Panasonic®

DBS

Digital Business System

Section 470

DBS Telephony Services

Programmer's Guide

DBS - Version 9.2
Issued April 2000
Doc. No. 570X01001

Contents

Chapter 1. Introduction

The intent of this document is to provide in-depth information on Panadrvr™, the Panasonic© PBX Driver Netware Loadable Module (NLM) designed to communicate with the Novell Tserver NLM. This information will allow TSAPI application developers to design customized applications for interfacing Panasonic DBS telephone systems with the Novell Netware® Telephony Services environment.

Users of this manual should have a working knowledge of CSTA architecture and services and with the Panasonic DBS telephone system. The following documents may be of assistance.

- Standard ECMA-179 Services for Computer-Supported Telecommunications Applications (CSTA)
  European Computer Manufacturer's Association, June 1992
- Standard ECMA-180 Protocol for Computer-Supported Telecommunications Applications (CSTA)
  European Computer Manufacturer's Association, June 1992
- Computer-Supported Telecommunications Applications ECMA TR/52, European Computer Manufacturer's Association, June 1990
- Novell Netware® Telephony Services: PBX Driver Interface Specification
- Panasonic DBS Section 520: DBS Telephony Services Installation and Feature Description
- Panasonic DBS Section 300: Installation Manual
- Panasonic DBS Section 400: Programming Guide
- Panasonic DBS Section 700: Feature Operation

Panasonic DBS TSAPI Overview

The essence of Computer Telephony Integration (CTI) is the ability for computing and switching networks to use the capabilities of the other. For instance, a computer can control call distribution on the switch network, routing calls to the most appropriate agent based on the caller's ANI, while automatically displaying the caller's account record or customer profile on the agent's computer screen. On the other hand, the switch network can use a database management system to automatically dial a telephone number associated with an account record. For this to occur, both the computer network and the switch network must use a common protocol to communicate with each other. The standard which makes this possible is called CSTA, or Computer-Supported Telecommunications Applications. Both the Novell Telephony Server NLM and the Panasonic PBX NLM (Panadrvr) are designed to CSTA standards.

Panadrvr performs the following functions:

- Receives CSTA telephony requests from the Tserver NLM and translates them into DBS-specific protocol requests before sending them on to the DBS.
- Communicates with the DBS via a serial link to the Panasonic API card.
- Receives requests, responses, and events from the DBS and translates them into the appropriate CSTA messages before sending on to the Tserver NLM.

- Provides an application programming interface for DBS-specific administration and maintenance.

Supported CSTA Service Groups

CSTA services are grouped according to their function. Panadrvr supports the following CSTA service groups:

Name	Description
Call Control Service Group	enables a telephony client application to control a call or connection on the DBS.
Set Feature Service Group	allows a client application to set switch-controlled features and parameters on a DBS device.
Query Service Group	allows a client application to query the switch for device features and static attributes of a device.
Monitor Service Group	allows a client application to request and cancel the reporting of state-changing events.
Event Report Service Group	provides a client application with reports of state-changing events to a call, a connection, or a device.

These service groups are described in detail on the following pages.

Chapter 2. Call-Control Service Group

Overview

Services in this group enable a telephony client application to control a call or connection on the DBS. Some examples are placing calls from a device and controlling connections on a call in progress in the DBS. Panadrvr Release 1.0 supports the following Call-Control Services:

- Answer Call Service
- Clear Call Service
- Clear Connection Service
- Conference Call Service
- Hold Call Service
- Make Call Service
- Retrieve Call Service
- Transfer Call Service

The following Call-Control Services are **not** supported by Panadrvr Release 1.0.

- Alternate Call Service
- Consultation Call Service
- Make Predictive Call Service
- Reconnect Call Service

Functional Descriptions

The following pages contain functional descriptions of each Call-Control Service item, along with illustrations depicting conditions before and after a successful service request. Please note the following conventions:

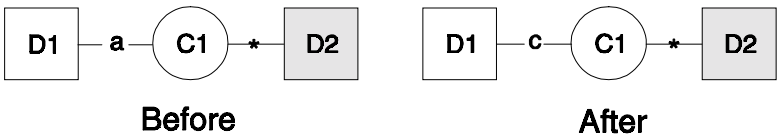
- D1..DX represent deviceIDs.
- Circles represent calls and C1, C2, and C3 represent callIDs.
- Lines represent connections between a call and a device and C1-D1, C1-D2, C2-D3, etc. represent connectionIDs.
- Absence of a line is equivalent to a connection in the Null connection state.
- Labels in boxes and circles represent call and device instances.
- Labels on lines represent a connection state using the following key:
  a = Alerting
  c = Connected
  f = Failed
  h = Held
  i = Initiated
  q = Queued
  a/h = Alerting or Held

* = Unspecified

- Grayed boxes represent devices in a call unaffected by the service or event report.
- White boxes and circles represent devices and calls affected by the service or event report.
- The parameters for the function call of the service are indicated in bold italics.

Answer Call Service

The Answer Call Service is used when an incoming call (C1) is alerting (ringing) a device (D1) with the connection ***alertingCall*** (C1-D1). The effect is the same as if the answering party answered the call via the telephone on/off button.



