ecma INTERNATIONAL

Technical Report

ECMA TR/82

2nd edition / June 2009

Scenarios for Computer Supported Telecommunications Applications (CSTA) Phase III

Rue du Rhône 114 CH-1204 Geneva T: +41 22 849 6000 F: +41 22 849 6001

is the registered trademark of Ecma International

COPYRIGHT PROTECTED DOCUMENT

Contents

Page

Introduction

This Technical Report provides example call scenarios based upon Phase III of Services for Computer Supported Telecommunications Applications (CSTA). This Technical Report is part of a Suite of Standards and Technical Reports for Phase III of CSTA. All of the Standards and Technical Reports in the Suite are based on practical experience of ECMA member companies and each one represents a pragmatic and widely-based consensus.

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

This 2nd edition of TR/82 provides advanced conferencing scenario examples that illustrate typical flows involving devices that are designed to host conference calls with a large number of participants.

This Ecma Technical Report has been adopted by the General Assembly of June 2009.

Scenarios for Computer Supported Telecommunications Applications (CSTA) Phase III

1 Scope

This Technical Report illustrates call scenarios for Services for Computer Supported Telecommunications Applications (CSTA) Phase III (ECMA-269).

The scenarios are only for information and as such the ECMA-269 Standard may define additional options or parameters. The purpose of this Technical Report is to provide examples of some CSTA Service invocations and illustrate associated call event reports. It is not an exhaustive document and some implementations may not perform as illustrated within this document, while still conforming to the Standard.

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 - call type monitors have not been illustrated. 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.

All mandatory parameters is CSTA messages are provided. In addition, all conditional parameters that are required in the context of the scenario are provided. Optional parameters are generally not included unless they are useful in the context of illustrating a specific scenario. The mandatory, conditional, and optional classification of parameters in CSTA messages are specified in ECMA-269.

The monitorCrossRefID parameter in events is not shown.

DeviceIDs are illustrated by Dn and ConnectionIDs in the form DnCn. 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.

2 Normative references

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

ECMA-269, Services for Computer Supported Telecommunications Applications (CSTA) Phase III, 8th edition (June 2009)

ECMA TR/72, Glossary of Definitions and Terminology for Computer Supported Telecommunications Applications (CSTA) Phase III, 3rd edition (June 2000)

3 Terms and definitions

The definitions and abbreviations used in this Technical Report are defined in ECMA TR/72.

4 Call Origination Scenarios

This clause includes examples of how calls can be initiated, either by using the Make Call service or by manual operation.

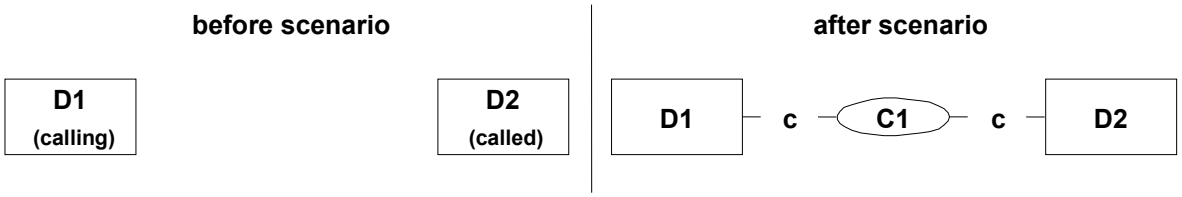
The first flow illustrates how a call is originated using the Make Call service. In this flow the calling device is prompted to go off-hook, and then the call is established between two devices.

Additional call origination flows are provided that illustrate how to originate a call using the Make Call service with hands free dialling, manual dialling, multi-stage dialling, and scenarios that show calls that fail, etc.

4.1 Make Call service - calling device is prompted to go off-hook

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.

In this scenario the Make Call service specifies that device D1 should be prompted to go off-hook (via the autoOriginate parameter) before D2 device is called.



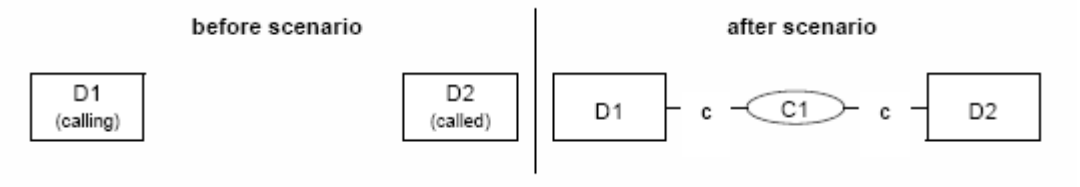
Activity	Monitored Device D1	Monitored Device D2		Comments
A Make Call service to a valid device is invoked on behalf of device D1.	MakeCallRequest • callingDevice D1 • calledDirectoryNumber D2 • autoOriginate Prompt			The Make Call service specifies that device D1 should be prompted to go off-hook.
Acknowl edgement.	MakeCallResult • initiatedCall D1C1			
Indication that the service has been initiated from this device.	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatedDevice D1 • localConnectionState Initiated • cause makeCall			The generation of this event is switch specific. The MakeCall cause indicates that the device D1 is being prompted (via ringing, for example) to go off-hook.
Device D1 goes off hook and is connected in the call.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice D2 • localConnectionState Connected • cause newCall			
Device D2 begins to ring and D1 receives ringing tone.	DeliveredEvent • connection D2C1 • alertingDevice D2 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice NR • localConnectionState Connected • cause newCall	DeliveredEvent • connection D2C1 • alertingDevice D2 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice NR • localConnectionState Alerting • cause newCall		
Device D2 answers the call by manually going off-hook.	EstablishedEvent • establishedConnection D2C1 • answeringDevice D2 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice NR • localConnectionState Connected • cause newCall	EstablishedEvent • establishedConnection D2C1 • alertingDevice D2 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice NR • localConnectionState Connected • cause newCall		

4.2 Make Call service - calling device is in hands free mode

This scenario illustrates the case when the calling device is requested to automatically connect to the call ("hands free" mode).

This scenario differs from the first scenario in the following ways:

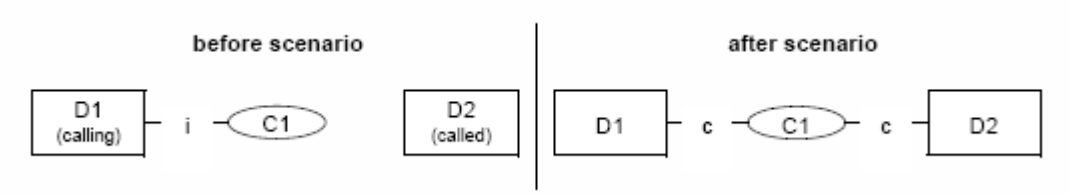
- The Make Call service request (via the autoOriginate parameter) specifies that the calling device should be automatically connected to the call ("hands free") mode.
- The NewCall cause on the Service Initiated event indicates that the device is not being prompted to go off-hook.



Activity	Monitored Device D1	Monitored Device D2		Comments
A MakeCall service to a valid device is invoked on behalf of device D1.	MakeCallRequest • callingDevice D1 • calledDirectoryNumber D2 • autoOriginate DoNotPrompt			The autoOriginate parameter specifies that the calling device should be automatically connected to the call (not prompted).
Acknowl-edgement.	MakeCallResult • initiatedCall D1C1			
Indication that the service has been initiated from this device.	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatedDevice D1 • localConnectionState Initiated • cause NewCall			The generation of this event is switch specific. The Service Initiated event with the NewCall cause indicates that there is no prompting at the device.
Device D1 is automatically connected to the call.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice D2 • localConnectionState Connected • cause newCall			Since the autoOriginate parameter indicates "DoNot-Prompt", device D1 is connected to the call without manual inter-vention (hands free mode).
	...scenario proceeds as shown in 4.1.			

4.3 Make Call service - calling device is already off-hook

This scenario illustrates the invoking of a call that already was initiated by a user going off-hook on a telephone. The call is invoked from device D1 to device D2.



Activity	Monitored Device D1	Monitored Device D2		Comments
Device D1 manually initiates a call by going off-hook.	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatedDevice D1 • localConnectionState Initiated • cause newCall			
MakeCall service is invoked on behalf of device D1.	MakeCallRequest • callingDevice D1 • calledDevice D2			
Acknowl-edgement.	MakeCallResult • initiatedCall D1C1			
Call proceeds from device D1.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice D2 • localConnectionState Connected • cause newCall			
	...scenario proceeds as shown in 4.1.			

4.4 Manually dialled call

This scenario illustrates a call originated through manual device activity.

The scenario differs from the first scenario in the following ways:

- The Make Call service is not included.
- The cause on the Service Initiated event does indicate prompting.

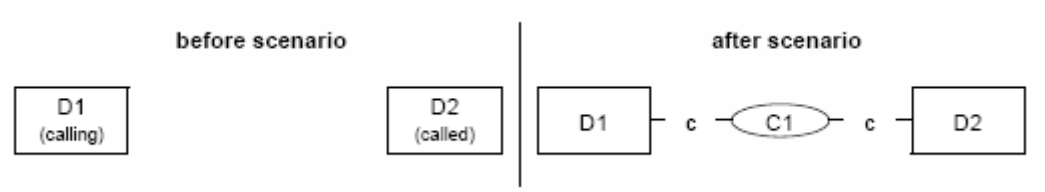
Note that in this scenario the implementation buffers the dialled digits until the complete dialling sequence has been dialled and provides the complete dialled digits in the Originated event (i.e., no Digits Dialled event(s) are provided in this scenario).

Activity	Monitored Device D1		Monitored Device D2		Comments
User at Device D1 goes off-hook and receives dial tone.	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • cause	 D1C1 D1 Initiated newCall			
Device D1 completes dialling device D2 and is connected to the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause	 D1C1 D1 D2 Connected newCall			
	...scenario proceeds as shown in 4.1.				

4.5 Manually dialled call showing individual digits dialled

This scenario differs from the previous scenarios because it illustrates how an individual Digits Dialled event is generated for each digit dialled. After all digits are dialled the Originated event provides the complete dialled sequence.

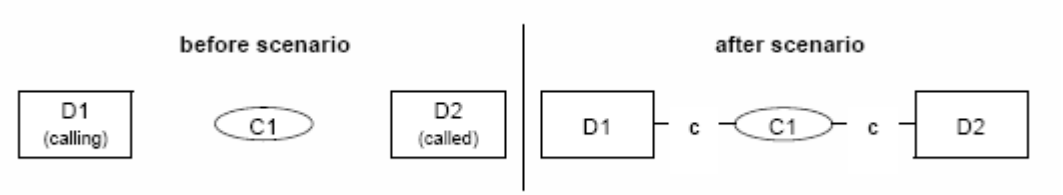
Note that it is implementation specific how many digits are buffered before they are sent in a Digits Dialled event, or if the digits are buffered until the complete sequence of digits is dialled (i.e., no Digits Dialled events prior to an Originated event).



Activity	Monitored Device D1	Monitored Device D2		Comments
User at Device D1 goes off-hook and receives dial tone.	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatingDevice D1 • localConnectionState Initiated • cause newCall			
Digit "2" is dialled.	DigitsDialledEvent • diallingConnection D1C1 • diallingDevice D1 • diallingSequence "2" • localConnectionState Initiated • cause normal			Digit "2" is dialled.
Digit "3" is dialled.	DigitsDialledEvent • diallingConnection D1C1 • diallingDevice D1 • diallingSequence "3" • localConnectionState Initiated • cause normal			Digit "3" is dialled.
Digit "4" is dialled.	DigitsDialledEvent • diallingConnection D1C1 • diallingDevice D1 • diallingSequence "4" • localConnectionState Initiated • cause normal			Digit "4" is dialled.
Digit "3" is dialled.	DigitsDialledEvent • diallingConnection D1C1 • diallingDevice D1 • diallingSequence "3" • localConnectionState Initiated • cause normal			Digit "3" is dialled.
Device D1 has completed dialling and is connected to the call.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice D2 • localConnectionState Connected • cause newCall			All of the digits for D2 ("2343") have been dialled. The Originated event contains the complete dialling sequence.
	...scenario proceeds as shown in 4.1.			

4.6 Dialling using Dial Digits service

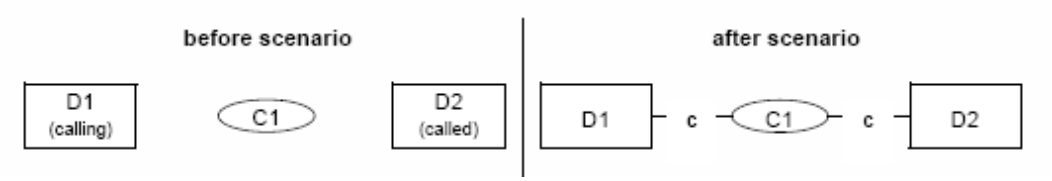
This scenario illustrates the use of the Dial Digits service for a call that has already been established by the user manually going off-hook.



Activity	Monitored Device D1	Monitored Device D2		Comments
Device D1 manually initiates a call by going off-hook.	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatedDevice D1 • localConnectionState Initiated • cause newCall			
The Dial Digits service with the complete dialling sequence is provided.	DialDigitsService • diallingConnection D1C1 • diallingSequencee D2			
Acknowl-edgement.	DialDigitsResult			
The event indicates that the requested digits were dialled.	DialDigitsEvent • diallingConnection D1C1 • diallingDevice D1 • diallingSequence D2 • localConnectionState Initiated • cause normal			
The dialling sequence is complete and device D1 is connected in the call.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice D2 • localConnectionState Connected • cause newCall			
	...scenario proceeds as shown in 4.1.			

4.7 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.



Activity	Monitored Device D1		Monitored Device D2		Comments
A MakeCall service to a valid device is invoked on behalf of device D1.	MakeCallRequest • callingDevice • calledDirectoryNumber • autoOriginate	D1 "2;" Prompt			The Make Call service specifies a partial dialling string that begins with a ("2") and the partial dialling indicator (";").
Acknowl-edgement.	MakeCallResult • initiatedCall	D1C1			
Indication that the service has been initiated from this device.	ServiceInitiatedEvent • initiatedConnection • initiatedDevice • localConnectionState • cause	D1C1 D1 Initiated makeCall			The generation of this event is switch specific. The MakeCall cause indicates that the device D1 is being prompted (via ringing, for example) to go off-hook.
The event indicates that a partial dialling sequence was received.	DialDigitsEvent • diallingConnection • diallingDevice • diallingSequence • localConnectionState • cause	D1C1 D1 "2;" Initiated normal			A ";" character indicates that there is an incomplete dialling string.
The Dial Digits service with the remainder of the dialling sequence is provided.	DialDigitsService • diallingConnection • diallingSequencee	D1C1 "3456"			A ";" is not provided in the dialling string since there are no more digits to be dialled.
Acknowl-edgement.	DialDigitsResult				
The event indicates that the requested digits were dialled.	DialDigitsEvent • diallingConnection • diallingDevice • diallingSequence • localConnectionState • cause	D1C1 D1 "3456" Initiated normal			A complete dialling sequence has been received (no ";" character).
The dialling sequence is complete and device D1 is connected in the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause	D1C1 D1 D2 Connected newCall			D2 is the called device. It contains the digits "23456" in this scenario.
	....scenario proceeds as shown in 4.1.				

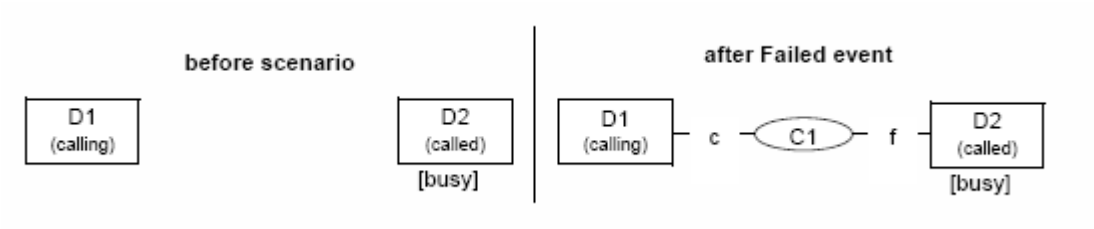
4.8 Make Call service - called device is busy

This scenario illustrates a Make Call from device D1 to device D2, where device D2 is busy and is not set to forward busy calls. The call fails because the called party is busy.

This scenario differs from the first scenario in the following ways:

- The Failed event is generated to indicate that the call has encountered a busy device.

Note that in this example the Make Call service is successful (positive acknowledgement) and the events indicate that device D2 is busy. Another possible scenario is where the Make Call service is negatively acknowledged with an error code indicating that device D2 is in an invalid state.



Activity	Monitored Device D1		Monitored Device D2			Comments
MakeCall service is invoked on behalf of device D1.	MakeCallRequest • callingDevice • calledDirectoryNumber • autoOriginate	 D1 D2 Prompt				
Acknowl-edgement.	MakeCallResult • initiatedCall	 D1C1				
Device D1 notified of initi-ating call.	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • cause	 D1C1 D1 Initiated MakeCall				The generation of this event is switch specific.
The call is being attempted from device D1.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause	 D1C1 D1 D2 Connected newCall				
Device D2 is busy - the call cannot be completed. Device D1 hears busy tone.	FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D2C1 D2 D1 D2 NR Connected busy	FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D2C1 D2 D1 D2 NR Failed busy		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. See ECMA-269, clause 2.8.2, item 2.
Device D1 replaces handset.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Failed normalClearing		
Failed connection D2C1 also clears.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D2C1 D2 Null normalClearing		

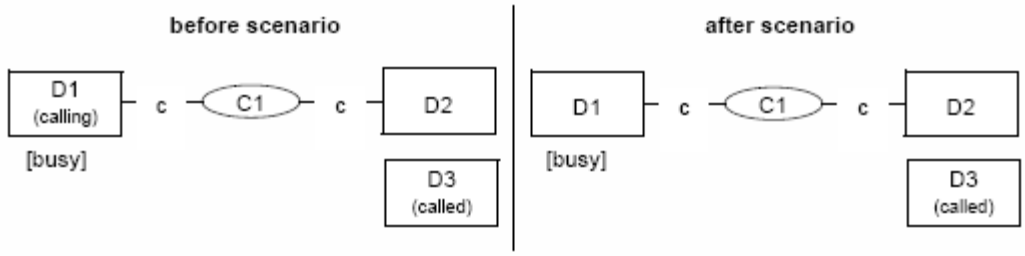
4.9 Make Call service - call attempted to a busy calling device (negative acknowledgement)

This scenario illustrates a Make Call from device D1 to device D3, where device D1 is busy. The call fails because the calling device is busy.

This scenario differs from the first scenario in the following ways:

- The (negative) response to the Make Call service request indicates that the call attempt has failed. No subsequent events are generated.

