to another extension or to an off-site telephone.

### **H**

### **OpenScape 4000 Communications Server**

Unify Communications private branch exchange (PBX) providing telephone communications between internal stations and between internal stations and external telephone networks.

### **hold**

A line is on hold when it is in use, but in a waiting state (no active connection).

**holding time**

The length of time for which a particular call occupies a communication channel.

**application environment**

The CSTA host computer or local area network to which the OpenScape 4000 Communications Server is connected via the CallBridge CTI link.

**host link**

The communications link that connects the CallBridge for CSTA server to the host environment (for example, RS/6000 or AS/400).

I

**idle agent queue**

A series of idle ACD agents waiting for incoming calls. See queue.

**Prompt Response IVR (with APRI) Server**

A digital IVR with special capabilities used in conjunction with ACD.

**Prompt Response IVR (with APRI) Server port**

A CorNet-N channel when connected to the Prompt Response IVR (with APRI) Server is referred to as a server port.

**intelligent answering**

A feature that allows the host application to display business application or customer data on the screen of the agent receiving an incoming call or placing an outgoing call.

**Interactive Voice Response (IVR)**

See voice response unit.

**internal call**

A call between two extensions, or between an extension and an operator within a single PBX.

L

**lines**

Lines are circuits that connect calls from the OpenScape 4000 Communications Server to telephones, terminals, printers, recording devices, and attendant consoles.

O

**off-hook**

The condition when the telephone device is first activated, typically when the handset is lifted from the telephone cradle to get dial tone.

**on-hook**

The condition when the telephone device is deactivated, typically when the handset is replaced on the telephone cradle at the end of a call.

### P

#### **performance data**

Diagnostic data, stored in a buffer, that allows you to analyze system performance based on traffic data collected during a specified time frame.

#### **profile**

A set of parameter values that allow you to customize your software; for example, you can specify the password you enter to access the system.

### Q

#### **queue**

A series of calls waiting for an available ACD agent, a computer port, a modem, or data group access.

### R

#### **RCG monitor group**

See ACD RCG monitor group.

#### **Route Control Group (RCG)**

A table that appears only in software (not assigned to a physical channel) that is used to access a group of telephones. Route Control Groups are used for hunt, distribution, ACD, control, and security groups.

#### **Routing Services**

The CallBridge routing features that allow a computer system application to influence the routing of an automatic call distribution (ACD) call before it is routed by the OpenScape 4000 ACD applications.

### S

#### **single-step transfer (SST)**

This service transfers (in a single step) a call to a new device, thereby combining the services of Consultation and Transfer into one service.

#### **SLES**

SUSE Linux Enterprise Server

#### **station**

Any building location that is wired to accept a telephone.

#### **system administration**

The process of setting up, monitoring, and maintaining the telephone system as done by the system administrator.

#### **system message**

A message, stored in a buffer, that records an internal system event, such as an error, configuration change, or program reload.

### T

**telephony application**

An application program running on a host computer or server that directly or indirectly performs telephony functions such as dialing, answering, transferring, or management of voice or data calls.

**transfer call**

A service that transfers a held call to an active call at the same device.

**trunks**

Trunks connect the OpenScape 4000 Communications Server to the public telephone network and connect one OpenScape 4000 to another. The actual transmission method can be digital or analog.

V

**Voice Response Unit (VRU)**

Hardware or software that receives incoming calls by playing one or more prerecorded messages. The messages may require the caller to give additional information by touching buttons on a touch-tone telephone.

# Index

## A

- ACD Queuing [396](#)
- ACD Routing Flow Diagram [397](#)
- ADR
  - see alternate destination redirection [22](#)
- alternate destination redirection [22](#)
- ANI
  - see automatic number identification [20](#)
- automated outbound dialing [27](#)

## C

- Call Origination Scenarios [315](#)
- Call Scenarios [270](#)
  - Call Origination Scenarios [315](#)
  - Consultation Call Scenarios [344](#)
- CallBridge for CSTA
  - business applications
    - automatic number identification [20](#)
    - coordinated voice and data transfer [22](#)
    - dialed number identification service [20](#)
    - intelligent answering [20](#)
- CCparmsNotSupported [38](#), [100](#), [145](#)
- Consultation Call Scenarios [344](#)
  - Held Party Releases [354](#)
  - Successful consultation Call [344](#)
- coordinated voice and data transfer [22](#)

## D

- DNIS
  - see dialed number identification service [20](#)

## G

- General Attendant
  - Diversion to GA fails [444](#), [444](#)
  - Intercept [442](#)
  - Night-service [442](#)
  - Personal calls [443](#)
  - Recalls to GA [443](#)

## H

- Held Party Releases [354](#)
- Hunting Groups (HG)
  - ACD routes MPC-call into HG [424](#)
  - Called device [424](#)
  - Deflect [423](#)
  - Known Restrictions [424](#)
  - Special Features [423](#)

## R

- Route Processing [395](#)
- routing dialogs
  - reject call [24](#)

## S

- services
  - summary of [31](#)
- Successful consultation Call [344](#)