Function: ***cstaAnswerCall( ), CSTAAnswerCallConfEvent***
Direction: C → S
Functional Description:

The Answer Call Service allows a client application to request that a call ringing at a station be answered. Answering a ringing call means to connect a call if the user is on-hook by forcing the station off-hook, or, if the user is off-hook, by cutting through the call to the headset or handset. An active call may be dropped or placed on hold when the new call is answered (depending on how the extension is programmed).

The deviceID in ***alertingCall*** must contain the station extension of the endpoint to be answered on the call. A Delivered Event Report must have been received by the application prior to this request.

The Answer Call Service can be used to answer calls presented to digital telephones only.

The Answer Call Service request is acknowledged (Ack) by the switch if the switch is able to connect the specified call either by forcing the station off-hook (i.e., turning the speakerphone on) or by taking the appropriate action required to handle an existing call. Answering a call which is already connected will result in a positive acknowledgment and the call will remain connected.

Service Parameters:

alertingCall	[mandatory] a valid connection identifier indicates the callID and the station extension (STATIC_ID).

Ack Parameters:

noData	None for this service.

Nak Parameter:
universalFailure

- INVALID_CSTA_CONNECTION_IDENTIFIER (13)
  An incorrect callID, or an incorrect deviceID is specified.
- INVALID_OBJECT_STATE (22)
  The specified connection at the station is not in the alerting or connected.

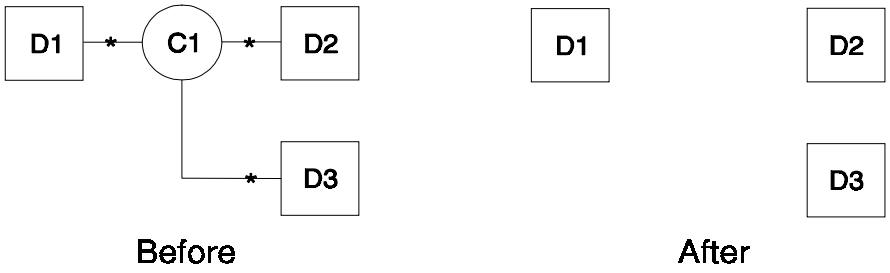
- NO_CALL_TO_ANSWER (28)
  The specified connection at the station is not alerting.
- MISTYPED_ARGUMENT_REJECTION (74)
  DYNAMIC_ID is specified in ***alertingCall***.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- GENERIC_UNSPECIFIED (0)
  An internal driver error has occurred.

Notes:

- If the station user is on-hook idle, the switch will turn on the speaker/headset and answer the call.
- If the station user is busy on a call, the current call with either be put on hold or dropped depending on the "key-bank-hold" option setting.

Clear Call Service

This service will cause each device associated with a ***call*** (C1) to be released. The conditions applied to individual extensions concerning the Clear Connection Service apply to each connection in the call.



Function: ***cstaClearCall(), CSTAClearCallConfEvent***
Direction: C → S
Functional Description:
The Clear Call Service disconnects all connections from the specified call and terminates the call itself. All connection identifiers previously associated with the call are no longer valid. It should be noted that the DBS in itself does not support the cstaClearCall function -- instead the Panasonic Telephony Services driver converts the cstaClearCall function into individual cstaClearConnection functions. The driver makes every attempt to verify that all the cstaClearConnection commands will succeed; however in some cases (e.g., the extension is physically off-hook), the call will not be cleared even though a positive confirmation is returned to the application.

Service Parameters:

call	[mandatory] a valid connection identifier indicates the call to be cleared. The deviceID of ***call*** is optional. If it is specified, it is ignored.

Ack Parameters:

noData	None for this service.

Nak Parameter:

universalFailure

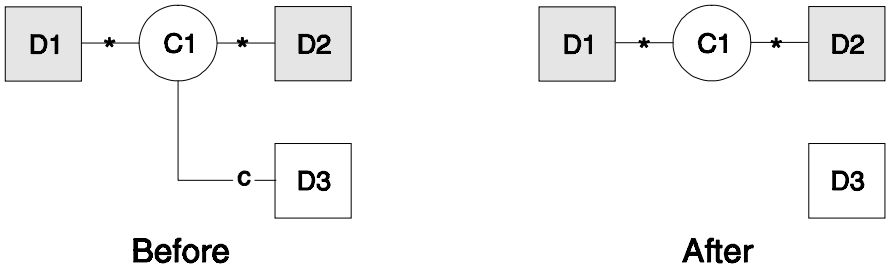
- NO_ACTIVE_CALL (24)
  The callID of the connectionID specified in the request is invalid.·
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.·
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- INVALID_OBJECT_STATE (22)
  1. One of the connections in the call has a call on hold.
  2. One on the connections in the call is being alerted by another monitored extension.

Notes:

- **Switch operation** - After a successful Clear Call Service request:
- Every station dropped will be in the on-hook idle state.
- Any lamps associated with the call are off.
- Displays are cleared.

Clear Connection Service

This service releases the specified connection, ***call*** (C1-D3), and its connectionID from the specified call (C1). The result is the same as if the user pressed the extension's on/off button during a hands-free conversation. The Clear Connection Service cannot be used if the extension is physically off-hook.