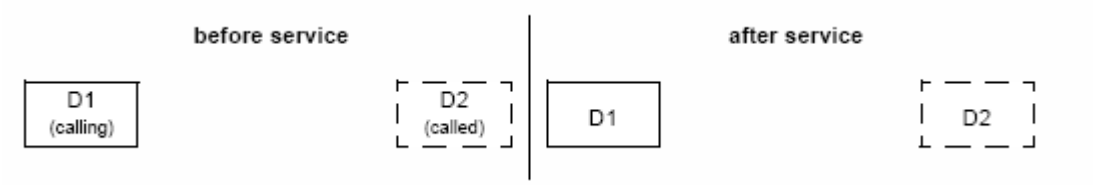
The Make Call request is negatively acknowledged because the calling party, device D1, is busy when a Make Call request is issued.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
A MakeCall service is invoked on behalf of device D1.	MakeCallRequest • callingDevice D1 • calledDevice D3			
Negative Acknowl-edgement.	MakeCallError • stateIncompatibility invalid calling device state			Make Call service fails because calling party is busy.

4.10 Make Call service - called number is an invalid number (negative acknowledgement)

This scenario illustrates a Make Call from device D1 to device D2. In this scenario device D1 is available, valid and permitted to make the call. Device D2 (illustrated by a box with a dotted line around it) is actually an invalid number (e.g., the number is correctly formatted but it is not part of the dialling plan) and the call fails.

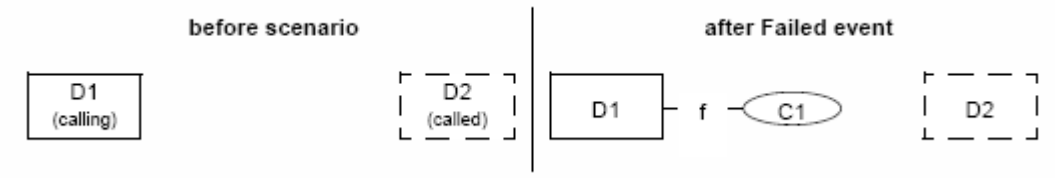


Activity	Monitored Device D1	Monitored Device D2		Comments
MakeCall service is invoked on behalf of device D1.	MakeCallRequest • callingDevice D1 • calledDevice D2			
Negative Acknowl-edgement.	MakeCallError • operationalError InvalidCalled-Device			

4.11 Manually dialled call - dialled number is invalid

This scenario illustrates a manually dialled call from device D1 to device D2. In this scenario device D1 is available, valid and permitted to make the call. Device D2 (illustrated by a box with a dotted line around it) is actually an invalid number (e.g., the number is correctly formatted but it is not part of the dialling plan) and the call fails.

Note that in this scenario the dialled digits are buffered in the switching function until the complete dialling sequence has been dialled and is providing the complete dialled digits in the Originated event (i.e., no Digits Dialled event(s) are provided in this scenario).



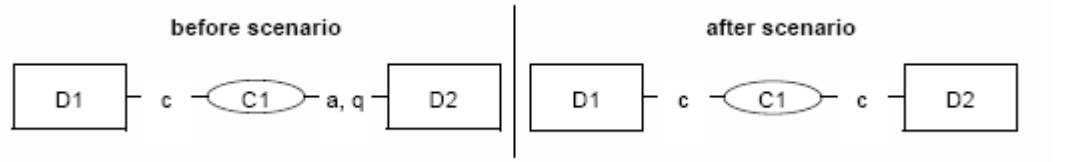
Activity	Monitored Device D1		Monitored Device D2		Comments
Device D1 goes off-hook and receives dial tone.	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • cause	 D1C1 D1 Initiated newCall			
Device D1 completes dialling device D2 and is connected to the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause	 D1C1 D1 D2 Connected newCall			
Device D2 is an invalid number the call cannot be completed. Device D1 receives reorder tone.	FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D1C1 D1 D1 D2 NR Failed reorderTone			
Device D1 goes on-hook.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Null normalClearing			

5 Answering Call Scenarios

This clause illustrates how calls are answered, manually and by CSTA services.

5.1 Answer Call service

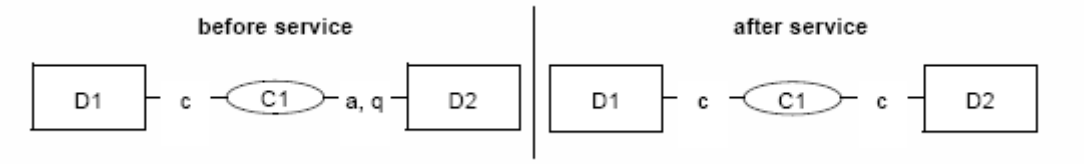
This scenario illustrates the successful use of the Answer Call service invoked on behalf of device D2.



Activity	Monitored Device D1		Monitored Device D2			Comments
Answer Call service is invoked on behalf of device D2.			AnswerCallRequest • callToBeAnswered		D2C1	An error will be returned if the device is not able to answer the call without manual intervention.
Acknowl-edgement.			AnswerCallResult			
Device D2 has been answered.	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D2C1 D2 D1 D2 NR Connected newCall	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D2C1 D2 D1 D2 NR Connected newCall		

5.2 Manually answering a call

This scenario illustrates the event sequence when alerting device D2 goes off-hook to answer the call.



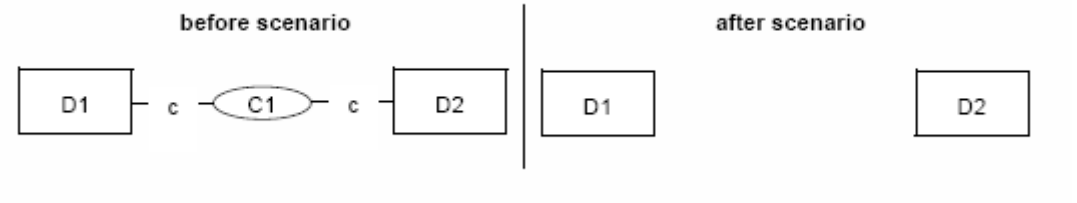
Activity	Monitored Device D1		Monitored Device D2			Comments
Device D2 answers the call by going off-hook.	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D2C1 D2 D1 D2 NR Connected newCall	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D2C1 D2 D1 D2 NR Connected newCall		

6 Call and Connection Termination Scenarios

This clause illustrates how calls and connections are ended.

6.1 Device disconnects from a call by going on-hook (remaining device is cleared from the call)

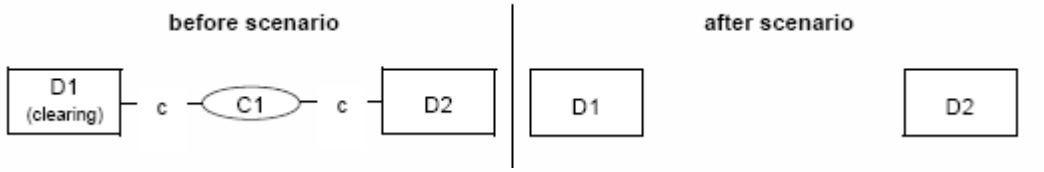
The user at device D1, while connected to device D2, replaces the handset.



Activity	Monitored Device D1		Monitored Device D2			Comments
D1 goes on-hook.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Connected normalClearing		
Since D2 is the only device in the call, it is cleared as the result of D1 being cleared.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D2C1 D2 Null normalClearing		

6.2 Device disconnects from a call by going on-hook (remaining device goes blocked)

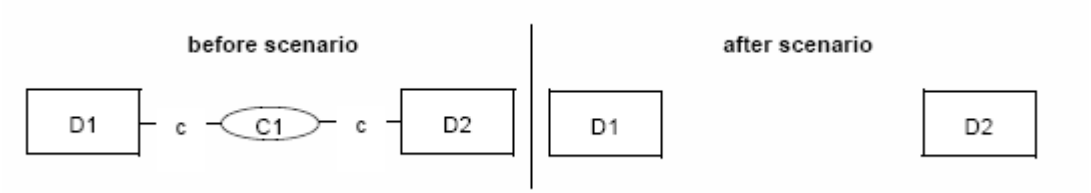
In this scenario device D1 is manually put on-hook to release itself from the call. The remaining device goes blocked, until the device goes on-hook.



Activity	Monitored Device D1		Monitored Device D2			Comments
Device D1 goes on-hook.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Connected normalClearing		
As a result of the "far end disconnect", the remaining connection D2C1 goes blocked.			FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D2C1 D2 D1 D2 NR Failed blocked		While D2C1 is blocked, the user at the device may be hearing error tone.
The remaining device goes on-hook.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D2C1 D2 Null normalClearing		

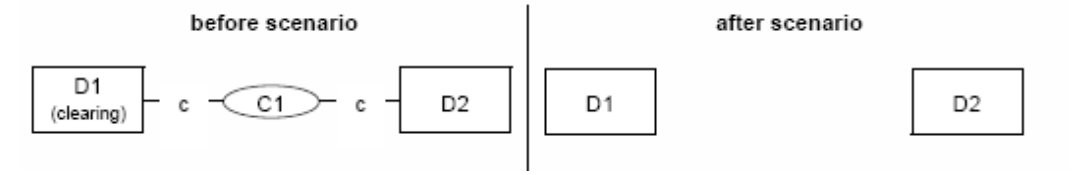
6.3 Device disconnects from a call using the Clear Connection service (remaining device is cleared)

The Clear Connection service is used to disconnect device D1 from the call. In this example, the remaining device in the call, D2, is cleared after D1 has been removed from the call.



Activity	Monitored Device D1		Monitored Device D2			Comments
A Clear Connection service is invoked on D1s behalf.	ClearConnectionRequest • connectionToBeCleared	 D1C1				
Acknowl-edgement.	ClearConnectionResult					
Event indicates that connection D1C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Connected normalClearing		
Since D2 is the only device in the call, it is cleared as the result of D1 being cleared.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D2C1 D2 Null normalClearing		

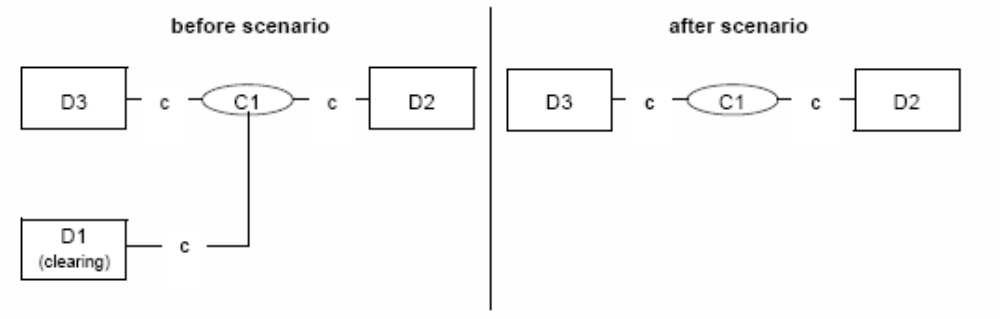
6.4 Device disconnects from a call using the Clear Connection service (remaining device goes blocked)



Activity	Monitored Device D1		Monitored Device D2			Comments
A Clear Connection service is invoked on D1s behalf.	ClearConnectionRequest • connectionToBeCleared	D1C1				
Acknowl-edgement.	ClearConnectionResult					
Event indicates that connection D1C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Connected normalClearing		
As a result of the "far end disconnect", the remaining connection D2C1 goes blocked.			FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	D2C1 D2 D1 D2 NR Failed blocked		While blocked, the user at the device may be hearing error tone.
The remaining device goes on-hook.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D2C1 D2 Null normalClearing		

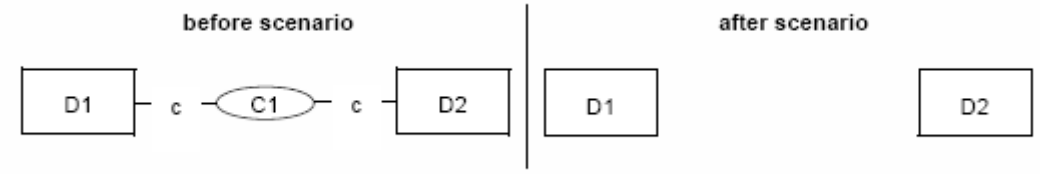
6.5 Device disconnects from a conference call using the Clear Connection service

This service releases a specific device from a call. In the case of a Conference Call this results in the specific party being removed from this conference.



Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
A Clear Connection service is invoked.	ClearConnectionRequest • connectionToBeCleared	D1C1					
Acknowl-edgement.	ClearConnectionResult						
Event indicates that connection D1C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Connected normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Connected normalClearing	Devices D2 and D3 remain connected in the call.

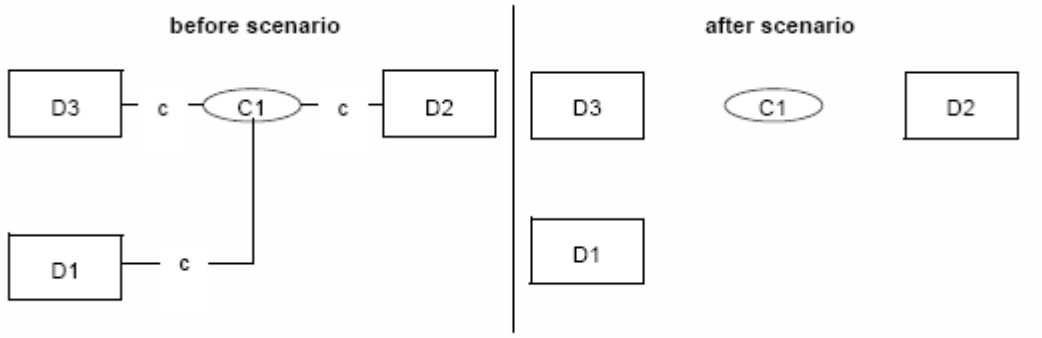
6.6 Clearing a two-party call using the Clear Call service



Activity	Monitored Device D1		Monitored Device D2			Comments
A Clear Call service is invoked.	ClearCallRequest • callToClear	D1C1				
Acknowl-edgement.	ClearCallResult					
The events indicate that D1 has discon-nected from the call.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Connected normalClearing		
The events indicate that D2 has discon-nected from the call.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D2C1 D2 Null normalClearing		

6.7 Clearing a conference call using the Clear Call service

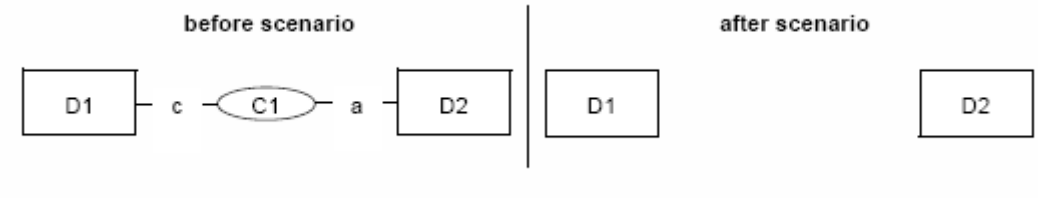
This scenario illustrates the use of the Clear Call service. This service releases all devices from an existing conference call.



Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
A Clear Call service is invoked.	ClearCallRequest • callToClear	D1C1					
Acknowl-edgement.	ClearCallResult						
D1s connection to the call is cleared.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Connected normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Connected normalClearing	D1C1s Connection Cleared event is reported for all device-type monitors in the call.
D2s connection to the call is cleared.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D2C1 D2 Null normalClearing	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D2C1 D2 Connected normalClearing	D2C1s Connection Cleared event is reported for all devices remaining in the call.
The remaining connection is cleared.					ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D3C1 D3 Null normalClearing	

6.8 Call is cleared after an alerting time-out

In this scenario, the call is cleared as the result of an alerting timer expiry.



Activity	Monitored Device D1		Monitored Device D2			Comments
D2C1 is cleared as the result of an alert timer expiry.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D2C1 D2 Connected callNotAn-swered	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D2C1 D2 Null callNotAn-swered		The cause of "callNotAnswered" indicates that the call was cleared as the result of a timer expiry.
Connection D1C1 clears as a result of D1 clearing.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Null normalClearing				

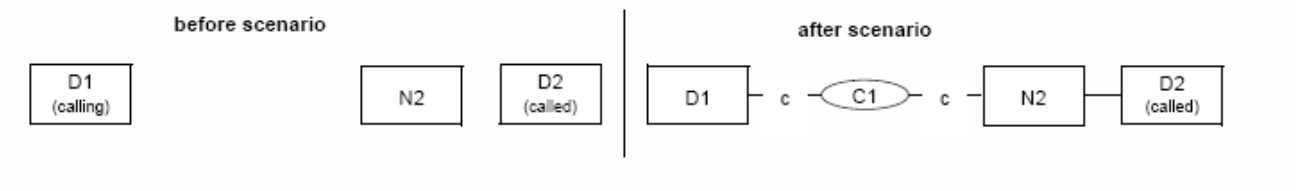
7 External Outgoing Call Scenarios

This clause includes examples of successful external outgoing calls, initiated manually and by CSTA services.

7.1 Make Call service - called device is outside the CSTA sub-domain

This scenario illustrates a Make Call service request on behalf of device D1 to the device D2 which is outside the CSTA sub-domain.

Since device D2 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 N2, which is a network interface device (NID) (e.g., trunk interface), acts as a proxy for device D2, and depending upon the type of signalling available via the external network, some signalling information can be made available through the connection of the NID.



Activity	Monitored Device D1		Monitored Device N2		Comments
A Make Call service to a valid device outside the CSTA sub-domain is invoked on behalf of device D1.	MakeCallRequest • callingDevice • calledDevice • autoOriginate	 D1 D2 doNotPrompt			The service request specifies "hands free" mode (See 4.2).
Acknowl-edgement.	MakeCallResult • initiatedCall	 D1C1			
Indication that service has been initiated from this device.	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • cause	 D1C1 D1 Initiated newCall			The generation of this event is switch specific.
D1 is connected to the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause	 D1C1 D1 D2 Connected newCall			

Activity	Monitored Device D1		Monitored Device N2			Comments
The call leaves the CSTA sub-domain.	NetworkReachedEvent • outboundConnection • networkInterfaceUsed • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 N2C1 N2 D1 D2 NR Connected newCall	NetworkReachedEvent • outboundConnection • networkInterfaceUsed • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 N2C1 N2 D1 D2 NR Connected newCall		
Device D2 is alerted.	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2		Receiving this event depends on the type of network interface. 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).
Device D2 answers the call.	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2		Answer supervision is received from the network (this depends upon the type of signalling supported by the network). 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).

7.2 Manually dialled call to a device outside the CSTA sub-domain

In this scenario the device D1 is manually lifted to initiate a call, and the call is routed out of the CSTA sub-domain. The user dials a trunk access code and the NID is selected. Then the user completes dialling the external number.