Function: ***cstaClearConnection(), CSTAClearConnectionConfEvent***
Direction: C → S
Private Parameters: ***userInfo***
Functional Description:

The Clear Connection Service disconnects the specified device from the designated call.

The connection is left in the Null state. The connection identifier is no longer associated with the call. The party to be dropped must be an extension.

A connection in the alerting or held state cannot be cleared.

Service Parameters:

call [mandatory] a valid connection identifier indicates the endpoint to be disconnected.

Ack Parameters:

noData None for this service.

Nak Parameter:

universalFailure

- INVALID_OBJECT_STATE (22)
  The specified connection at the station is not currently active (in alerting or held state) so it cannot be dropped or the station user is off-hook.
- NO_ACTIVE_CALL (24)
  The connectionID contained in the request is invalid. CallID may be incorrect.
- NO_CONNECTION_TO_CLEAR (27)
  The connectionID contained in the request is invalid. CallID may be correct, but deviceID is wrong.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- MISTYPED_ARGUMENT_REJECTION (74)
  DYNAMIC_ID is specified in ***alertingCall***.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

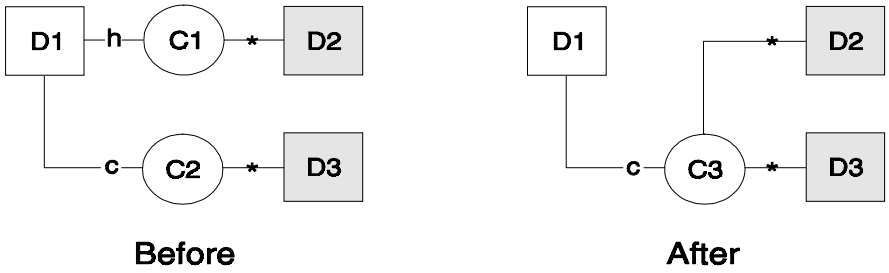
Notes:

- **Switch Operation** - When a party is dropped from an existing conference call with three or more parties (directly connected to the switch), the other parties remain on the call. If this was a two-party call, the non-dropped party is listening to busytone.

  Only connected parties can be dropped from a call. Held and alerting parties cannot be dropped by the Clear Connection Service.

Conference Call Service

This service provides the conference of an existing ***heldCall*** (C1-D1), and another ***activeCall*** (C2-D1) at the same device. The two calls are merged into a single call, (C3) and the two connections (C1-D1, C2-D1) at the conferencing device (D1) are resolved into a single connection, ***newCall*** (C3-D1), in the Connected state.



Function: ***cstaConferenceCall(), CSTAConferenceCallConfEvent***
Direction: C → S
Functional Description:

This service provides the conference of an existing held call (***heldCall***) and another active call (***activeCall***) at the controlling device. The two calls are merged into a single call and the two connections at the conference controlling device are resolved into a single connection in the connected state. The pre-existing CSTA connectionID associated with the device creating the conference are released, and a new callID for the resulting conferenced call is provided.

Service Parameters:

heldCall	[mandatory] must be a valid connection identifier for the call which is on hold at the controlling device and is to be conferenced with the ***activeCall***. The deviceID in ***heldCall*** must contain the station extension of the controlling device.
activeCall	[mandatory] must be a valid connection identifier for the call which is active at the controlling device and is to be conferenced with the ***heldCall***. The deviceID in ***activeCall*** must contain the station extension of the controlling device.

Ack Parameters:

newCall	[mandatory - partially supported] a connection identifier specifies the resulting new call identifier for the calls which were conferenced at the conference controlling device. This connection identifier replaces the two previous call identifiers at that device.
connList	[optional - supported] specifies the devices on the resulting newCall. This includes a count of the number of devices in the new call and a list of up to four connectionIDs and up to four deviceIDs which define each connection in the call. • If a device is on the DBS, the extension is specified. • If a party is off the DBS, then its assigned dynamic trunk identifier is specified.

Nak Parameter:

universalFailure

- INVALID_CSTA_CONNECTION_IDENTIFIER (13)
  The controlling deviceID in ***heldCall*** or ***activeCall*** has not been specified correctly.
- REQUEST_INCOMPATIBLE_WITH_OBJECT (2)
  The active call is alerting.
- NO_HELD_CALL (25)
  The held call parameter is incorrect.

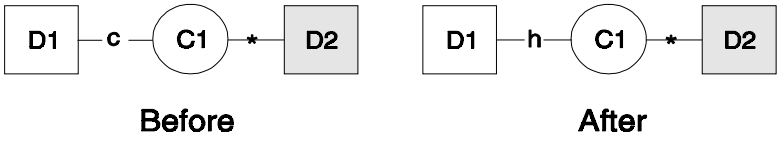
- NO_ACTIVE_CALL (24)
  The active call parameter is incorrect.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

- The active call must be in the connected state.
- A maximum of 4 parties can be conferenced.