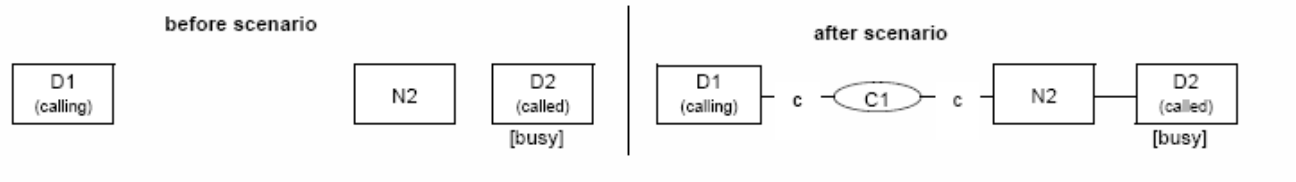
Activity	Monitored Device D1		Monitored Device N2			Comments
User at Device D1 goes Off-Hook and dials the trunk access code.	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • cause	 D1C1 D1 Initiated newCall				
The call leaves the CSTA sub-domain.	NetworkReachedEvent • outboundConnection • networkInterfaceUsed • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 N2C1 N2 D1 NK NR Initiated newCall	NetworkReachedEvent • outboundConnection • networkInterfaceUsed • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 N2C1 N2 D1 NK NR Connected newCall		
User at Device D1 completes dialling device D2 and D1 is connected to the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause • assoc.CalledDevice	 D1C1 D1 D2 Connected newCall N2	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause • assoc.CalledDevice	 D1C1 D1 D2 Connected newCall N2		
Device D2 is alerted.	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2		Receiving this event depends on the type of network interface. 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).

Activity	Monitored Device D1		Monitored Device N2			Comments
Device D2 answers the call.	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2	EstablishedEvent • establishedConnection • answeringDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2		Answer supervision is received from the network (this depends upon the type of signalling supported by the network). 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).

7.3 Make Call service - busy called device is outside the CSTA sub-domain

This scenario illustrates a Make Call service request on behalf of device D1 to a busy device D2 outside the CSTA sub-domain.

Event information after the Network Reached event depends on the type of the network interface.



Activity	Monitored Device D1		Monitored Device N2			Comments
A Make Call service to a valid device outside the CSTA sub-domain is invoked on behalf of device D1.	MakeCallRequest • callingDevice • calledDevice • autoOriginate	 D1 D2 doNotPrompt				
Acknowl-edgement.	MakeCallResult • initiatedCall	 D1C1				
Indication that service has been initiated from this device.	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • cause	 D1C1 D1 Initiated newCall				
D1 is connected to the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause	 D1C1 D1 D2 Connected newCall				
The call leaves the CSTA sub-domain.	NetworkReachedEvent • outboundConnection • networkInterfaceUsed • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 N2C1 N2 D1 D2 NR Connected newCall	NetworkReachedEvent • outboundConnection • networkInterfaceUsed • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 N2C1 N2 D1 D2 NR Connected newCall		

Activity	Monitored Device D1		Monitored Device N2			Comments
Device D2 is busy - the call cannot be completed. Device D1 receives busy tone.	FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2	FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause • assoc.CalledDevice	 N2C1 D2 D1 D2 NR Connected networkSignal N2		Receiving this event depends on the type of network interface. 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). A cause code of Busy may also be used.

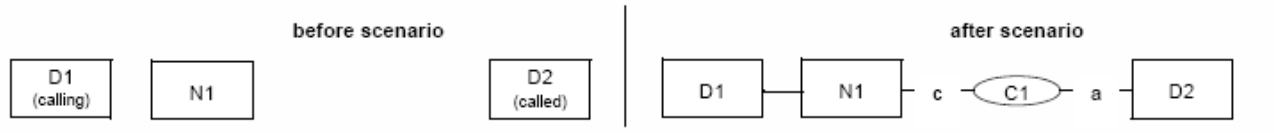
8 External Incoming Call Scenarios

This clause includes examples of successful external incoming calls.

8.1 External incoming call (no network information)

This scenario illustrates the successful incoming call from device D1. Because device D1 is located outside the CSTA sub-domain, it cannot be monitored and therefore events will be seen only for the devices N1 (NID - Network Interface Device, e.g., trunk) and D2.

In this scenario, no calling party or called party information is passed over the NID. The called device is determined via a dedicated trunk device.

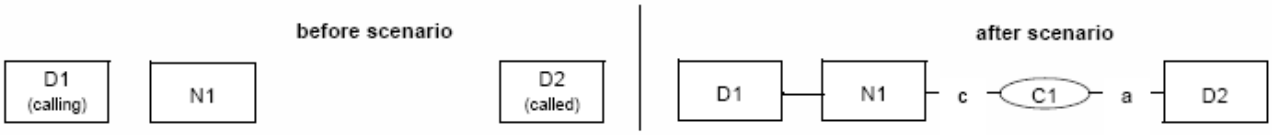


Activity	Monitored Device N1		Monitored Device D2			Comments
Indicates an external incoming call on the NID (e.g. trunk).	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • eventCause • assoc.CallingDevice	 N1C1 N1 Initiated newCall N1				
The NID has connected in the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause • assoc.CallingDevice	 N1C1 NK D2 Connected newCall N1				
Device D2 is available and is alerted.	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause • assoc.CallingDevice	 D2C1 D2 NK D2 NR Connected newCall N1	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause • assoc.CallingDevice	 D2C1 D2 NK D2 NR Alerting newCall N1		In this scenario, the network does not provide calling and called device information.

8.2 External incoming call (with network information)

This scenario illustrates the successful incoming call from device D1.

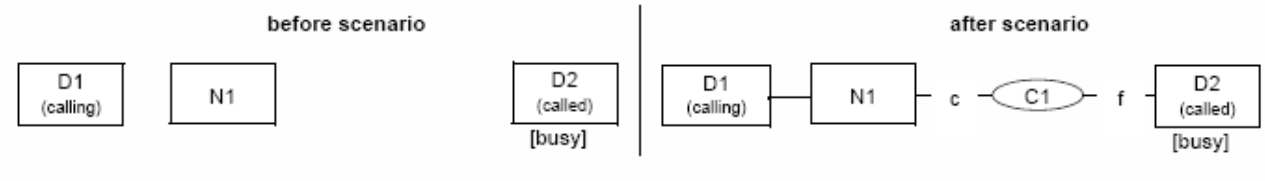
In this scenario, the network provides the calling and called party information over the NID.



Activity	Monitored Device N1		Monitored Device D2			Comments
Indicates an external incoming call on the NID (e.g. trunk).	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • eventCause • assoc.CallingDevice	 N1C1 N1 Initiated newCall N1				
The network has provided the calling and called device information over the NID.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • networkCallingDevice • networkCalledDevice • localConnectionState • cause • assoc.CallingDevice	 N1C1 D1 D2 D1 D2 Connected newCall N1				The networkCallingDevice and the networkCalledDevice parameters will not change as long as N1 is involved with the call. The callingDevice and the calledDevice parameters may change as the result of features.
Device D2 is available and begins to ring.	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • networkCallingDevice • networkCalledDevice • lastRedirectionDevice • localConnectionState • eventCause • assoc.CallingDevice	 D2C1 D2 D1 D2 D1 D2 NR Connected newCall N1	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • networkCallingDevice • networkCalledDevice • lastRedirectionDevice • localConnectionState • eventCause • assoc.CallingDevice	 D2C1 D2 D1 D2 D1 D2 NR Alerting newCall N1		

8.3 External incoming call to a busy device (with network information)

This scenario illustrates an external incoming call service to a busy device. The calling device D1 is located outside the CSTA sub-domain.



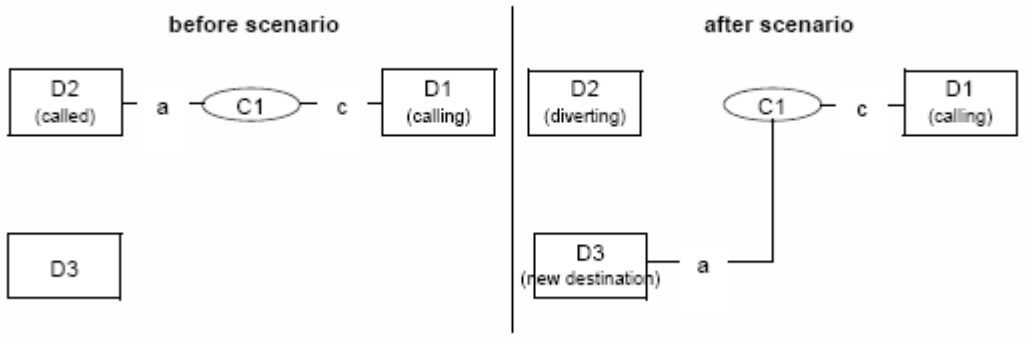
Activity	Monitored Device N1		Monitored Device D2			Comments
Indicates an external incoming call on the NID (e.g. trunk).	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • eventCause • assoc.CallingDevice	 N1C1 N1 Initiated newCall N1				
The NID is connected in the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • networkCallingDevice • networkCalledDevice • cause • assoc.CallingDevice	 N1C1 D1 D2 Connected D1 D2 newCall N1				The network has provided the calling and called devices.
Device D2 is busy - the call cannot be completed.	FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause • networkCallingDevice • networkCalledDevice • assoc.CallingDevice	 D2C1 D2 D1 D2 NR Connected busy D1 D2 N1	FailedEvent • failedConnection • failingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause • networkCallingDevice • networkCalledDevice • assoc.CallingDevice	 D2C1 D2 D1 D2 NR Failed busy D1 D2 N1		Called device D2 is busy.

9 Forwarding Call Scenarios

This clause includes examples of successful forwarding calls, forwarding calls on busy, no answer and immediate.

9.1 Call forward - no answer

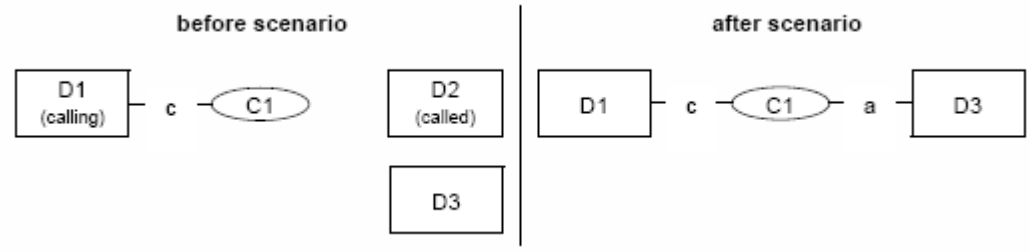
This scenario illustrates the flow for a basic call forward no answer. A call comes to a device which is set to forward calls to a predefined device after a specified number of rings.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Device D2 is alerted for a specified number of rings and then forwards the call to device D3.	DivertedEvent • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState connected • eventCause callForward-NoAnswer	DivertedEvent • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause callForward-NoAnswer		Device D3 is the device predefined by device D2 to forward its call. This illustrates the CSTA modeling option (as specified via the capability exchange services) where the Diverted event is being sent to all devices in the call, not just for the diverting device (D2) monitor.
The Call is forwarded to device D3.	DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Connected • eventCause callForward-NoAnswer		DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Alerting • eventCause callForward-NoAnswer	

9.2 Call forward - immediate

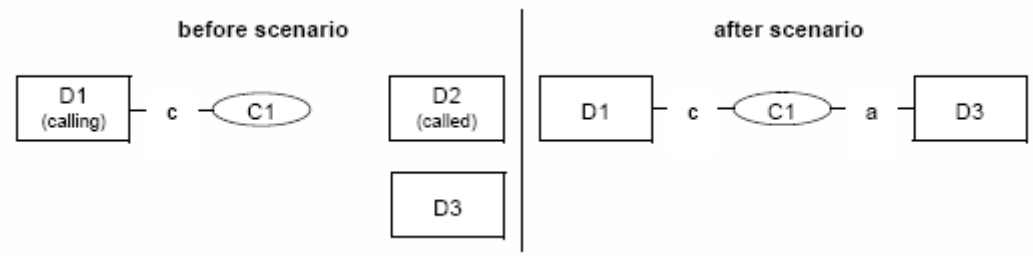
This scenario illustrates the flow for a basic call forward immediate. A call comes to a device which is set to forward calls immediately to a predefined device.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
A call is initiated by device D1 to device D2. Device D1 is already connected to the call. D2 is set to forwarded calls immediately to device D3.	DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Connected • eventCause forward-Immediate		DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Alerting • eventCause forward-Immediate	In this example, the monitor for device D2 never receives an event with respect to this call.

9.3 Call forward - immediate (with Diverted events)

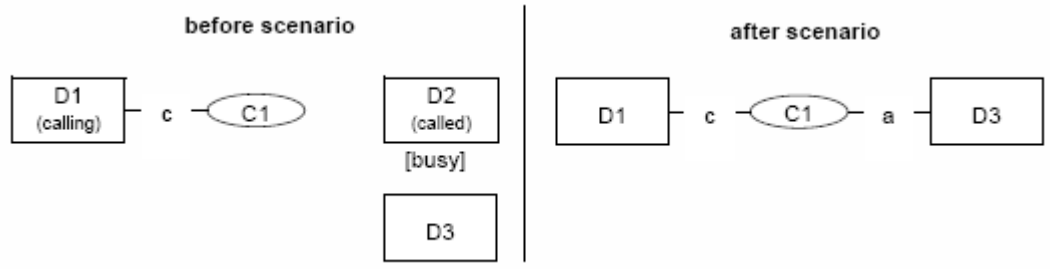
This scenario illustrates the flow for a basic call forward immediate. A call comes to a device which is set to forward calls immediately to a predefined device.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
A call is initiated by device D1 to device D2. Device D1 is already connected to the call. D2 is set to forwarded calls immediately to device D3.	DivertedEvent (See 2nd bullet in the comments column) • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause forward-Immediate	DivertedEvent (See 1st bullet in the comments column) • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause forward-Immediate		This illustrates two different CSTA modeling options (as specified via the capability exchange services): • the forwarding model where forwarding is processed after a call arrives at the device. In this case the D2 monitor flows a Diverted event representing a Null/ Null connection state transition. • the Diverted event option where the Diverted event is being sent to all devices in the call, not just for the diverting device monitor. In this case the Diverted event flows on the D1 monitor.
The call is forwarded to device D3.	DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Connected • eventCause forward-Immediate		DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Alerting • eventCause forward-Immediate	In this example, the monitor for device D2 never receives an event with respect to this call.

9.4 Call forward - busy

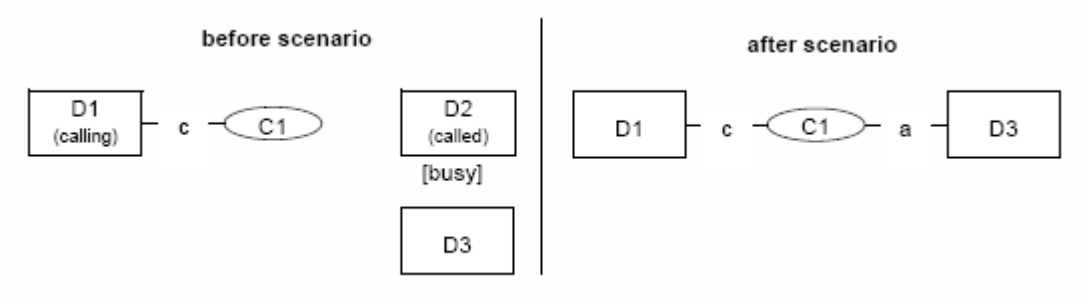
This scenario illustrates the flow for a basic call forward busy. A call comes to a device which is set to forward calls immediately to a predefined device if the called device is busy.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Since Device D2 is busy on another call, when D1 calls D2 the call is forwarded to device D3.	DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Connected • eventCause forwardBusy		DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Alerting • eventCause forwardBusy	In this example, the monitor for device D2 never receives an event with respect to this call.

9.5 Call forward - busy (with Diverted events)

This scenario illustrates the flow for a basic call forward busy. A call comes to a device which is set to forward calls immediately to a predefined device if the called device is busy.



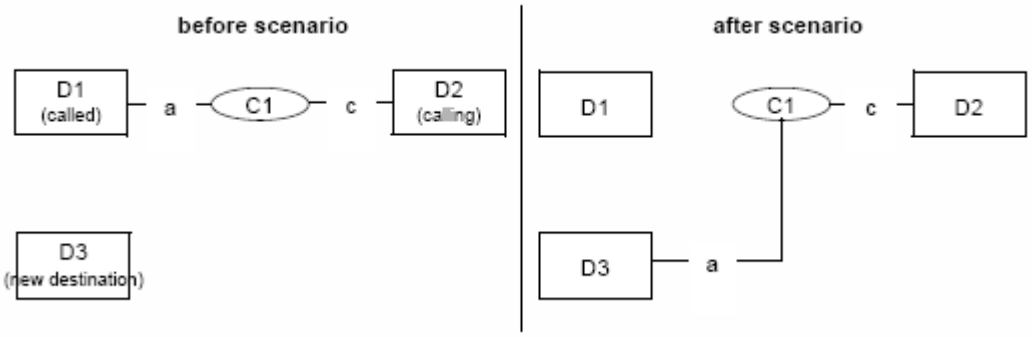
Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
A call is initiated by device D1 to device D2. Device D2 is busy with another call and has busy forwarding set to forward calls to D3.	DivertedEvent (See 2nd bullet in the comments column) • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause forward-busy	DivertedEvent (See 1st bullet in the comments column) • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause forward-busy		This illustrates two different CSTA modeling options (as specified via the capability exchange services): • the forwarding model where forwarding is processed after a call arrives at the device. In this case the D2 monitor flows a Diverted event representing a Null/ Null connection state transition. • the Diverted event option where the Diverted event is being sent to all devices in the call, not just for the diverting device monitor. In this case the Diverted event flows on the D1 monitor.
The call is forwarded to device D3.	DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Connected • eventCause forwardBusy		DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice D2 • localConnectionState Alerting • eventCause forwardBusy	

10 Call Movement Scenarios

This clause includes examples of moving calls from one device to another, initiated manually and by CSTA services.

10.1 Deflect Call service

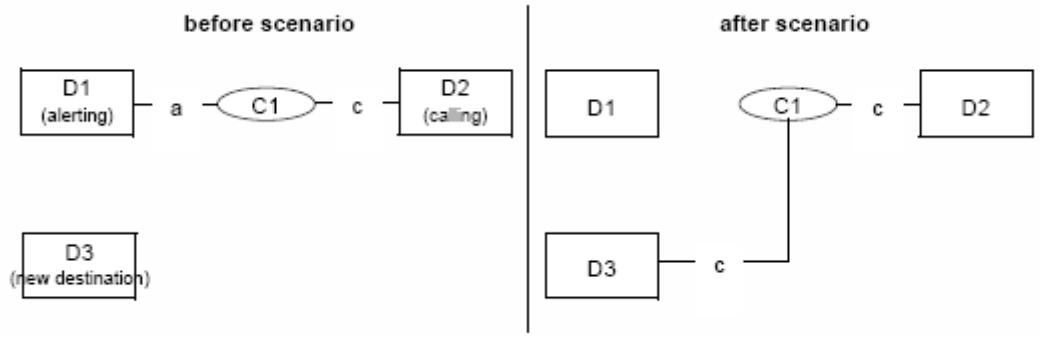
This scenario illustrates how an alerting call is diverted to another destination.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Deflect Call service is invoked on behalf of device D1.	DeflectCallRequest • callToBeDiverted D1C1 • newDestination D3			
Acknowl-edgement.	DeflectCallResult			
The event indicates that the call has been diverted from D1.	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause redirected	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause redirected		This illustrates the CSTA modeling option (as specified via the capability exchange services) where the Diverted event is being sent to all devices in the call, not just for the diverting device (D1) monitor.
The call is alerting D3.		DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause redirected	DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Alerting • eventCause redirected	

10.2 Directed Pickup Call service

This service illustrates how an alerting connection is moved from one device and connected to another device via the Directed Pickup service.

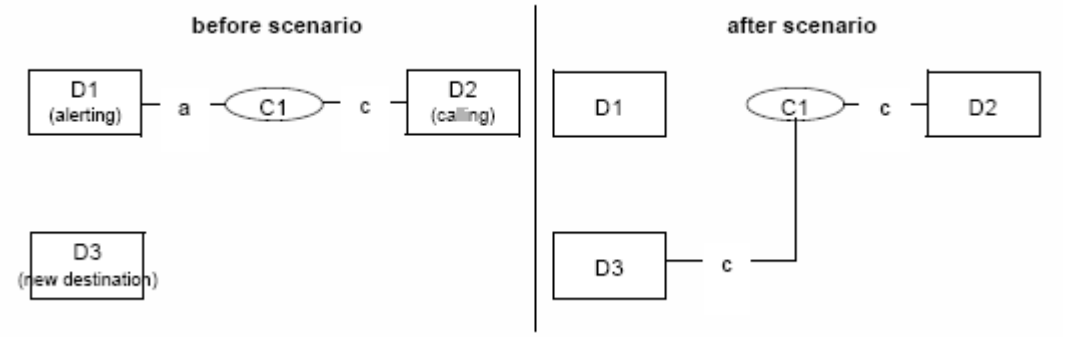


Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
A Directed Pickup Call service is invoked.			DirectedPickupCallRequest • callToBePickedUp D1C1 • newDestination D3	
Acknowl-edgement.			DirectedPickupCallResult D3C1 • pickedCall	
The call is diverted from D1.	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause callPickup	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause callPickup		This illustrates the CSTA modeling option (as specified via the capability exchange services) where the Diverted event is being sent to all devices in the call, not just for the diverting device (D1) monitor.
The call is connected to D3.		EstablishedEvent • establishedConnection D3C1 • answeringDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause callPickup	EstablishedEvent • establishedConnection D3C1 • answeringDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause callPickup	

10.3 Group Pickup Call service

This scenario illustrates a pickup of a call that is alerting at a device as a member of a specified or default pickup group.

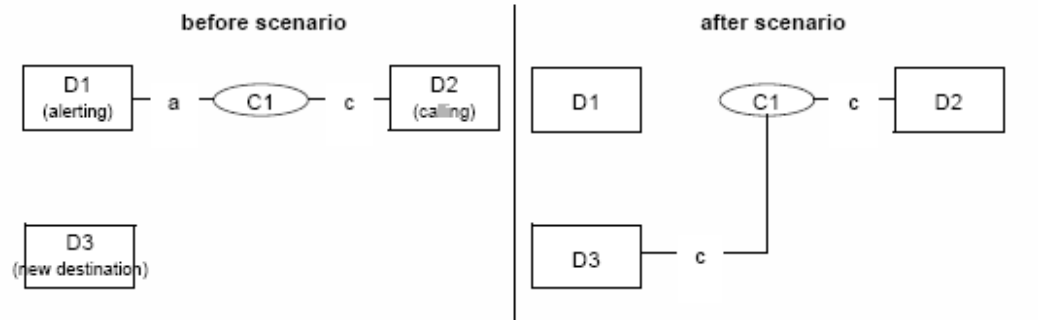
The call is moved and connected at the new specified destination.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
A Group Pickup Call service is invoked.			GroupPickupCallRequest • newDestination D3	Device D3 is in the same pickup group as device D1.
Acknowl-edgement.			GroupPickupCallResult	
The call has been diverted from device D1.	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause callPickup	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause callPickup		This illustrates the CSTA modeling option (as specified via the capability exchange services) where the Diverted event is being sent to all devices in the call, not just for the diverting device (D1) monitor.
The call is connected to device D3.		EstablishedEvent • establishedConnection D3C1 • answeringDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause callPickup	EstablishedEvent • establishedConnection D3C1 • answeringDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause callPickup	

10.4 Manual group pick up

Device D2 has called device D1. Device D3 and device D1 are in the same pickup group. Device D3 is going Off-Hook and dialing a specific code number issuing the Group Pickup Call to answer the call.

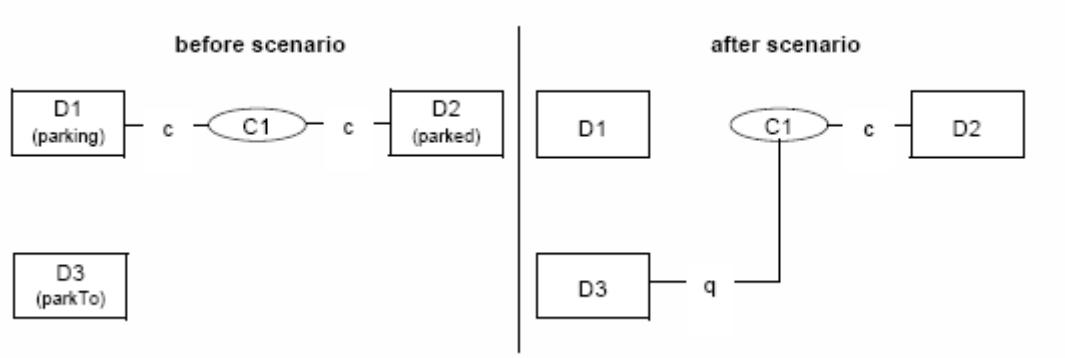


Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Device D3 goes Off-Hook in order to invoke a feature.			ServiceInitiatedEvent • initiatedConnection D3C2 • initiatingDevice D3 • localConnectionState Initiated • eventCause newCall	Device D3 is in the same pickup group as device D1. A manual pickup could also be invoked via a button on the phone.
The connection was cleared after the feature access code was entered.			ConnectionClearedEvent • droppedConnection D3C2 • releasingDevice D3 • localConnectionState Null • eventCause normalClearing	The feature could also have been invoked via a button on the phone.
The call is diverted from device D1.	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause callPickup	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause callPickup		This illustrates the CSTA modeling option (as specified via the capability exchange services) where the Diverted event is being sent to all devices in the call, not just for the diverting device (D1) monitor.

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
The call is connected at device D3.		EstablishedEvent • establishedConnection D3C1 • answeringDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause callPickup	EstablishedEvent • establishedConnection D3C1 • answeringDevice D3 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause callPickup	

10.5 Park Call service

This scenario illustrates how a connected call is parked at another device.



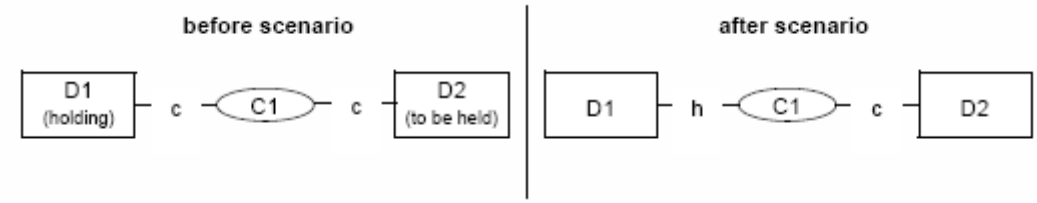
Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Park Call service is invoked on behalf of device D1.	ParkCallRequest • parking D1C1 • parkTo D3			
Acknowl-edgement.	ParkCallResult			
The event indicates that the call has been diverted from D1.	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause park	DivertedEvent • divertingConnection D1C1 • divertingDevice D1 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause park		This illustrates the CSTA modeling option (as specified via the capability exchange services) where the Diverted event is being sent to all devices in the call, not just for the diverting device (D1) monitor.
The call is parked at D3.		QueuedEvent • queuedConnection D3C1 • queue D3 • callingDevice D2 • calledDevice D3 • lastRedirectionDevice D1 • localConnectionState Connected • eventCause park	QueuedEvent • queuedConnection D3C1 • queue D3 • callingDevice D2 • calledDevice D3 • lastRedirectionDevice D1 • localConnectionState Queued • eventCause park	

11 Holding/Retrieving Call Scenarios

This clause includes examples of successful Hold and Retrieve Call scenarios.

11.1 Hold Call service

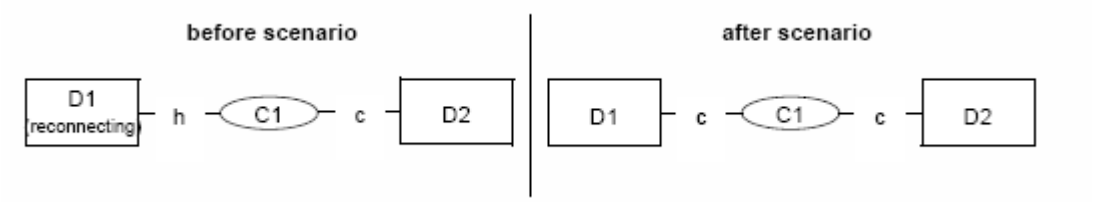
This scenario illustrates the successful use of a Hold Call service. The service places an existing connection on hold.



Activity	Monitored Device D1		Monitored Device D2			Comments
A Hold Call service is issued on behalf of D1.	HoldCallRequest • callToBeHeld	D1C1				
Acknowl-edgement.	HoldCallResult					
Connection placed on hold.	HeldEvent • heldConnection • holdingDevice • localConnectionState • eventCause	D1C1 D1 Held normal	HeldEvent • heldConnection • holdingDevice • localConnectionState • eventCause	D1C1 D1 Connected normal		

11.2 Retrieve Call service

This service reconnects to a call that has previously been placed on hold.



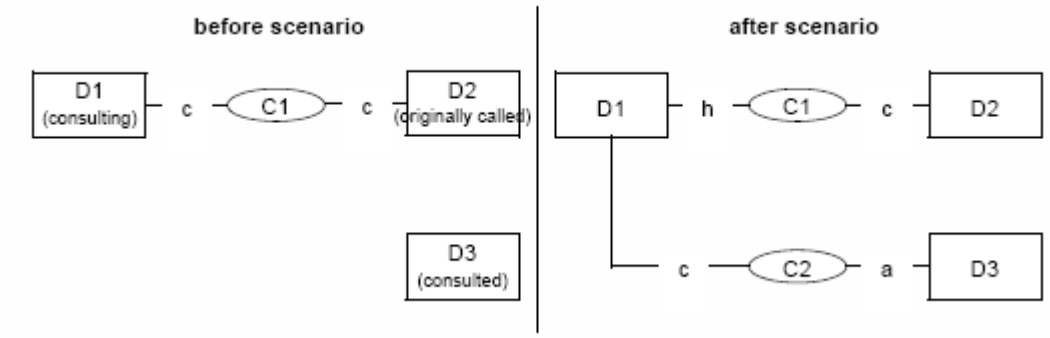
Activity	Monitored Device D1		Monitored Device D2			Comments
Device D1 issues the Retrieve Call service to retrieve the held call.	RetrieveCallRequest • heldCall	D1C1				
Acknowl-edgement.	RetrievedCallResult					
Device D2 is connected back into the previ-ously held call.	RetrievedEvent • retrievedConnection • retrievingDevice • localConnectionState • eventCause	D1C1 D1 Connected normal	RetrievedEvent • retrievedConnection • retrievingDevice • localConnectionState • eventCause	D1C1 D1 Connected normal		

12 Consultation Call Scenarios

This clause illustrates examples of successful Consultation Calls, Reconnect Calls and Alternate Calls initiated manually and by CSTA services.

12.1 Consultation Call service

This service places an existing active call at a device on hold and initiates a new call from the same device.

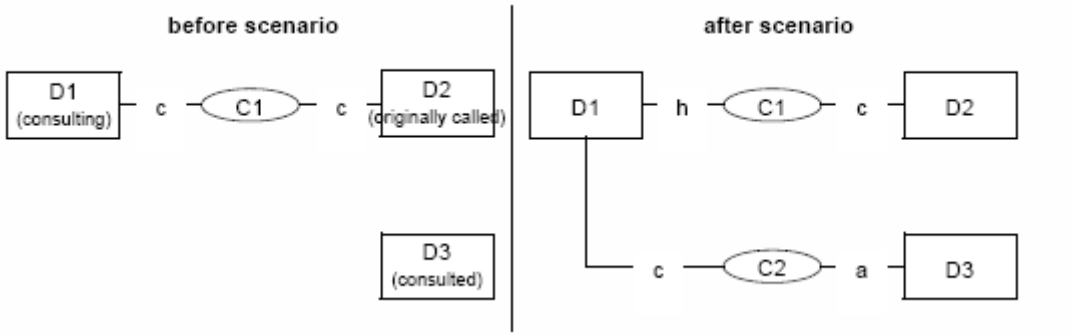


Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Consultation Call service to device D3 is invoked.	ConsultationCallRequest • existingCall D1C1 • consultedDevice D3			
Acknowl-edgement.	ConsultationCallResult • initiatedCall D1C2			
Connection placed on hold.	HeldEvent • heldConnection D1C1 • holdingDevice D1 • localConnectionState Held • eventCause consultation	HeldEvent • heldConnection D1C1 • holdingDevice D1 • localConnectionState Connected • eventCause consultation		
Consultation call is initiated.	ServiceInitiatedEvent • initiatedConnection D1C2 • initiatingDevice D1 • localConnectionState Initiated • eventCause consultation			The generation of this event is switch specific.

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Device D1 is connected in the call.	OriginatedEvent • originatedConnection D1C2 • callingDevice D1 • calledDevice D3 • localConnectionState Connected • eventCause consultation			
Device D3 begins to ring.	DeliveredEvent • connection D3C2 • alertingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause newCall		DeliveredEvent • connection D3C2 • alertingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice NR • localConnectionState Alerting • eventCause newCall	

12.2 Manual consultation call

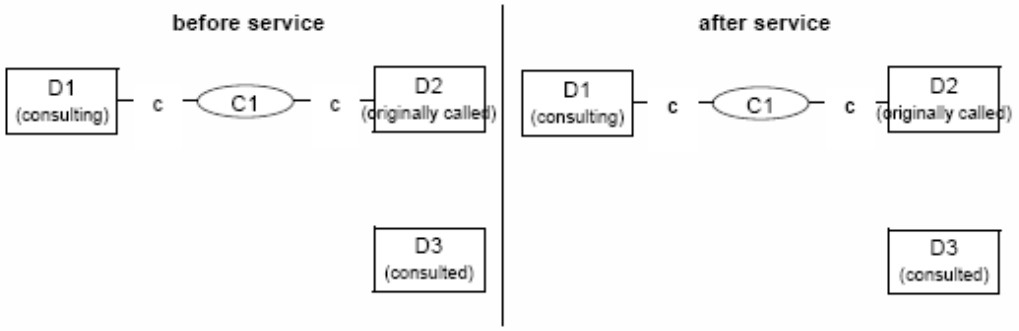
Device D1 is connected with device D2. Device D1 manually places device D2 on hold and creates a new call to device D3.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Device D1 manually places device D2 on hold.	HeldEvent • heldConnection D1C1 • holdingDevice D1 • localConnectionState Held • eventCause normal	HeldEvent • heldConnection D1C1 • holdingDevice D1 • localConnectionState Connected • eventCause normal		
A new connection is created at device D1.	ServiceInitiatedEvent • initiatedConnection D1C2 • initiatingDevice D1 • localConnectionState Initiated • eventCause newCall			The generation of this event is switch specific.
Device D1 is connected in the call.	OriginatedEvent • originatedConnection D1C2 • callingDevice D1 • calledDevice D3 • localConnectionState Connected • eventCause newCall			
Device D2 is available and begins to ring.	DeliveredEvent • alertingConnection D3C2 • alertingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause newCall		DeliveredEvent • alertingConnection D3C2 • alertingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice NR • localConnectionState Alerting • eventCause newCall	

12.3 Consultation Call service (negative acknowledgement)

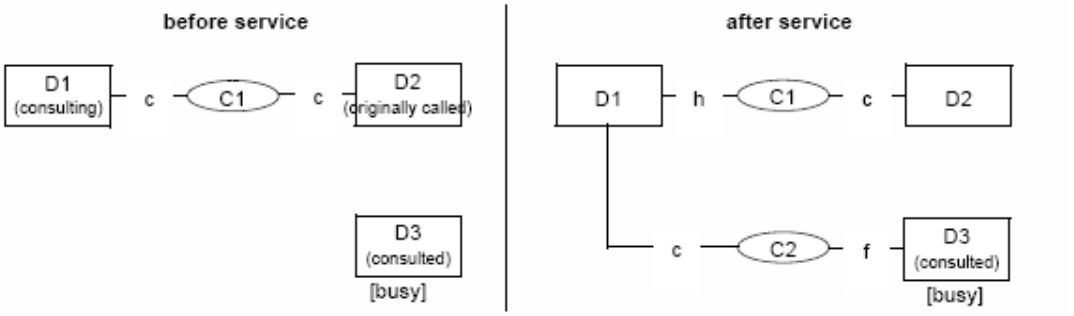
Device D1, which already has an active call, invokes the Consultation Call service which provides the compound action of the Hold Call service and the Make Call service. The Hold Call request may be rejected due to service feature restrictions.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Consultation Call service to device D3 is invoked on behalf of device D1.	ConsultationCallRequest • existingCall D1C1 • consultedDevice D3			
Negative Acknowl-edgement.	ConsultingCallError • operationalError invalidFeature			The possible error categories and error values are described in ECMA-269.

12.4 Consultation Call service - consulted party is busy

This scenario illustrates a Consultation Call service request invoked on behalf of device D1 to the device D3 that is busy. The original call is successfully placed on hold.

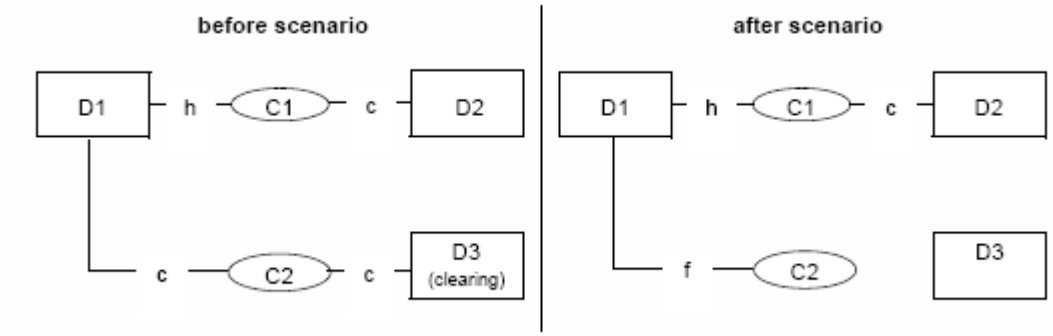


Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Consultation Call service to device D3 is invoked.	ConsultationCallRequest • existingCall D1C1 • consultedDevice D3			
Acknowl-edgement.	ConsultationCallResult • initiatedCall D1C2			
The D1C1 connection is placed on hold.	HeldEvent • heldConnection D1C1 • holdingDevice D1 • localConnectionState Held • eventCause consultation	HeldEvent • heldConnection D1C1 • holdingDevice D1 • localConnectionState Connected • eventCause consultation		
A new call is initiated.	ServiceInitiatedEvent • initiatedConnection D1C2 • initiatingDevice D1 • localConnectionState Initiated • eventCause consultation			The generation of this event is switch specific.
Device D1 is connected in the call.	OriginatedEvent • originatedConnection D1C2 • callingDevice D1 • calledDevice D3 • localConnectionState Connected • eventCause consultation			

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
The new call will not be delivered, because device D3 is busy in another call.	FailedEvent • failedConnection D3C2 • failingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause busy		FailedEvent • failedConnection D3C2 • failingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice NR • localConnectionState Failed • eventCause busy	This scenario illustrates the CSTA modeling option where the Failed event is generated for the busy device, with a complete connectionID.