Hold Call Service

The Hold Call Service places a call (C1) at a device (D1) with the connection ***activeCall*** (C1-D1) on hold. The effect is as if the specified party depressed the hold button on the device. This service maintains a relationship between the holding device (D1) and the held call (C1) that lasts until the call is retrieved from the hold status, or until the call is cleared.



Function: ***cstaHoldCall( ), CSTAHoldCallConfEvent***
Direction: C → S
Functional Description:
The Hold Call Service places a call on hold at a DBS station. The effect is as if the specified party depressed the hold button on their station to locally place the call on hold.

Service Parameters:

activeCall	[mandatory] a valid connection identifier indicates the connection to be placed on hold. This party must be in the active (talking) state or already held. The device associated with the ***activeCall*** must be a station. If the party specified in the request refers to a trunk device, the request will be denied. The deviceID in ***activeCall*** must contain the station extension of the controlling device.
reservation	[optional - not supported] specifies whether reserves the facility for reuse by the held call. The DBS switch does **not** support this parameter.

Ack Parameters:

noData	None for this service.

Nak Parameter:
universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  An invalid device identifier or extension is specified in ***activeCall***.

- INVALID_CSTA_CONNECTION_IDENTIFIER (13)
  The connection identifier contained in the request is invalid or does not correspond to a station.
- NO_ACTIVE_CALL (24)
  The party to be put on hold is not currently active (e.g., in alerting state) so it cannot be put on hold.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

- **Held State** - If the party is already on hold on the specified call when the switch receives the request, a positive request acknowledgment is returned.
- **Music on Hold** - Music on Hold (if administered and available) will be given to a trunk party placed on hold from the other end either manually or via the Hold Call Service.
- **Internal hold tones** - Internal hold tones (if administered) will be given to a trunk party placed on hold from the other end either manually or via the Hold Call Service.
- **Switch Operation** - After a party is placed on hold through a Hold Call Service request, the user will receive dial tone. Subsequent calls can be placed directly or through the Make Call Service request.

Make Call Service

The Make Call Service originates a call between two devices designated by the application. When the service is initiated, a call to the ***calledDevice*** (D2) is originated. A call is established as if D1 had called D2, and the client is returned with the connection ***newCall*** (C1-D1).

Function: ***cstaMakeCall(), CSTAMakeCallConfEvent***
Direction: C → S
Private Parameters: ***destRoute, userInfo***
Functional Description:

The Make Call Service originates a call between two devices. The service attempts to create a new call and establish a connection with the originating device (***callingDevice***). The Make Call Service also provides a connection identifier (***newCall)*** that indicates the connection of the originating device in the ***CSTAMakeCallConfEvent***.

The client application uses this service to set up a call on behalf of a station extension (calling party) to an on- or off-DBS endpoint (***calledDevice***).

All trunk types are supported as facilities for reaching called endpoints for outbound ***csta-MakeCall*** calls. The only call progress feedback that is reported as an event to the application via Monitor Services is Networked Reached.

For the originator to place the call, the ***callingDevice*** (digital extension) must have an available appearance for call origination and must not be in the talking (active) state on any appearances. The originator is allowed to have a call(s) on hold or alerting at the device.

The originator may go off-hook or turn the speaker on and receive dial tone first, then issue the Make Call Service request for that station. The switch will originate the call on the same callID to establish the call.

If the originator is off-hook busy, the call can not be placed and the request is denied. If the originator is unable to originate for other reasons (see ***universalFailure***), the switch denies the request.

Service Parameters:

callingDevice	[mandatory] must be a valid station extension
calledDevice	[mandatory] must be a valid on-DBS extension or off-DBS number. as if they were entered from the telephone using the key pad.

Ack Parameters:

newCall	[mandatory] a connection identifier indicates the connection between the originating device and the call. The ***newCall*** parameter contains the callID of the call and the station extension of the ***callingDevice***.

Nak Parameter:

A MakeCall request will be denied if the request fails before the call is attempted by the DBS:

universalFailure

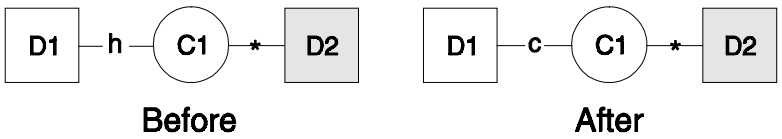
- INVALID_CALLING_DEVICE (5)
  The ***callingDevice*** is out of service or invalid.
- INVALID_CALLED_DEVICE (6)
  The ***calledDevice*** number is too long or contains an illegal digit.
- REQUEST_INCOMPATABLE_WITH_OBJECT (2)
  The ***callingDevice*** is not monitored.
- INVALID_OBJECT_STATE (22)
  The ***callingDevice*** is not in a legal state for makecall.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

- TRS - The same Toll Restriction Services apply to calls generated via a CSTAMake-Call request as exist with user dialed numbers.
- LCR - Least Cost Routing will apply to a CSTAMakeCall request with the first dialed digit "9".
- Direct Trunk Access - A trunk can be directly accessed via a CSTAMakeCall request, with the dialed digits containing "88XX".
- Forced Entry Account Codes - Non-verified account codes are supported via a CSTA-MakeCall request, using "A" for the auto key. Verified account codes are supported via a CSTAMakeCall request.
- Called Destination - if the called device is a DBS station extension, the user at the station will receive alerting tones. The calling station user will hear dialtone but no touchtones.
- Call Forwarding Immediate - No CSTA events will be delivered to an extension which is call forwarded immediately.
- Display - Most extension indicators and displays will function identically whether under user or CSTA control. However, CSTAMakeCall dialed digits will not appear on an extensions display.
- Last Number Dialed - Sends the same events as the original call.
- SMDR - Any calls made via a CSTAMakeCall request and any call answered via a CSTAAnswerCall request will produce SMDR records when appropriate.
- Switch Operation - if the digits dialed result in listening to busy tone, the busy tone will last until the user hangs up via the handset and/or speakerphone or CSTA Clear Call or CSTA Clear Connection services.
- Make Tone Call - CSTAMakeCall request with digits plus "1" if switch is set to voice call default.
- Make Voice Call - CSTA Make Call request with digits plus "1" if switch is set to tone call default.
- Make Page Call - CSTA Make Call request with "#"<page group>

Retrieve Call Service

This service restores a held connection ***heldCall*** (C1-D1) to the Connected state (active).



Function: ***cstaRetrieveCall(), CSTARetrieveCallConfEvent***
Direction: C → S
Functional Description:
The Retrieve Call Service connects an on-DBS held connection.

Service Parameters:

heldCall	[mandatory] a valid connection identifier indicates the endpoint to be connected. The deviceID in ***heldCall*** must contain the station extension of the endpoint.

Ack Parameters:

noData	None for this service.

Nak Parameter:

universalFailure

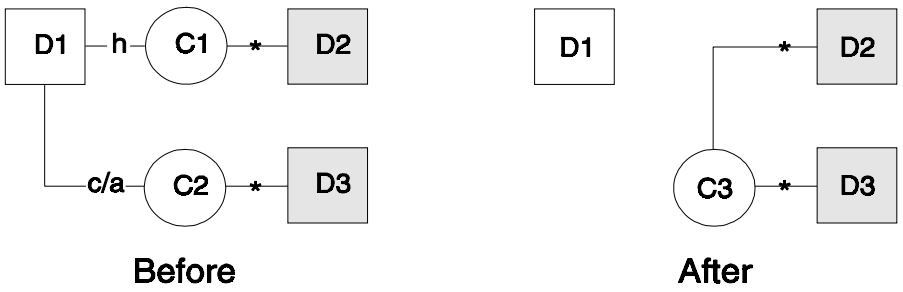
- INVALID_CSTA_CONNECTION_IDENTIFIER (13)
  The connectionID contained in the request is invalid.
- NO_ACTIVE_CALL (24)
  The specified call at the station is cleared so it cannot be retrieved.
- NO_HELD_CALL (25)
  The specified connection at the station is not in the held state (e.g., alerting state) so it cannot be retrieved.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

- If the user is listening to dial tone while a request for Retrieve Call Service is received, the dial tone will be dropped and the user reconnected to the held call.

Transfer Call Service

This service provides the transfer of a ***heldCall*** (C1-D1) with an ***activeCall*** (C2-D1) at the same device (D1). The transfer service merges two calls (C1, C2) with connections (C3-D2, C3-D3) at a single common device (D1) into one call (C3). Also, both of the connections to the common device become Null and their connectionIDs are released. When the transfer completes, the common device (D1) is released from the calls (C1, C2). A callID, ***newCall*** (C3), that specifies the resulting new call for the transferred call is provided.



Function: ***cstaTransferCall(), CSTATransferCallConfEvent***
Direction: C → S
Functional Description:

This service provides the transfer of an existing held call (***heldCall***) and another active or proceeding call (alerting, or connected) (***activeCall***) at a device. The Transfer Service merges two calls with connections at a single common device into one call. Also, both of the connections to the common device become Null and their connectionIDs are released. A connectionID that specifies the resulting new connection for the transferred call is provided.

Service Parameters:

heldCall	[mandatory] must be a valid connection identifier for the call which is on hold at the controlling device and is to be transferred to the ***activeCall***. The deviceID in ***heldCall*** must contain the station extension of the controlling device of the controlling device.
activeCall	[mandatory] must be a valid connection identifier of an active or proceeding call at the controlling device to which the ***heldCall*** is to be transferred. The deviceID in ***activeCall*** must contain the station extension of the controlling device.

Ack Parameters:

newCall	[mandatory] a connection identifier that specifies the resulting new call identifier for the transferred call.
connList	[optional - supported] specifies the devices on the resulting newCall. This includes a count of the number of devices in the new call and a list of up to four connectionIDs and up to four deviceIDs which define each connection in the call. •If a device is on-DBS, the extension is specified. •If a party is off-DBS, then its assigned trunk identifier is specified except if Caller ID digits are available.