12.5 Consulted party disconnects using the Clear Connection service

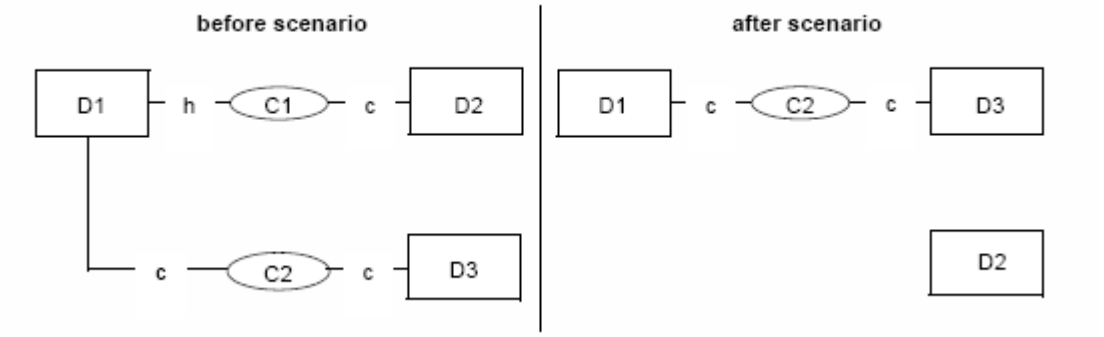
This scenario illustrates the successful use of the Clear Connection service. The Clear Connection service request is issued for the consulted party after the consulted device D3 is connected into the new call.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Device D3 wants to clear from the call and a Clear Connection service is invoked.			ClearConnectionRequest • connectionToBeCleared D3C2	
Acknowl-edgement.			ClearConnectionResult	
Connection D3C2 is cleared.	ConnectionClearedEvent • droppedConnection D3C2 • releasingDevice D3 • localConnectionState Connected • eventCause normalClearing		ConnectionClearedEvent • droppedConnection D3C2 • releasingDevice D3 • localConnectionState Null • eventCause normalClearing	
Connection D1C2 fails as a result of the clearing of D3.	FailedEvent • failedConnection D1C2 • failingDevice D1 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice NR • localConnectionState Failed • eventCause blocked			In this example, the user at D1 hears, as a result of the far end disconnect, error tone, until D1 goes on-hook.

12.6 Held party disconnects using the Clear Connection service

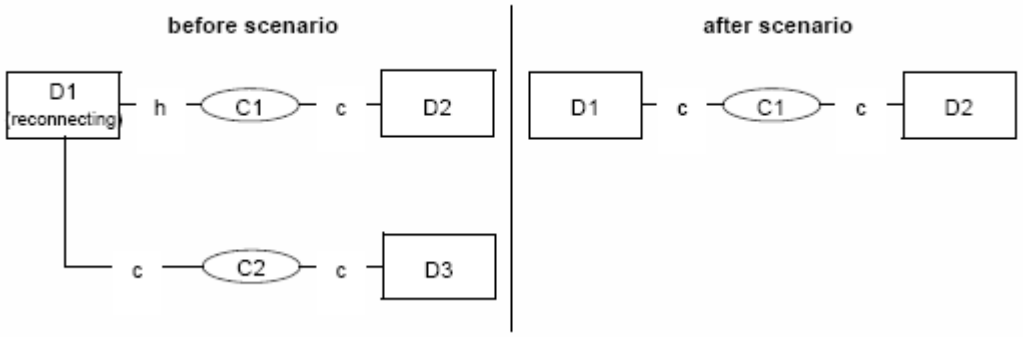
This scenario illustrates the successful use of the Clear Connection service. The Clear Connection service request is issued for the held party after the consulted device D3 is connected into the new call.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Device D2 wants to clear from the call and a Clear Connection service is invoked.		ClearConnectionRequest • connectionToBeCleared D2C1		
Acknowl-edgement.		ClearConnectionResult		
Connection D2C1 is cleared.	ConnectionClearedEvent • droppedConnection D2C1 • releasingDevice D2 • localConnectionState Held • eventCause normalClearing	ConnectionClearedEvent • droppedConnection D2C1 • releasingDevice D2 • localConnectionState Null • eventCause normalClearing		
As a result of the D2C1 clearing, D1C1 is also cleared.	ConnectionClearedEvent • droppedConnection D1C1 • releasingDevice D1 • localConnectionState Null • eventCause normalClearing			

12.7 Reconnect Call service

This service clears an existing connection and then retrieves a previously held connection at the same device.

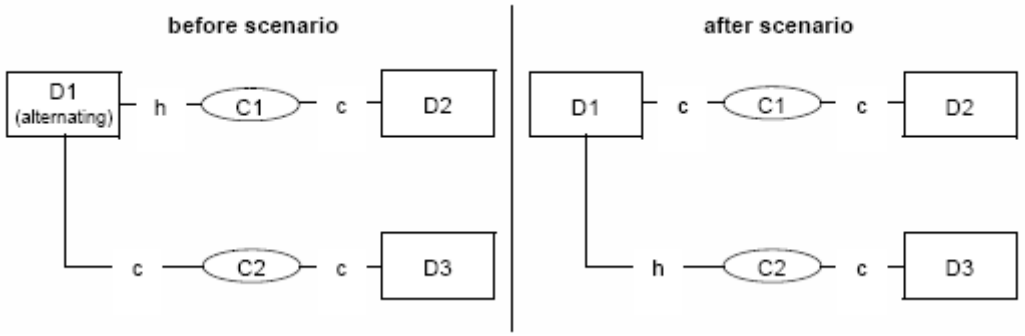


Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Device D1 has finished consultation and issues the Reconnect Call service to drop from the active call and retrieve the held call.	ReconnectCallRequest • heldCall D1C1 • activeCall D1C2			
Acknowl-edgement.	ReconnectCallResult			
Device D1 is cleared from the active call.	ConnectionClearedEvent • droppedConnection D1C2 • releasingDevice D1 • localConnectionState Null • eventCause normalClearing		ConnectionClearedEvent • droppedConnection D1C2 • releasingDevice D1 • localConnectionState Connected • eventCause normalClearing	
Device D2 is connected back into the previ-ously held call.	RetrievedEvent • retrievedConnection D1C1 • retrievingDevice D1 • localConnectionState Connected • eventCause normal	RetrievedEvent • retrievedConnection D1C1 • retrievingDevice D1 • localConnectionState Connected • eventCause normal		

Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
As the result of D1C2 clearing from the call, D3C2 is also cleared.			ConnectionClearedEvent • droppedConnection D3C2 • releasingDevice D3 • localConnectionState Null • cause normalClearing	

12.8 Alternate Call service

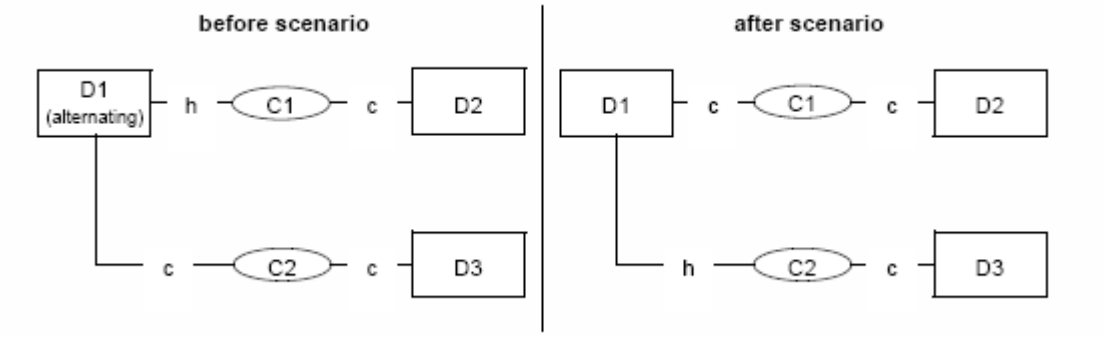
This service places an existing active call on hold and then retrieves a previously held call at the same device. The effect of this service is to swap the device's active and held calls.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
The Alternate Call service is invoked.	AlternateCallRequest • heldCall D1C1 • activeCall D1C2			
Acknowl-edgement.	AlternateCallResult			
Connection D1C2 is placed on hold in the active call.	HeldEvent • heldConnection D1C2 • holdingDevice D1 • localConnectionState Held • eventCause alternate		HeldEvent • heldConnection D1C2 • holdingDevice D1 • localConnectionState Connected • eventCause alternate	
Device D1 is connected into the previously held call.	RetrievedEvent • retrievedConnection D1C1 • retrievingDevice D1 • localConnectionState Connected • eventCause alternate	RetrievedEvent • retrievedConnection D1C1 • retrievingDevice D1 • localConnectionState Connected • eventCause alternate		

12.9 Manual alternate call

This scenario illustrates the successful use of the manually alternate feature to place an existing active call on hold and then to retrieve a previously held call at the same device. The effect of this scenario is to swap the devices active and held calls.



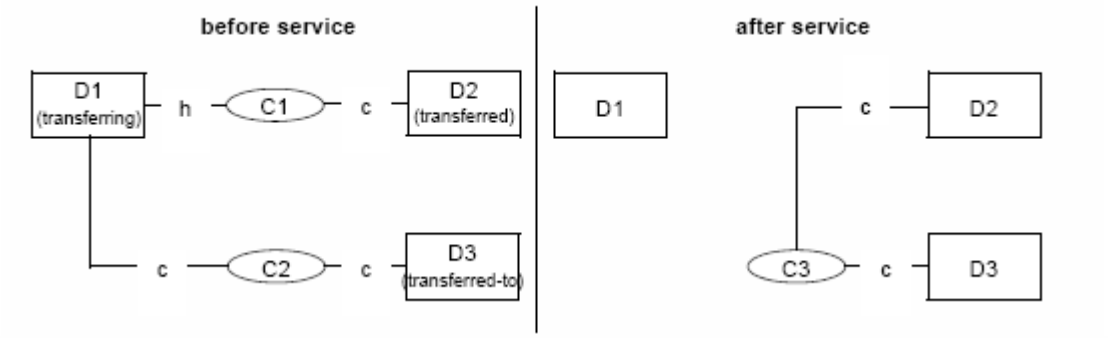
Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Connection D1C2 is placed on hold in the active call.	HeldEvent • heldConnection D1C2 • holdingDevice D1 • localConnectionState Held • eventCause normal		HeldEvent • heldConnection D1C2 • holdingDevice D1 • localConnectionState Connected • eventCause normal	Device D1 wishes to place device D3 on hold and to connect to device D2. It issues the alternate feature manually to do this.
Device D1 is connected into the previously held call.	RetrievedEvent • retrievedConnection D1C1 • retrievingDevice D1 • localConnectionState Connected • eventCause normal	RetrievedEvent • retrievedConnection D1C1 • retrievingDevice D1 • localConnectionState Connected • eventCause normal		

13 Transfer Call Scenarios

This clause includes examples of successful call transfers.

13.1 Transfer Call service - screened transfer (with fixed view in Transferred event)

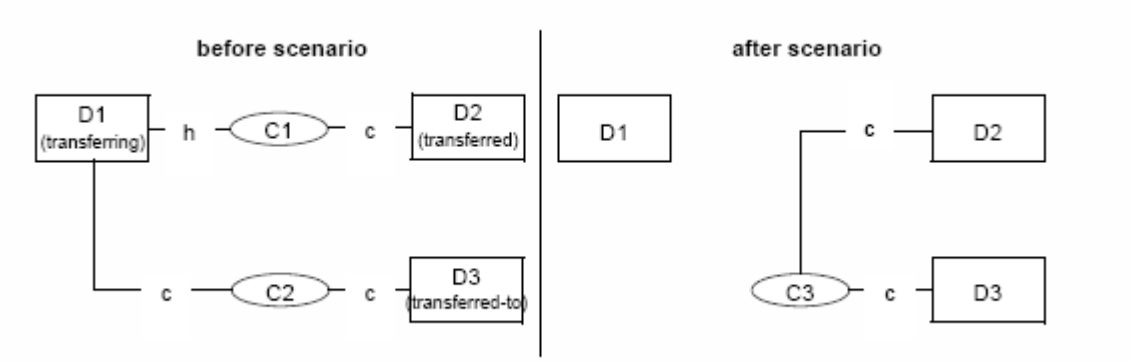
This service transfers a held party to a consulted party.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Transfer Call service is invoked on behalf of device D1.	TransferCallRequest • heldCall D1C1 • activeCall D1C2			
Acknowl-edgement.	TransferCallResult • transferredCall D3C3			
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.	TransferredEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • transferringDevice D1 • transferredToDevice D3 • transferredConnection D2C3 • transferredConnection D3C3 • localConnectionState Null • eventCause transfer	TransferredEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • transferringDevice D1 • transferredToDevice D3 • transferredConnection D2C3 • transferredConnection D3C3 • localConnectionState Connected • eventCause transfer	TransferredEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • transferringDevice D1 • transferredToDevice D3 • transferredConnection D2C3 • transferredConnection D3C3 • localConnectionState Connected • eventCause transfer	The CSTA Transferred event Fixed View modeling option is illustrated in this scenario. This means that the primary old call parameters in the Transferred event represent a fixed view in contrast to a device oriented view.

13.2 Transfer Call service - screened transfer (with local view in Transferred event)

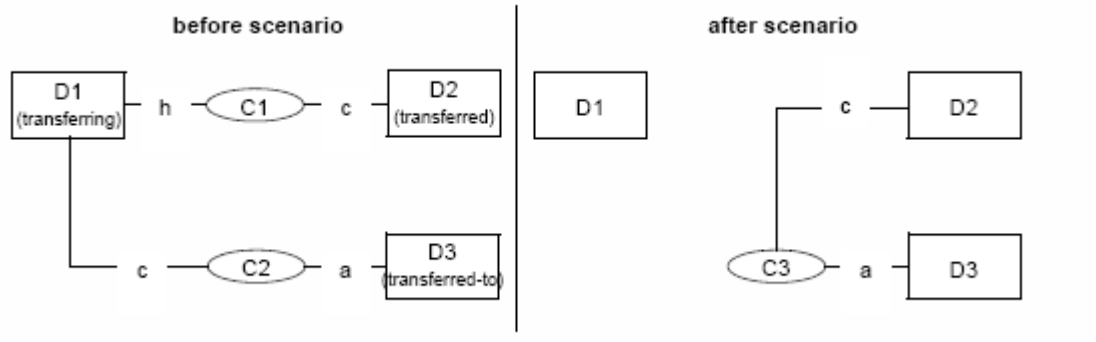
This service transfers a held party to a consulted party.



Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
Transfer Call service is invoked on behalf of device D1.	TransferCallRequest • heldCall • activeCall	 D1C1 D1C2					
Acknowl-edgement.	TransferCallResult • transferredCall	 D3C3					
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.	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D1C1 D1C2 D1 D3 D2C3 D3C3 Null Transfer	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D2C1 NR D1 D3 D2C3 D3C3 Connected Transfer	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D3C2 NR D1 D3 D2C3 D3C3 Connected Transfer	The CSTA Transferred event Local View modeling option is illustrated in this scenario. This means that the primary old call parameters in the Transferred event represent a device oriented view.

13.3 Transfer Call Call service - blind transfer (with local view in Transferred event)

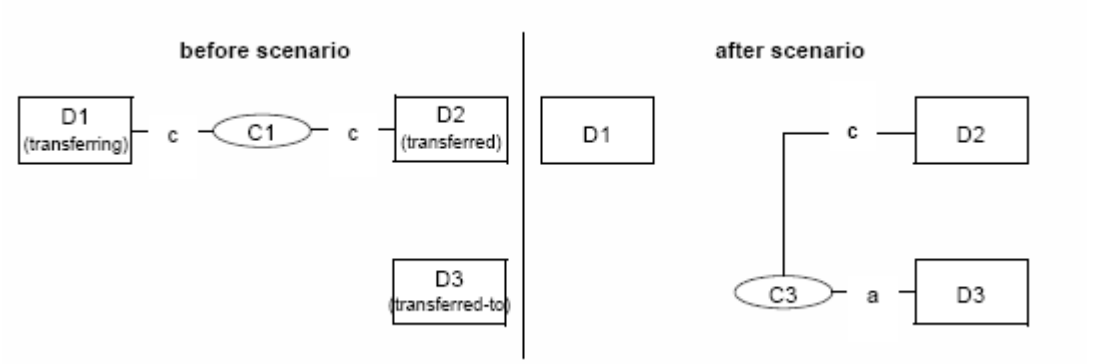
This service transfers a held party to a consulted party. The transfer service request is issued before the consulted device connects into the new call.



Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
Transfer Call service is invoked on behalf of device D1.	TransferCallRequest • heldCall • activeCall	 D1C1 D1C2					
Acknowl-edgement.	TransferCallResult • transferredCall	 D3C3					
Calls between D1,D2 and D1,D3 are released. The connections between D2, D1 and D3,D1 are replaced with a single alerting connection between D2 and D3.	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D1C1 D1C2 D1 D3 D2C3 D3C3 Null transfer	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D1C1 D1C2 D1 D3 D2C3 D3C3 Connected transfer	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D1C1 D1C2 D1 D3 D2C3 D3C3 Alerting transfer	The CSTA Transferred event Fixed View modeling option is illustrated in this scenario. This means that the primary old call parameters in the Transferred event represent a fixed view in contrast to a device oriented view.

13.4 Single Step Transfer Call service

This service transfers a device in one step (i.e., without putting the device on hold).



Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
Single Step Transfer Call service is invoked on behalf of device D1.	SingleStepTransferRequest • activeCall • deviceToTransferTo	 D1C1 D3					
Acknowl-edgement.	SingleStepTransferResult • transferredCall	 D3C3					
The call between D1 and D2 is replaced with an alerting call between D2 and D3.	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D1C1 NR D1 D3 D2C3 D3C3 Null singleStep-Transfer	TransferredEvent • primaryOldCall • secondaryOldCall • transferringDevice • transferredToDevice • transferredConnection • transferredConnection • localConnectionState • eventCause	 D2C1 NR D1 D3 D2C3 D3C3 Connected singleStep-Transfer			The CSTA Transferred event Local View modeling option is illustrated in this scenario. This means that the primary old call parameters in the Transferred event represent a device oriented view. Since there is no connection at D3 at the time of the transfer, there is no Transferred event generated on D3's monitor.