Nak Parameter:

universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  An invalid device identifier or extension is specified in ***heldCall*** or ***activeCall***.
- INVALID_CSTA_CONNECTION_IDENTIFIER (13)
  The controlling deviceID in ***activeCall*** or ***heldCall*** has not been specified correctly.
- NO_HELD_CALL (25))
  The held call parameter is incorrect.
- NO_ACTIVE_CALL (24)
  The active call parameter is incorrect.
- MISTYPED_ARGUMENT_REJECTION (74)
  DYNAMIC_ID is specified in ***heldCall*** or ***activeCall***.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request..
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

- Trunk to Trunk Transfer - Existing rules for trunk-to-trunk transfer from a station user will remain unchanged for application monitored calls. In such cases, transfer requested via Transfer Call Service will be denied. When this feature is enabled, application monitored calls transferred from trunk to trunk will be allowed, but there will be no further event reports (except for the Network Reached, Established, Connection Cleared Event Reports sent to the application).

Chapter 3. Set Feature Service Group

Overview

These services allow a client application to set switch-controlled features on a Panasonic DBS.

The following CSTA Services are supported in the NetWare Telephony Services product:

- Set Do Not Disturb Feature Service
- Set Forwarding Feature Service
- Set Message Waiting Indicator Feature Service

Set Do Not Disturb Feature Service

Function: ***cstaSetDoNotDisturb( ), CSTASetDndConfEvent***
Direction: C → S
Functional Description:
This service turns on or off the Do Not Disturb feature for a user station.

Service Parameters:

device	[mandatory] must be a valid DBS extension.
doNotDisturb	[mandatory] specifies either on (TRUE) or off (FALSE).

Ack Parameters:

noData	None for this service.

Nak Parameter:
universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  An invalid device identifier has been specified in device.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- GENERIC_SUBSCRIBED_RESOURCE_AVAILABILITY (41)
  The user does not have the permissions to set DND.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

- COS - the user must have a COS that given them the ability to set DND.

Set Forwarding Feature Service

Function: ***cstaSetForwarding( ), CSTASetFwdConfEvent***
Direction: C → S
Functional Description:
The Set Forwarding Service sets the DBS Call Forwarding feature on or off for a user station. The DBS supports Call Forward Immediate, Call Forward Busy, and Call Forward No Answer types.

Service Parameters:

device	[mandatory] specifies the station on which the Call Forwarding feature is to be set. It must be a valid DBS extension.
forwardingType	[mandatory - partial] specifies the type of forwarding to set or clear.
forwardingOn	[mandatory] specifies either on (TRUE) or off (FALSE).
forwardingDN	[mandatory] specifies the station extension or off premise destination to which the calls are to be forwarded. It is mandatory if forwardingOn is set to on. It is ignored if the forwardingOn is set to off.

Ack Parameters:

noData	None for this service.

Nak Parameter:
universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  An invalid device identifier has been specified in ***device***.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.
- GENERIC_OPERATION (1)
  An unsupported call forward type was specified.

Notes:

- COS - the user must have a COS that given them the ability to set DND.

Set Message Waiting Indicator Feature Service

Function: ***cstaSetMsgWaitingInd( ), CSTASetMwiConfEvent***
Direction: C → S
Functional Description:
This service sets on or off the DBS message waiting indicator (MWI) for a user station.

Service Parameters:

device	[mandatory] must be a valid DBS extension that supports the MWI feature.
messages	[mandatory] specifies either on (TRUE) or off (FALSE).

Ack Parameters:

noData	None for this service.

Nak Parameter:

universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  An invalid device identifier has been specified in device.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is off-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has too many outstanding CSTA requests.
- GENERIC_UNSPECIFIED (0)
  Internal driver error
- GENERIC_OPERATION_REJECTION (71)
  The Panasonic Telephony Services driver (Panadrvr) does not have a legal message waiting indicator value. The DBS requires the driver to indicate which extension port is activating an extension's message waiting lamp. In Release 1.0, their value is supplied by the driver.

Notes:

- Extension must be installed.

Chapter 4. Set Query Service Group

Overview

These services allow a client application to query the switch for the state of device features and static attributes of a device. Panadrvr Release 1.0 supports the following Query Services:

- **Query Do Not Disturb Service**
- **Query Forwarding Service**
- **Query Message Waiting Service**
- **Query Last Number**

The following Query Services are **not** supported by Panadrvr Release 1.0.

- **Query Agent State Service**
- **Query Device Info**

Query Do Not Disturb Service

Function: *cstaQueryDoNotDisturb( ),* C*STAQueryDoNotDisturbConfEvent*
Direction: C → S
Functional Description:
This service provides the status of the do not disturb feature expressed as on or off on a device.

Service Parameters:

device	[mandatory] Must be a valid DBS station extension that supports the do not disturb feature.

Ack Parameters:

doNotDisturb	[mandatory] Status of the do not disturb feature expressed as on or off.

Nak Parameter:
universalFailure

- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.
- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  AN invalid device identifier has been specified in ***device***.

Query Forwarding Service

Function: cstaQueryForwarding (), CSTAQueryForwardingConfEvent
Direction: C → S
Functional Description:

This service provides the status and forward-to-number of the Call Forwarding feature for a device. The status is expressed as on or off. The DBS supports types of call forwarding which cannot be directly specified in CSTA terms. Call forward no answer, call forward busy and call forward immediate are expressed as their CSTA values, call forward busy/no answer is expressed as CSTA call forward no answer.