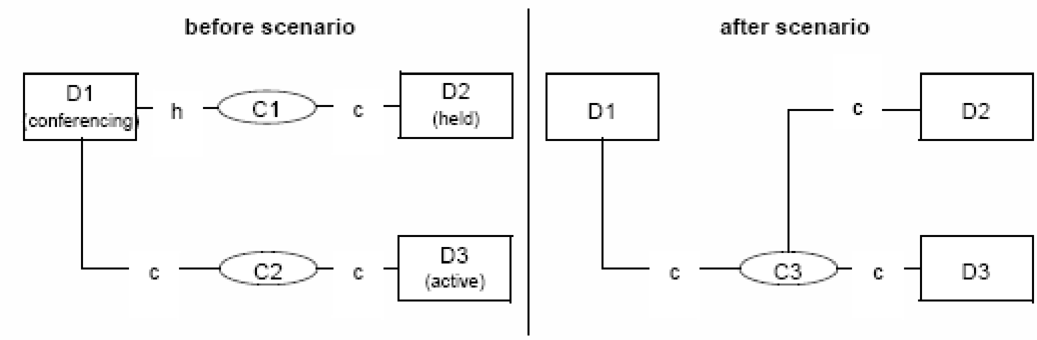
Activity	Monitored Device D1		Monitored Device D2		Monitored Device D3		Comments
The call alerts device D3.			DeliveredEvent • connection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D3C3 D3 D2 D3 NR Connected singleStep-Transfer	DeliveredEvent • connection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • cause	 D3C3 D3 D2 D3 NR Alerting singleStep-Transfer	This event reflects the connection state change at D3C3.

14 Conference Call Scenarios

This clause includes examples of successful conference calls.

14.1 Conference Call service

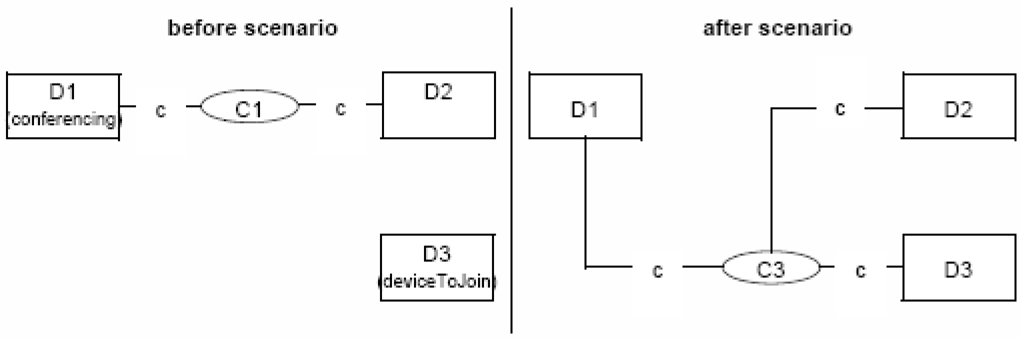
This service provides a conference of an existing held call and another active call at a conferencing device. The two calls are merged into a single call at the conferencing device.



Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Conference Call Service is requested on behalf of device D1.	ConferenceCallRequest • heldCall D1C1 • activeCall D1C2			
Acknowl-edgement.	ConferenceCallResult • conferenceCall D1C3			
Conference established.	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • conferencingDevice D1 • addedDevice D3 • conferenceConnection D1C3 • conferenceConnection D2C3 • conferenceConnection D3C3 • localConnectionState Connected • eventCause conference	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • conferencingDevice D1 • addedDevice D3 • conferenceConnection D1C3 • conferenceConnection D2C3 • conferenceConnection D3C3 • localConnectionState Connected • eventCause conference	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • conferencingDevice D1 • addedDevice D3 • conferenceConnection D1C3 • conferenceConnection D2C3 • conferenceConnection D3C3 • localConnectionState Connected • eventCause conference	The added device is the device representing the person who has just joined the call from the perspective of the participants. Note that the primaryOldCall and the secondaryOldCall parameters illustrate the "fixed view" modeling option.

14.2 Single Step Conference Call service

This service provides a conference of an existing call in one step (i.e., without first having to put the existing call on hold). The Single Step Conference Call service is invoked on behalf of device D1 which wishes to silently join the call. The three devices D1, D2, and D3 are then involved in a single call C1. The scenario begins at the point where device D1 begins to join the call. This scenario assumes that device D3 is set to auto-answer the call so device D1 is not shown being prompted to lift the handset.



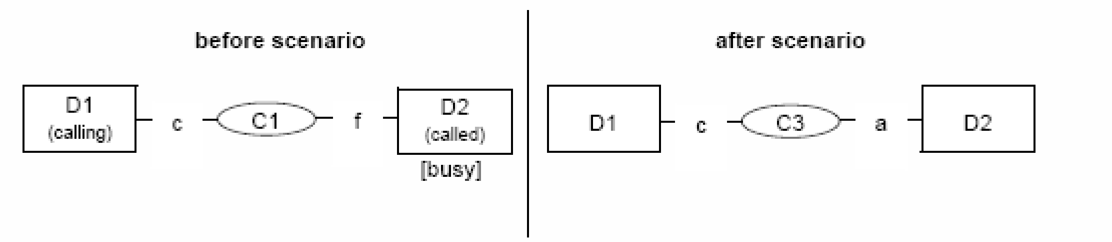
Activity	Monitored Device D1	Monitored Device D2	Monitored Device D3	Comments
Conference Call Service is requested on behalf of device D1.	SingleStepConfCallRequest • activeCall D1C1 • deviceToJoin D3			
Acknowl-edgement.	SingleStepConfCallResult • conferenceCall D1C3			
Conference is established.	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall NR • conferencingDevice D1 • addedDevice D3 • conferenceConnection D1C3 • conferenceConnection D2C3 • conferenceConnection D3C3 • localConnectionState Connected • eventCause singleStepConference	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall NR • conferencingDevice D1 • addedDevice D3 • conferenceConnection D1C3 • conferenceConnection D2C3 • conferenceConnection D3C3 • localConnectionState Connected • eventCause singleStepConference	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall NR • conferencingDevice D1 • addedDevice D3 • conferenceConnection D1C3 • conferenceConnection D2C3 • conferenceConnection D3C3 • localConnectionState Connected • eventCause singleStepConference	The added device is the device representing the person who has just joined the call from the perspective of the participants.

15 Call Completion Scenarios

This clause includes examples of call completion scenarios including call back and camp on.

15.1 Call Back Call-Related service - called device is busy

This scenario illustrates the use of the Call Back Call-Related service where the called device is busy.

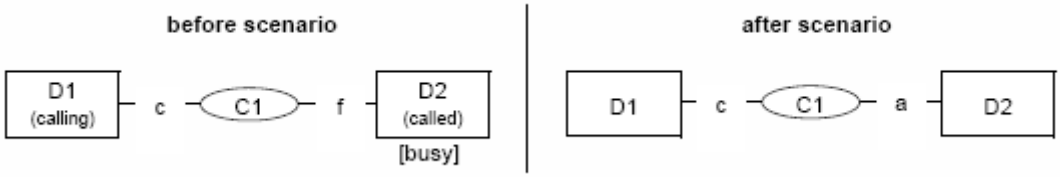


Activity	Monitored Device D1	Monitored Device D2		Comments
The Call Back Call-Related service is invoked on behalf of device D1.	CallBackCallRelatedRequest • callBack D1C1			
Acknowl-edgement.	CallBackCallRelatedResult			
The connection is cleared as a result of the call back request.	ConnectionClearedEvent • droppedConnection D1C1 • releasingDevice D1 • localConnectionState Null • eventCause callback	ConnectionClearedEvent • droppedConnection D1C1 • releasingDevice D1 • localConnectionState Failed • eventCause callback		
Failed connection D2C1 also clears.		ConnectionClearedEvent • droppedConnection D2C1 • releasingDevice D2 • localConnectionState Null • eventCause normalClearing		
Device D2 sometime later clears from its active call.		ConnectionClearedEvent • droppedConnection D2C2 • releasingDevice D2 • localConnectionState Null • eventCause normalClearing		

Activity	Monitored Device D1		Monitored Device D2			Comments
Since device D2 is now available, the callback is initiated from device D1. D1 is being prompted to go off-hook.	ServiceInitiatedEvent • initiatedConnection • initiatingDevice • localConnectionState • eventCause	 D1C3 D1 Initiated callback				The cause code of callback indicates that the device is being prompted to go off-hook.
Device D1 goes off hook and is connected in the call.	OriginatedEvent • originatedConnection • callingDevice • calledDevice • localConnectionState • cause	 D1C3 D1 D2 Connected callback				
Device D2 is alerted.	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause	 D2C3 D2 D1 D2 NR Connected callback	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause	 D2C3 D2 D1 D2 NR Alerting callback		

15.2 Camp On Call service

This service queues a call for a device (that typically is busy) until that device becomes available.



Activity	Monitored Device D1		Monitored Device D2			Comments
The Camp On Call service is invoked on behalf of device D1.	CampOnCallRequest • campOn	 D1C1				
Acknowl-edgement.	CampOnCallResult					
The call is queued at D2.	QueuedEvent • queuedConnection • queueDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause	 D2C1 D2 D1 D2 NR Connected campOn	QueuedEvent • queuedConnection • queueDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause	 D2C1 D2 D1 D2 NR Queued campOn		Queued events may be provided instead of a Delivered event (for example for a second caller).
Device D2 sometime later clears from its active call.			ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • eventCause	 D2C2 D2 Null normalClearing		
Since D2 is available, the call alerts D2.	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause	 D2C1 D2 D1 D2 NR Connected campOn	DeliveredEvent • alertingConnection • alertingDevice • callingDevice • calledDevice • lastRedirectionDevice • localConnectionState • eventCause	 D2C1 D2 D1 D2 NR Alerting campOn		

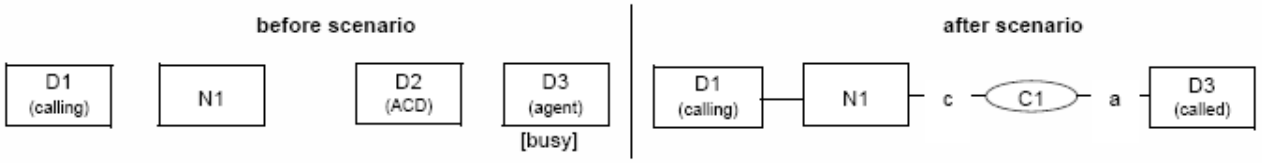
16 Distributing Call Scenarios

This clause illustrates calls that are distributed by a CSTA distribution device (ACD) to an available device.

16.1 Incoming Call to ACD with no available agents

This scenario illustrates an incoming call to an ACD distribution mechanism (ACD). Since there is no available device (ACD agent) the call is queued at the distribution device until an agent becomes available.

The calling device D1 is located outside the CSTA sub-domain and therefore no events are shown for this device. Device N1 in this scenario represents the Network Interface Device (trunk) associated with the calling device.



Activity	Monitored Device N1 (trunk)	Monitored Device D2 (ACD)	Monitored Device D3 (agent).	Comments
Indicates an external incoming call on the NID (e.g. trunk).	ServiceInitiatedEvent • initiatedConnection N1C1 • initiatingDevice N1 • localConnectionState Initiated • eventCause newCall • assoc.CallingDevice N1			
The NID has connected in the call.	OriginatedEvent • originatedConnection N1C1 • callingDevice D1 • calledDevice D2 • localConnectionState Connected • cause newCall • assoc.CallingDevice N1			

Activity	Monitored Device N1 (trunk)	Monitored Device D2 (ACD)	Monitored Device D3 (agent).	Comments
The call arrives at a distribution device (ACD).	DeliveredEvent • alertingConnection D2C1 • alertingDevice D2 • callingDevice D1 • calledDevice D2 • networkCallingDevice D1 • networkCalledDevice D2 • lastRedirectionDevice NR • localConnectionState Connected • eventCause enteringDistri-bution • assoc.CallingDevice N1	DeliveredEvent • alertingConnection D2C1 • alertingDevice D2 • callingDevice D1 • calledDevice D2 • networkCallingDevice D1 • networkCalledDevice D2 • lastRedirectionDevice NR • localConnectionState Alerting • eventCause enteringDistri-bution • assoc.CallingDevice N1		
There are no available devices (agents) associated with the distribution device (D2) - the call will be queued.	QueuedEvent • queuedConnection D2C1 • queue D2 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice NR • localConnectionState Connected • eventCause noAvailable-Agents • networkCallingDevice D1 • networkCalledDevice D2 • assoc.CallingDevice N1	QueuedEvent • queuedConnection D2C1 • queue D2 • callingDevice D1 • calledDevice D2 • lastRedirectionDevice NR • localConnectionState Queued • eventCause noAvailable-Agents • networkCallingDevice D1 • networkCalledDevice D2 • assoc.CallingDevice N1		Device D2 may identify an ACD group or the ACD queueing mechanism. A call may also be queued for multiple devices (not shown).
Agent D3 becomes available.			AgentReadyEvent • agentDevice D3	
The call leaves the ACD Distri-bution device.	DivertedEvent • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Connected • eventCause distributed • networkCallingDevice D1 • networkCalledDevice D2 • assoc.CallingDevice N1	DivertedEvent • divertingConnection D2C1 • divertingDevice D2 • newDestination D3 • lastRedirectionDevice NR • localConnectionState Null • eventCause distributed • networkCallingDevice D1 • networkCalledDevice D2 • assoc.CallingDevice N1		This illustrates the CSTA modeling option (as specified via the capability exchange services) where the Diverted event is being sent to all devices in the call, not just for the diverting device (D2) monitor. An announcement can be provided prior to this event for which events may be generated (not shown).

Activity	Monitored Device N1 (trunk)	Monitored Device D2 (ACD)	Monitored Device D3 (agent).	Comments
The call is delivered to the available agent device (D3).	DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice D2 • networkCallingDevice D1 • networkCalledDevice D2 • localConnectionState Connected • eventCause distributed • assoc.CallingDevice N1		DeliveredEvent • alertingConnection D3C1 • alertingDevice D3 • callingDevice D1 • calledDevice D3 • lastRedirectionDevice D2 • networkCallingDevice D1 • networkCalledDevice D2 • localConnectionState Alerting • eventCause distributed • assoc.CallingDevice N1	

17 Advanced Conferencing Scenarios

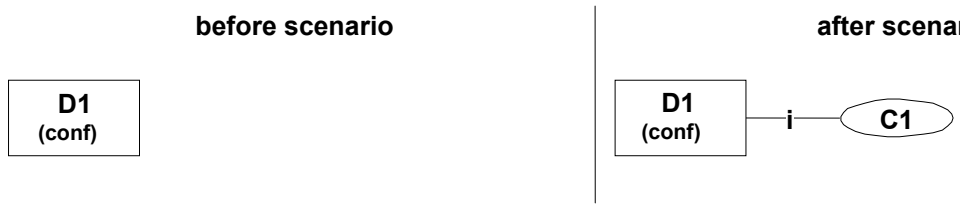
This clause illustrates calls at devices that are designed to host conference calls with a large number of participants (i.e. conference devices). The Monitored Device(s) columns for the participant devices are not shown in the tables.

17.1 Creating and enabling a conference

17.1.1 Creating a conference

The conference call is created at the conference device D1 by invoking Make Connection service. As a result the Connection D1C1 identifies the conference and may be used by the Computing Function to identify the conference to potential future participants.

In this scenario, the service's autoOriginate parameter is set to 'prompt' causing the call not to be auto-originated (the call does not implicitly reach the connected state). This scenario may be used in cases where conferences need to be planned and be propagated to potential participants, but where it is not yet necessary to attach resources (e.g. media).

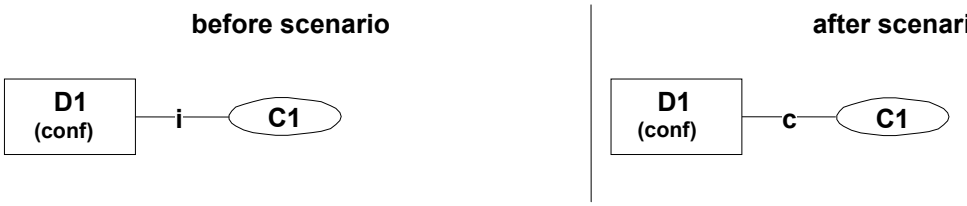


Activity	Monitored Device D1			Comments
A Make Connection service is invoked to create a connection that models the conference.	MakeConnectionRequest • initiatingDevice D1 • autoOriginate Prompt			
Acknowl edgment.	MakeConnectionResult • initiatingDevice D1C1			
The conference call is initiated but not yet enabled (i.e originated).	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatedDevice D1 • localConnectionState Initiated • cause makeConnection			The generation of this event is switch specific. In this scenario, the Switching Function is aware that D1 is a conference device and C1 is a conference call. Therefore, 'conference' is provided as cause code.

17.1.2 Enabling a conference

After a conference has been created according to scenario 17.1.1, the call is not auto-originated and is prompting.

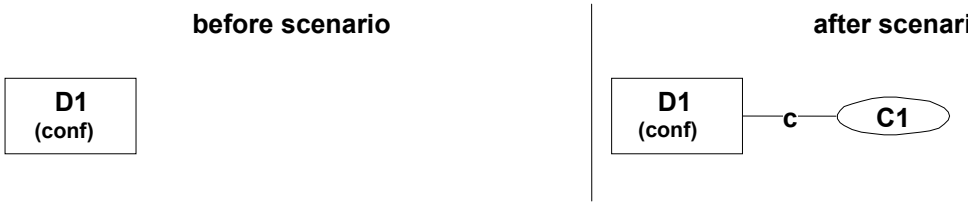
In the follow-up scenario shown here, the Computing Function requests the transition of connection D1C1 to connected state by issuing an Answer Call service. This transition enables the conference (e.g. the Switching Function may attach media to the call now).



Activity	Monitored Device D1			Comments
Answer Call service is invoked to enable the conference.	AnswerCallRequest • callToBeAnswered D1C1			
Acknowledge-ment.	AnswerCallResult			
Event indicates conference is enabled.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice NR • localConnectionState Connected • cause conference			

17.1.3 Creating and enabling a conference in one step

In this scenario, the conference call is created (as in 17.1.1) and enabled (as in 17.1.2) in one step by auto-originating the call while creating D1C1. In contrast to 17.1.1, the call C1 originates without prompting.

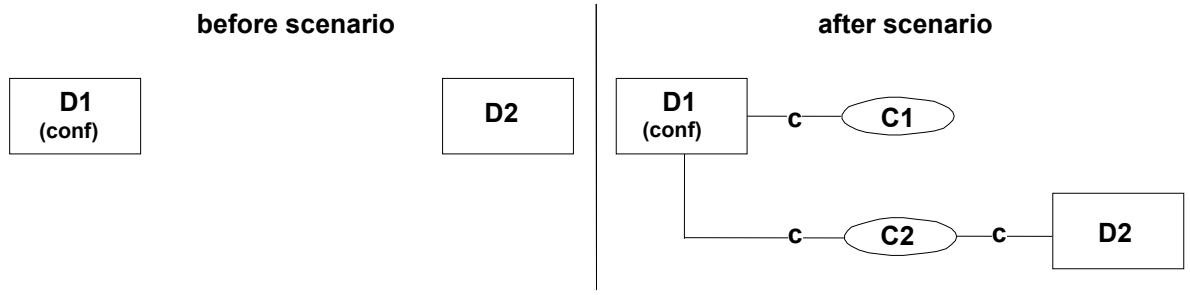


Activity	Monitored Device D1			Comments
A Make Connection service is invoked to create a connection that models the conference.	MakeConnectionRequest • initiatingDevice D1 • autoOriginate doNot-Prompt			
Acknowl edgment.	MakeConnectionResult • initiatingDevice D1C1			
The conference call is initiated but not yet enabled (i.e originated).	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatedDevice D1 • localConnectionState Initiated cause conference			The generation of this event is switch specific. In this scenario, the Switching Function is aware that D1 is a conference device and C1 is a conference call. Therefore, 'conference' is provided as cause code.
Event indicates conference is enabled.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice NR • localConnectionState Connected • cause conference			

17.1.4 Conference is being created and enabled implicitly

In this scenario the conference does not exist when the first participant is trying to enter it. The Switching Function implicitly creates and enables the conference when the first participant joins. This scenario is applicable for conference devices that allow participants to start a conference just by dialing into the conference device.

Scenario17.2.8 shows a possible continuation of this scenario.

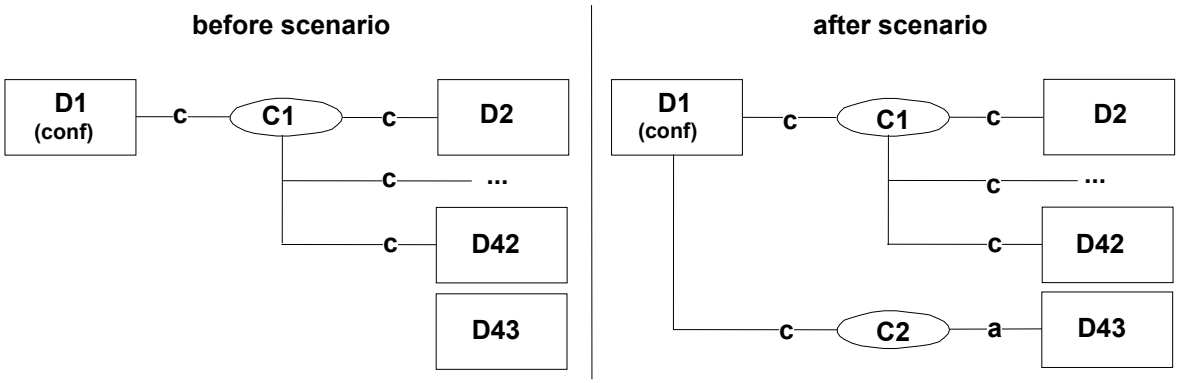


Activity	Monitored Device D1	Monitored Devices D2 (not shown)		Comments
D2 dials into the conference device D1.	DeliveredEvent • connection D1C2 • alertingDevice D1 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice NR • localConnectionState Alerting • cause newCall			
The call is being answered implicitly.	EstablishedEvent • establishedConnection D1C2 • answeringDevice D1 • callingDevice D2 • calledDevice D1 • lastRedirectionDevice NR • localConnectionState Connected • cause newCall			
The conference connection D1C1 has not been created yet and will be created implicitly now.	ServiceInitiatedEvent • initiatedConnection D1C1 • initiatedDevice D1 • localConnectionState Initiated • cause conference			D1C1 models the conference. Note that C2 is created before C1 in this scenario. The generation of this event is switch specific. In this scenario, the SF is aware that D1 is a conference device and C is a conference call. Therefore, 'conference' is provided as cause code.
D1 implicitly connects the call.	OriginatedEvent • originatedConnection D1C1 • callingDevice D1 • calledDevice NR • localConnectionState Connected • cause conference			

17.2 Conference Population

17.2.1 Inviting a party

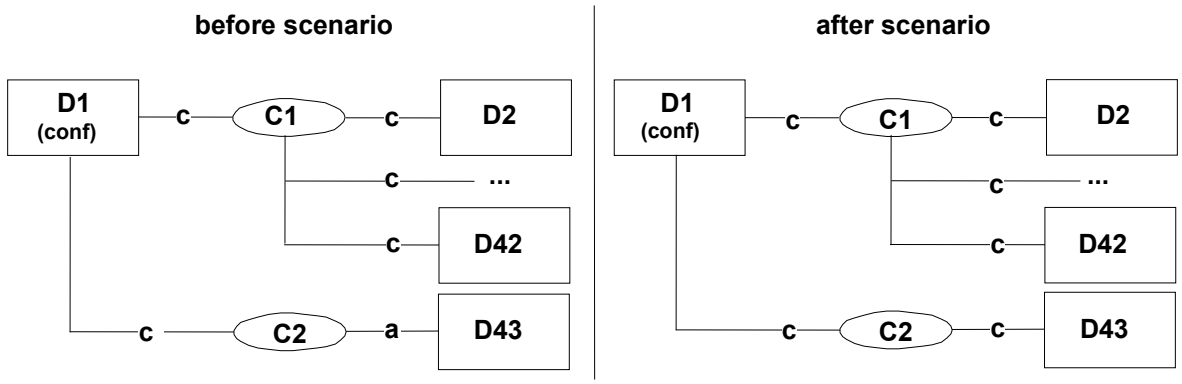
This scenario starts with a conference already hosting 42 participants. The conference device calls D43 to move it into the conference call C1 later. Scenarios 17.2.5 and 17.2.6 show possible continuations.



Activity	Monitored Device D1	Monitored Devices D2 to D43(not shown)		Comments
A Make Call service to a valid device is invoked on behalf of conference device D1.	MakeCallRequest • callingDevice D1 • calledDirectoryNumber D43 • autoOriginate DoNot-Prompt			
Acknowl edgement.	MakeCallResult • initiatedCall D1C2			
Indication that the service has been initiated from this device.	ServiceInitiatedEvent • initiatedConnection D1C2 • initiatedDevice D1 • localConnectionState Initiated • cause newCall			The generation of this event is switch specific.
Conference device D1 is connected to the call.	OriginatedEvent • originatedConnection D1C2 • callingDevice D1 • calledDevice D43 • localConnectionState Connected • cause newCall			
Device D43 begins to ring.	DeliveredEvent • connection D43C2 • alertingDevice D43 • callingDevice D1 • calledDevice D43 • lastRedirectionDevice NR • localConnectionState Connected • cause newCall			

17.2.2 Invited party answers a call

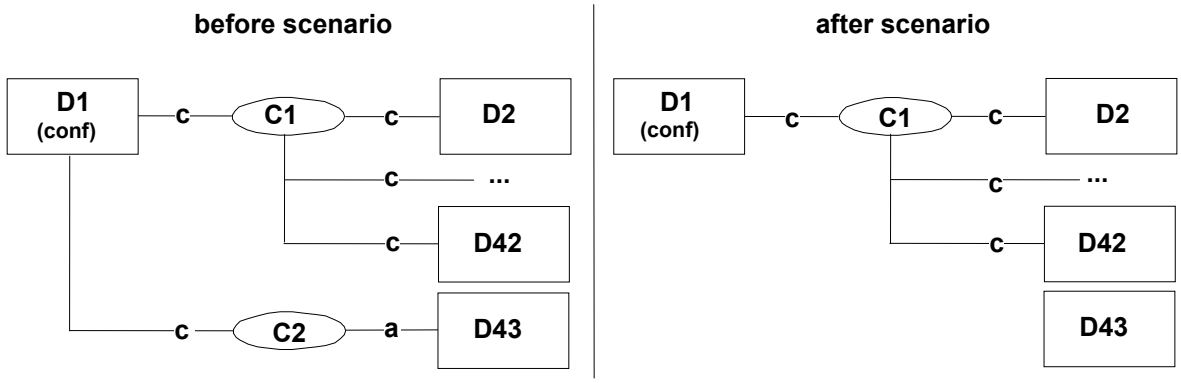
This scenario shows the invited party accepting the invitation by answering the call. Scenarios 17.2.5 and 17.2.6 show how the invited party may be moved into the ongoing conference D1 later.



Activity	Monitored Device D1	Monitored Devices D2 to D43 (not shown)		Comments
Device D43 answers the call.	EstablishedEvent • establishedConnection D43C2 • answeringDevice D43 • callingDevice D1 • calledDevice D43 • lastRedirectionDevice NR • localConnectionState Connected • cause newCall			

17.2.3 Conference invitation is explicitly cancelled

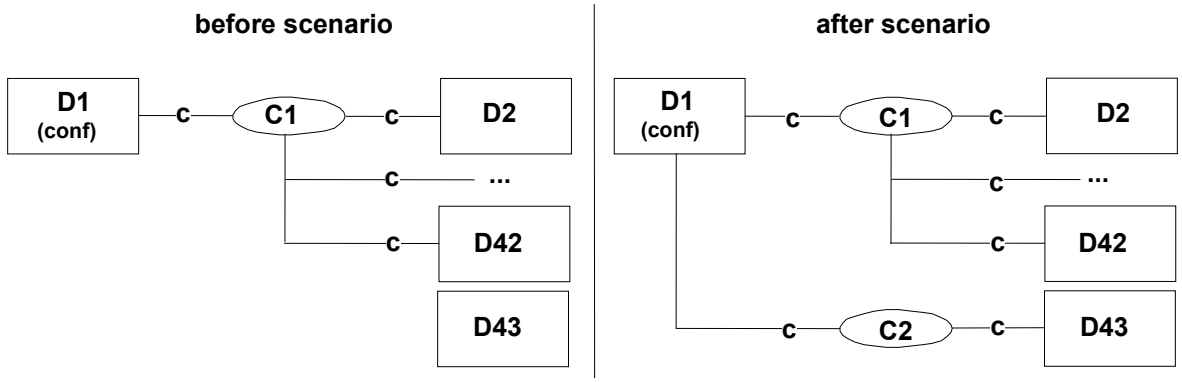
This scenario shows how the Computing Function cancels an invitation for a party that had not answered the invitation yet.



Activity	Monitored Device D1	Monitored Devices D2 to D43 (not shown)		Comments
Clear Connection service is invoked to cancel the conference invitation.	ClearConnectionRequest • connectionToBeCleared D1C2			
Acknowledgement.	ClearConnectionResult			
Event indicates that connection D1C2 has been removed from the call.	ConnectionClearedEvent • droppedConnection D1C2 • releasingDevice D1 • localConnectionState Null • cause normal-Clearing			

17.2.4 Party dials into the conference

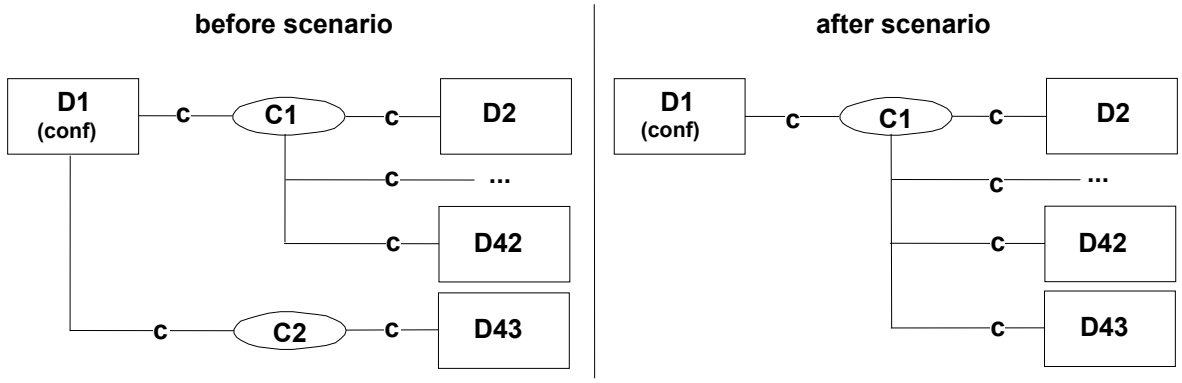
In contrast to scenario 17.2.1, this scenario shows how a participant dials into the conference on its own behalf instead of being invited.



Activity	Monitored Device D1	Monitored Devices D2 to D43(not shown)		Comments
Call arrives at the conference device D1.	DeliveredEvent • connection D1C2 • alertingDevice D1 • callingDevice D43 • calledDevice D1 • lastRedirectionDevice NR • localConnectionState Alerting • cause newCall			
Conference device implicitly answers the call.	EstablishedEvent • establishedConnection D1C2 • answeringDevice D1 • callingDevice D43 • calledDevice D1 • lastRedirectionDevice NR • localConnectionState Connected • cause newCall			

17.2.5 Party joins the conference (implicit conferencing)

This scenario shows how a party is moved into a conference implicitly (e.g. because certain preset criteria are satisfied). Prior to this scenario, the party was either invited as in scenario 17.2.2 or dialled into the conference as in scenario 17.2.4.

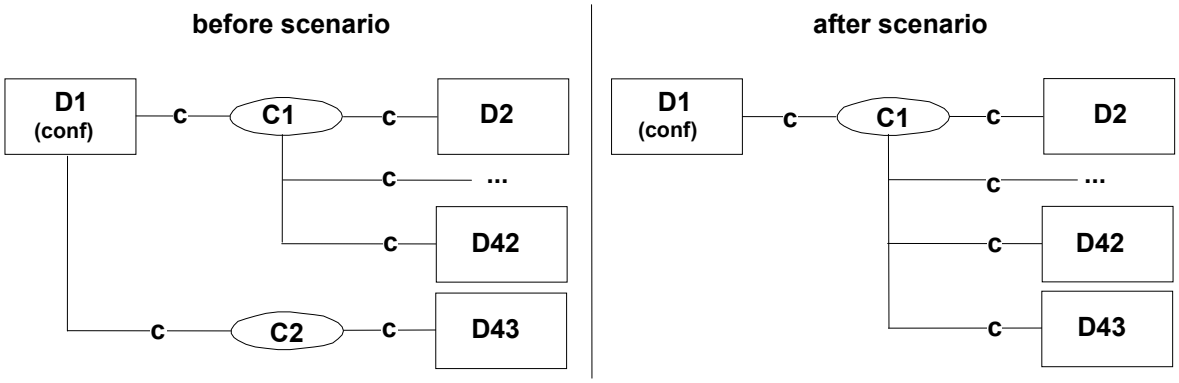


Activity	Monitored Device D1	Monitored Devices D2 to D43(not shown)		Comments
Party implicitly joins the conference.	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • conferencingDevice D1 • addedDevice D43 • conferenceConnection D1C1 • conferenceConnection … • conferenceConnection D1C42 • conferenceConnection D1C43 • localConnectionInfo Connected • cause conference			In this scenario, the Switching Function reuses C1 as the Call ID of the resulting conference call

17.2.6 Party joins the conference (explicit conferencing)

This scenario shows how a party is moved into a conference explicitly. Prior to this scenario, the party was either invited as in scenario 17.2.2 or dialled into the conference as in scenario 17.2.4.

This scenario is similar to scenario 17.2.5.

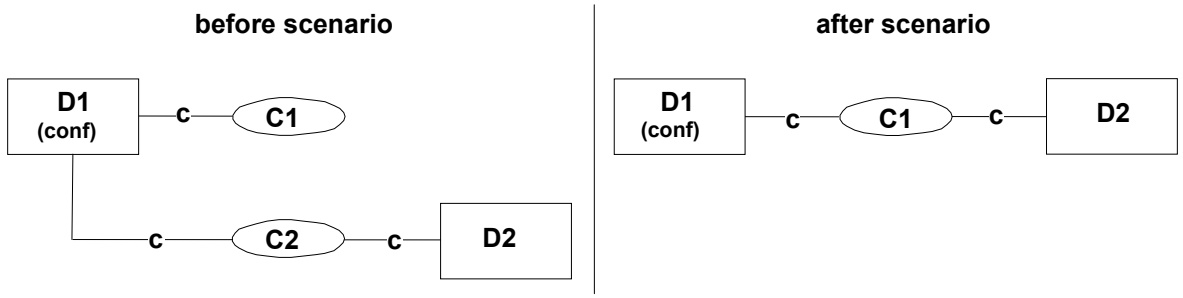


Activity	Monitored Device D1	Monitored Devices D2 to D43(not shown)		Comments
Conference Call service is invoked to add the party to the conference.	ConferenceCallRequest • heldCall D1C2 • activeCall D1C1			Please note that the held call is in connected state in this scenario.
Acknowl edgement.	ConferenceCallResult • conferenceCall D1C1			
Party joins the conference.	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • conferencingDevice D1 • addedDevice D43 • conferenceConnection D1C1 • conferenceConnection … • conferenceConnection D1C42 • conferenceConnection D1C43 • localConnectionInfo Connected • cause conference			In this scenario, the Switching Function reuses C1 as the Call ID of the resulting conference call

17.2.7 First party joins the conference (explicit conferencing)

This scenario is similar to the previous scenario 17.2.6, but in this case, the conference is not populated yet. This scenario results in C1 having two connections D1C1 and D2C1. Since C1 is a conference call Conferenced events are signalled instead of Established events.

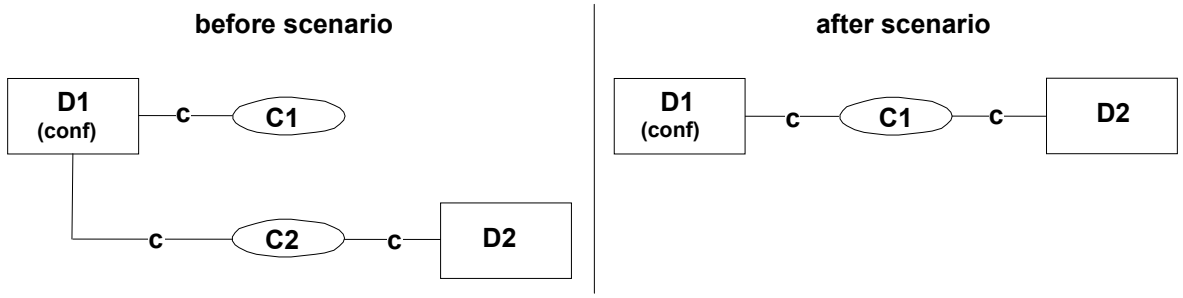
When a conference has only one participant, the Conference Device may play media streams to the participant by attaching the call to a media service instance. This may lead to additional events that are not shown here.



Activity	Monitored Device D1	Monitored Device D2 (not shown)		Comments
Conference Call service is invoked to add the party to the conference.	ConferenceCallRequest • heldCall D1C2 • activeCall D1C1			Please note that the held call is in connected state in this scenario
Acknowl edgement.	ConferenceCallResult • conferenceCall D1C1			
First party joins the conference.	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • conferencingDevice D1 • addedDevice D2 • conferenceConnection D1C1 • conferenceConnection D1C2 • localConnectionInfo Connected • cause conference			In this scenario, the Switching Function reuses C1 as the Call ID of the resulting conference call.