Service Parameters:

device [mandatory] must be a valid DBS station extension that supports the Call Forwarding feature.

Ack Parameters:

forward [mandatory] This is a list of forwarding parameters. The list contains a count of how many items are in the list. Since the DBS switch stores only one forwarding address, the list is of length one and the count is one. Each element in the list contains the following: *forwardingType*, *forwardingOn*, and *forwardDN*. ForwardingType will be one of the values mentioned above. ForwardingOn will indicate ″on/off″ status, and forwardDN will contain the forward-to-number.

Nak Parameter:

universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  AN invalid device identifier has been specified in device.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request..
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Query Message Waiting Service

Function: cstaQueryMsgWaitingInd(), CSTAQueryMwiConfEvent
Direction: C → S
Private Ack Parameters: applicationType
Functional Description:

The Query Message Waiting Service provides status of the message waiting indicator expressed as on or off for a device.

Service Parameters:

device [mandatory] must be a valid DBS station extension that supports the MWI feature.

Ack Parameters:

messages [mandatory] Indicates the on/off status of the message waiting indicator for this device.

Nak Parameter:

universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  AN invalid device identifier has been specified in device.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

The DBS requires the driver to specify not only which extension's message waiting indicator to activate, but also the extension number (e.g., voice mail) that is activating the indicator. In future releases, applications will inform the driver of the activator's id in private data; however in Release 1.0 the extension value must be specified when the driver is loaded or during runtime using the driver's command line switches and/or runtime menus.

Query Last Number Service

Function: cstaQueryLastNumber( ), CSTAQueryLastNumberEvent( )
Direction: C → S
Functional Description:
The Query Last Number Service provides last number dialed information for the specified device in the DBS. This number may be an internal extension or an outside trunk accessed number.

Service Parameters:

device	[mandatory] must be a valid DBS station extension.

Ack Parameters:

forward	[mandatory] This is the value held within the DBS's last number dialed buffer. Note: This may be a NULL entry.

Nak Parameter:
universalFailure

- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request.
- GENERIC_UNSPECIFIED (0)
  Internal Driver Error
- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  An invalid device identifier has been specified in ***device***.

Chapter 5. Monitor Service Group

Overview

Services in this group allow a client application to request and cancel the reporting of state-changing events. Panadrvr Release 1.0 supports the following Call-Control Services:

- Monitor Device Service
- Monitor Stop Service
- Monitor Ended Event
- Change Monitor Filter Service

The following Monitor Services are **not** supported by Panadrvr Release 1.0.

- Monitor Call Service
- Monitor Calls Via Device Service

Monitor Device Service

Function: cstaMonitorDevice(), CSTAMonitorConfEvent
Direction: C → S
Functional Description:

This service provides call event reports for all devices on all calls at a device. Event reports are provided for calls that occurred previous to the monitor request and arrive at the device after the monitor request is acknowledged. No further events of a call are reported if that call is dropped, forwarded, or transferred and the device no longer is participating in the call.

There are no subsequent event reports for a call after a Call Cleared or a Connection Cleared or a Diverted Event Report has been received for this service. Reporting of the subsequent call event reports after a Transferred Event Report is dependent on whether the call is merged-in or merged-out from the monitored device.

Service Parameters:

deviceID	[mandatory] must be a valid digital DBS extension. Analog devices are not supported in release 1.0.
monitorFilter	[optional - not supported] specifies the filters to be used with deviceID. Call Filter/Event Reports and Feature Filter/Event Reports are supported for station device.

Ack Parameters:

monitorCrossRefID	[mandatory] contains the handle chosen by the DBS Driver. This handle is a unique value within an ***acsOpenStream*** session for the duration of the monitor and is used by the application to correlate subsequent event reports to the monitor request that initiated them. It is also allows the correlation of the Monitor Stop to the original Monitor Service request.
monitorFilter	[optional - not supported] specifies the event reports that are to be filtered out on the object being monitored by the application. This may not be the ***monitorFilter*** specified in the

service request, because filters for events that are not supported by the DBS are always turned on in ***monitorFilter***.

Nak Parameter:

universalFailure

- INVALID_CSTA_DEVICE_IDENTIFIER (12)
  An invalid device identifier or extension is specified in deviceID.
- OVERALL_MONITOR_LIMIT_EXCEEDED (37)
  The request cannot be executed because the system limit would be exceeded for the maximum number of monitors or the individual device's overall monitor limit would be exceeded.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Notes:

Multiple Requests - Multiple applications can have multiple ***cstaMonitorDevice*** requests on one object. An application can have more than one ***cstaMonitorDevice*** request on multiple objects.

Monitor Ended Event Report

Function: CSTAMonitorEndedEvent
Direction: C ← S
Functional Description:

The DBS uses the Monitor Ended Event Report to cancel a subscription to a previously requested when a monitor object is removed or changed to become an invalid object. Once a Monitor Ended Event Report is generated, event reports cease to be sent to the client application by the DBS and the Cross Reference Association that was established by the original service request is terminated.

Service Parameters:

monitorCrossRefID	[mandatory] must be a valid Cross Reference ID of this ***acsOpenStream*** session.
cause	[optional - not supported] specifies the reason for this event.