17.2.8 First party joins the conference (implicit conferencing)

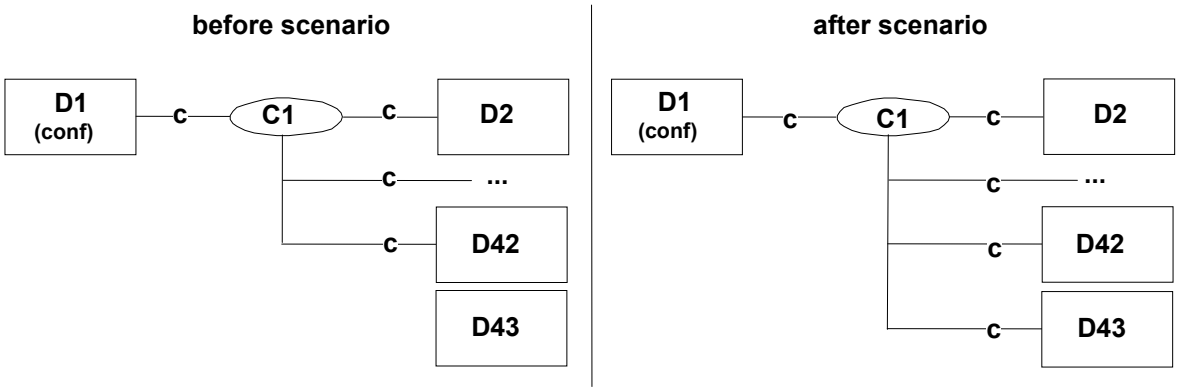
This scenario is similar to the previous scenario 17.2.7, but in this case, the conference is joined implicitly.



Activity	Monitored Device D1	Monitored Device D2 (not shown)		Comments
First party joins the conference.	ConferencedEvent • primaryOldCall D1C1 • secondaryOldCall D1C2 • conferencingDevice D1 • addedDevice D2 • conferenceConnection D1C1 • conferenceConnection D1C2 • localConnectionInfo Connected • cause conference			In this scenario, the Switching Function reuses C1 as the Call ID of the resulting conference call.

17.2.9 Inviting a party and moving it into the conference unconditionally (in one step)

In this scenario the conference device calls D43 and moves it into the conference call C1 using Single Step Conference Call service.

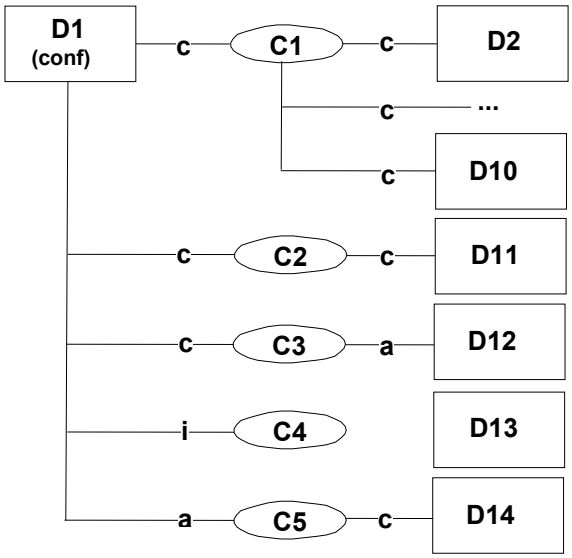


Activity	Monitored Device D1	Monitored Devices D2 to D43 (not shown)		Comments
Single Step Conference Call service is invoked on behalf of the conference device.	SingleStepConferenceCallRequest • activeCall D1C1 • deviceToJoin D43			
Acknowledge-ment	SingleStepConferenceCallResult • conferencedCall D43C1			
D43 has been added to the conference.	ConferencedEvent • primaryOldCall D1C1 • secondaryOldcall NR • conferencingDevice D1 • addedDevice D43 • conferenceConnection D1C1 • conferenceConnection D2C1 • ... • conferenceConnection D42C1 • conferenceConnection D43C1 • localConnectionState Connected • eventCause singleStep-Conference			

17.2.10 Multiple parties joining the conference at the same time

Several participants may be invited to the conference or dial into the conference at the same time. The figure on the right shows a scenario at a point in time where

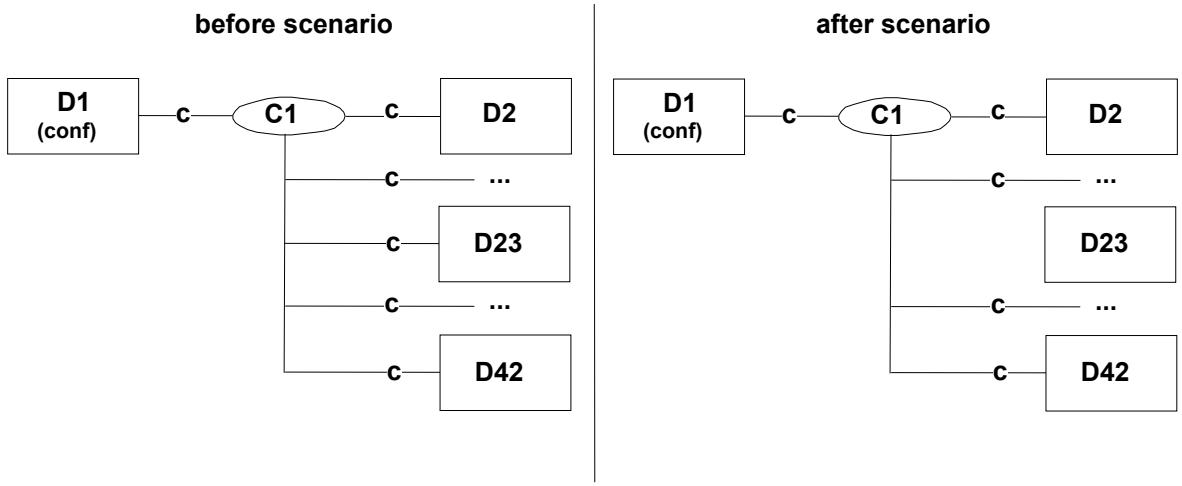
- D2 to D10 are already participating in the conference
- D11 has been invited (or dialed in) but not yet moved into the conference
- D12 has been invited but has not answered the invitation call yet
- D13 is about to be invited (the invitation call is initiated but not yet originated)
- D14 is dialing into the conference and the conference device has not answered the call yet



17.3 Conference depopulation

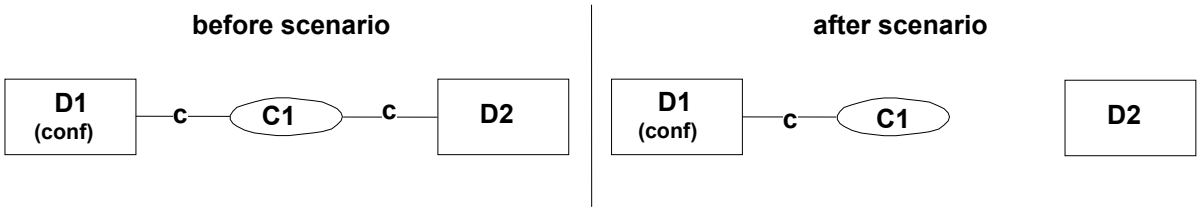
17.3.1 A participant leaves the conference

In this scenario D23 leaves a conference with 41 participants (D2 to D42).



Activity	Monitored Device D1	Monitored Devices D2 to D42 (not shown)		Comments
D23 leaves the conference.	ConnectionClearedEvent • droppedConnection D23C1 • releasingDevice D23 • localConnectionState Connected • cause normal-Clearing			

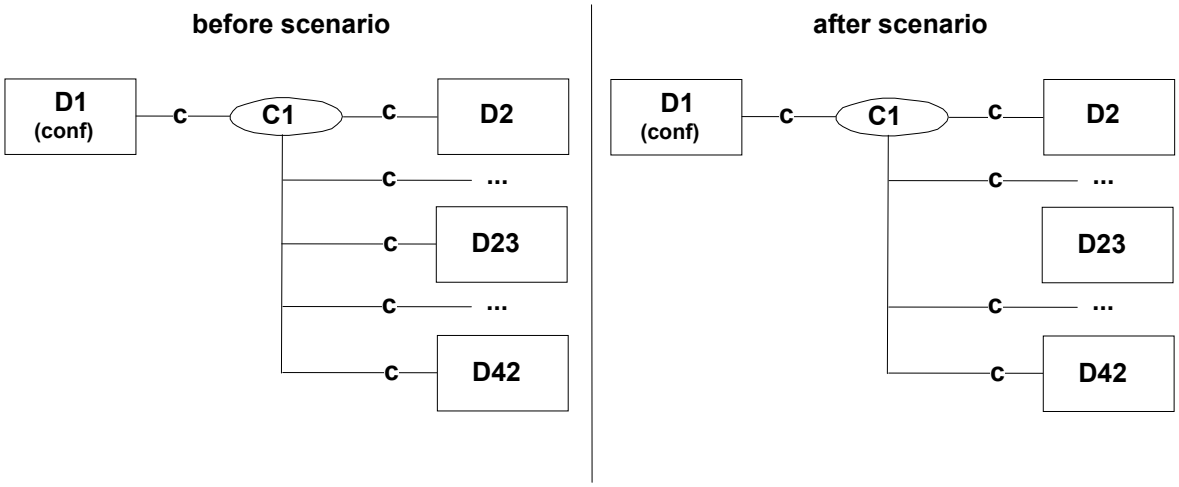
17.3.2 Last participant leaves the conference



Activity	Monitored Device D1	Monitored Device D2 (not shown)		Comments
D2 leaves the conference.	ConnectionClearedEvent • droppedConnection D2C1 • releasingDevice D2 • localConnectionState Connected • cause normal-Clearing			Some switches may suspend the conference after the last participant left it. Scenario 17.5illustrates the event sequence for that case.

17.3.3 A participant is removed from the conference by the Computing Function

In this scenario D23 is removed from a conference with 41 participants (D2 to D42).

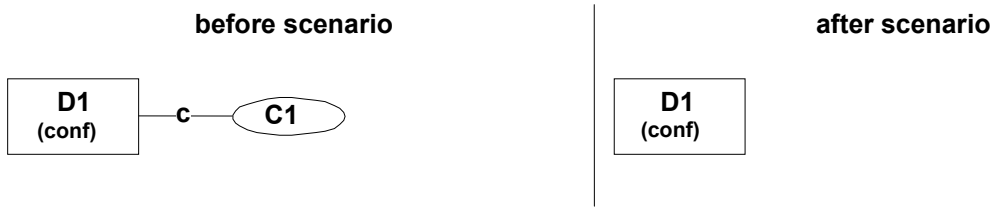


Activity	Monitored Device D1	Monitored Devices D2 to D42 (not shown)		Comments
A Clear Connection service is invoked to remove D23 from the conference call.	ClearConnectionRequest • connectionToBeCleared D23C1			
Acknowledge-ment	ClearConnectionResult			
Event indicates that connection D23C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection D23C1 • releasingDevice D23 • localConnectionState Connected • cause normal-Clearing			

17.4 Releasing a conference

17.4.1 Releasing an empty conference (without participants)

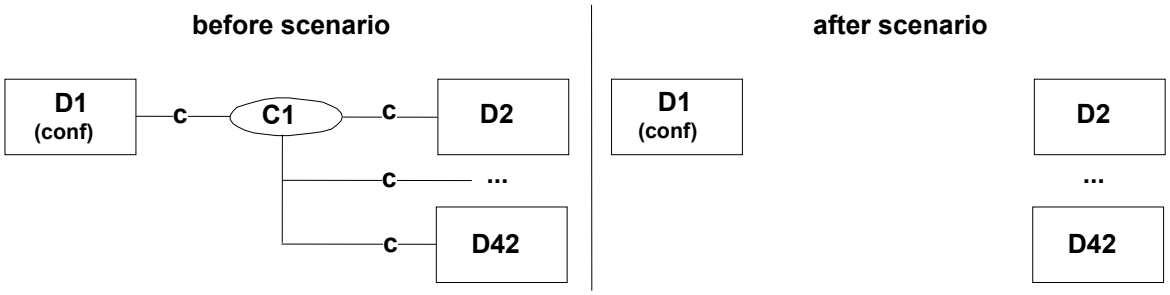
Using Clear Connection service the Computing Function requests the conference to be released (i.e. the connection D1C1 to be cleared).



Activity	Monitored Device D1			Comments
The Computing Function invokes Clear Connection service to release the conference.	ClearConnectionRequest • connectionToBeCleared D1C1			
Acknowledge-ment	ClearConnectionResult			
Event indicates that connection D1C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection D1C1 • releasingDevice D1 • localConnectionState Null • cause normal-Clearing			

17.4.2 Releasing a populated conference (with devices participating in the conference call)

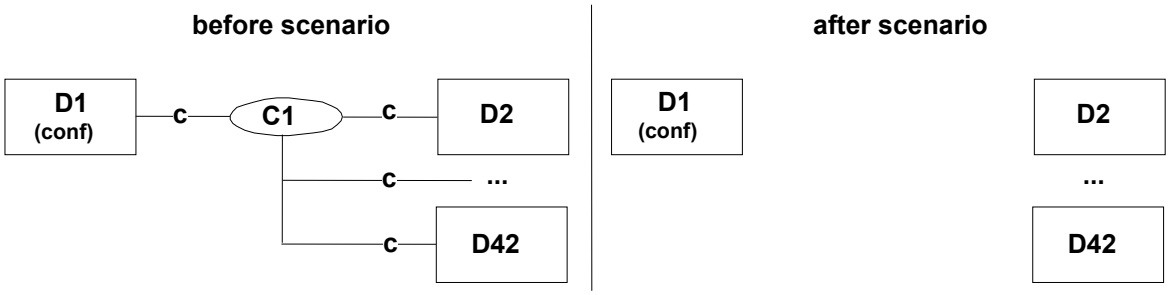
This scenario is identical to 17.4.1 with the exception that this time there are devices still participating in the conference when the conference is being released.



Activity	Monitored Device D1	Monitored Devices D2 to D42 (not shown)		Comments
The Computing Function invokes Clear Connection service to release the conference.	ClearConnectionRequest • connectionToBeCleared D1C1			
Acknowledge-ment	ClearConnectionResult			
Event indicates that connection D1C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection D1C1 • releasingDevice D1 • localConnectionState Null • cause normal-Clearing			

17.4.3 Implicitly releasing a populated conference

A conference may even be released implicitly, e.g. if the Switching Function releases a conference after a certain duration or if other preset criteria are fulfilled. The events in this scenario do not differ from 17.4.2.



Activity	Monitored Device D1		Monitored Devices D2 to D42 (not shown)		Comments
Event indicates that connection D1C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	 D1C1 D1 Null normal-Clearing			

17.5 Suspending and resuming a conference

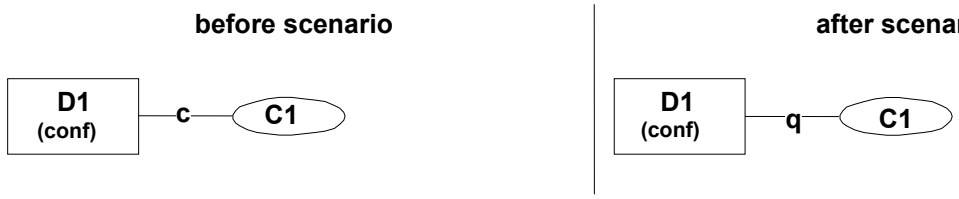
Instead of being released as in scenarios of 17.4, a conference may be suspended temporarily to be resumed later.

In suspend/resume scenarios the connection D1C1 (which identifies the conference) does not change. This is different to releasing the conference and creating a new one.

Suspending may be more convenient than releasing because already propagated conference parameters remain valid. Suspending may preserve conference resources (like media).

17.5.1 Implicitly suspending a conference

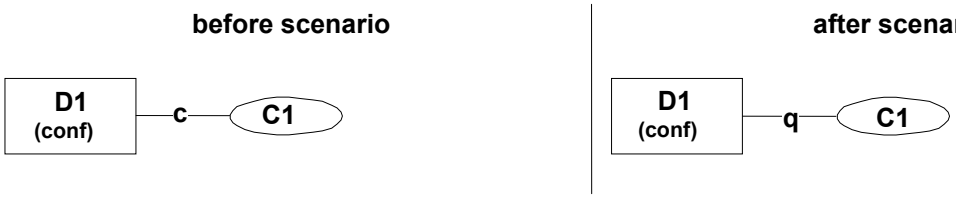
In the following scenario the Switching Function implicitly suspends the conference. The last participant leaving a conference would be a typical trigger for the implicit suspension of the conference.



Activity	Monitored Device D1				Comments
The conference is suspended implicitly.	QueuedEvent • queuedConnection • queue • callingDevice • calledDevice • localConnectionState • cause	 D1C1 D1 D1 NR Queued park			

17.5.2 Explicitly suspending a conference using Park Call service

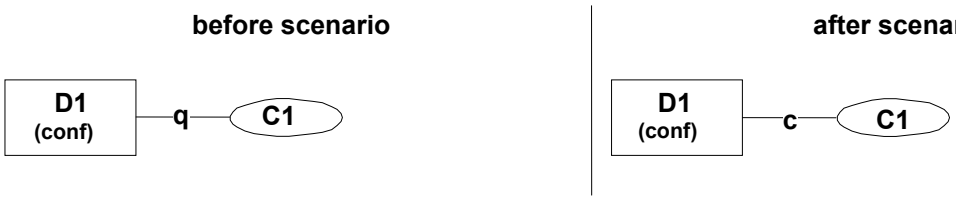
In this scenario the Computing Function explicitly requests the conference to be suspended by issuing Park Call service.



Activity	Monitored Device D1			Comments
Park Call service is invoked to suspend the conference.	ParkCallRequest • parking D1C1 • parkTo D1			
Acknowledge-ment.	ParkCallResult • parkedTo D1C1			
Event indicates conference suspension.	QueuedEvent • queuedConnection D1C1 • queue D1 • callingDevice D1 • calledDevice NR • localConnectionState Queued • cause park			

17.5.3 Resuming a suspended conference

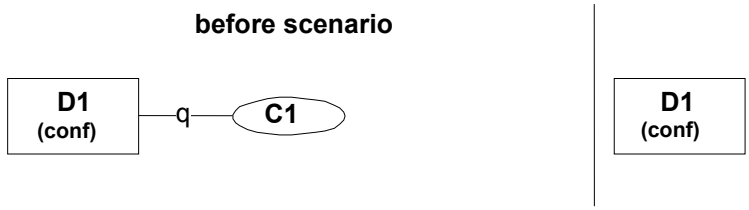
A conference will be resumed by the Computing Function using Answer Call service.



Activity	Monitored Device D1			Comments
Answer Call service is invoked to resume the conference.	AnswerCallRequest • callToBeAnswered D1C1			
Acknowledge-ment.	AnswerCallResult			
Event indicates conference resumation.	EstablishedEvent • establishedConnection D1C1 • answeringDevice D1 • callingDevice NR • calledDevice D1 • lastRedirectionDevice NR • localConnectionState Connected • cause conference			

17.5.4 Releasing a suspended conference

This scenario shows how a suspended conference is being released (instead of being resumed) using Clear Connection service.



Activity	Monitored Device D1			Comments
Clear Connection service is invoked to release the suspended conference.	ClearConnectionRequest • connectionToBeCleared	D1C1		
Acknowledge-ment	ClearConnectionResult			
Event indicates that connection D1C1 has been removed from the call.	ConnectionClearedEvent • droppedConnection • releasingDevice • localConnectionState • cause	D1C1 D1 Null normal-Clearing		