Monitor Stop Service

Function: cstaMonitorStop(), CSTAMonitorStopConfEvent
Direction: C → S
Functional Description:

An application uses the Monitor Stop Service to cancel a subscription to a previously requested cstaMonitorDevice when it no longer has an interest in continuing a monitor. Once a Monitor Stop request has been acknowledged, event reports cease to be sent to the

client application by the DBS and the Cross Reference Association that was established by the original service request is terminated.

Service Parameters:

monitorCrossRefID	[mandatory] must be a valid Cross Reference ID that was returned in a previous CSTAMonitorConfEvent of this acsOpenStream session.

Ack Parameters:

noData	None for this service.

Nak Parameter:

universalFailure

- INVALID_CROSS_REF_ID (17)
  The service request specified a Cross Reference ID that is not in use at this time.
- RESOURCE_OUT_OF_SERVICE (34)
  The DBS is not on-line.
- OUTSTANDING_REQUEST_LIMIT_EXCEEDED (44)
  The application has a pending CSTA request.
- GENERIC_UNSPECIFIED (0)
  Internal driver error.

Chapter 6. Event Report Service Group

Overview

Services in this group provide a client application with reports of state-changing events to a call, a connection, or a device. Panadrvr Release 1.0 supports the following Event Report Services:

- Call Cleared Event
- Conferenced Event
- Connection Cleared Event
- Delivered Event
- Diverted Event
- Established Event
- Failed Event
- Held Event
- Network Reached Event
- Retrieved Event
- Service Initiated Event
- Transferred Event

The following Event Report Services are **not** supported by Panadrvr Release 1.0.

- Queued Event
- Logged On Event
- Logged Off Event

Definitions

Following are the definitions of the enumerated types ***CSTAEventCause*** and ***LocalConnectionState***. These data structures are used extensively by the Event Report Service Group members described in this chapter.

```
typedef enum CSTAEventCause_t {
   EC_NONE = -1,              // no cause value is specified
   EC_ACTIVE_MONITOR = 1,
   EC_ALTERNATE = 2,
   EC_BUSY = 3,
   EC_CALL_BACK = 4,
   EC_CALL_CANCELLED = 5,
   EC_CALL_FORWARD_ALWAYS = 6,
   EC_CALL_FORWARD_BUSY = 7,
   EC_CALL_FORWARD_NO_ANSWER = 8,
   EC_CALL_FORWARD = 9,
   EC_CALL_NOT_ANSWERED = 10,
```

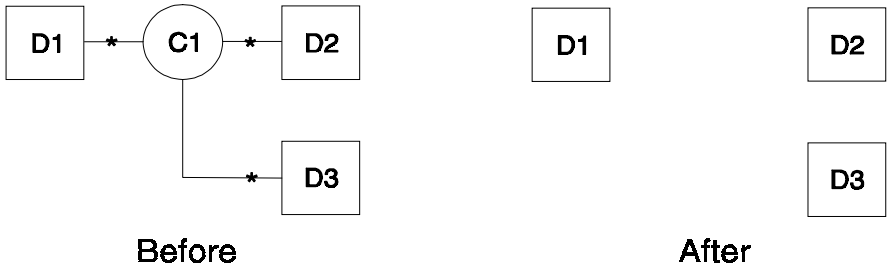
```
    EC_CALL_PICKUP = 11,
    EC_CAMP_ON = 12,
    EC_DEST_NOT_OBTAINABLE = 13,
    EC_DO_NOT_DISTURB = 14,
    EC_INCOMPATIBLE_DESTINATION = 15,
    EC_INVALID_ACCOUNT_CODE = 16,
    EC_KEY_CONFERENCE = 17,
    EC_LOCKOUT = 18,
    EC_MAINTENANCE = 19,
    EC_NETWORK_CONGESTION = 20,
    EC_NETWORK_NOT_OBTAINABLE = 21,
    EC_NEW_CALL = 22,
    EC_NO_AVAILABLE_AGENTS = 23,
    EC_OVERRIDE = 24,
    EC_PARK = 25,
    EC_OVERFLOW = 26,
    EC_RECALL = 27,
    EC_REDIRECTED = 28,
    EC_REORDER_TONE = 29,
    EC_RESOURCES_NOT_AVAILABLE = 30,
    EC_SILENT_MONITOR = 31,
    EC_TRANSFER = 32,
    EC_TRUNKS_BUSY = 33,
    EC_VOICE_UNIT_INITIATOR = 34
} CSTAEventCause_t;

typedef enum LocalConnectionState_t {
    CS_UNKNOWN = -2,
    CS_NONE = -1,
    CS_NULL = 0,
    CS_INITIATE = 1,
    CS_ALERTING = 2,
    CS_CONNECT = 3,
    CS_HOLD = 4,
    CS_QUEUED = 5,
    CS_FAIL = 6
} LocalConnectionState_t;
```

Call Cleared Event

Function: *CSTACallClearedEvent*
Direction: C ← S
Functional Description:

The Call Cleared Event Report indicates that a call is ended. Normally this occurs when the last remaining monitored device disconnects from the call.



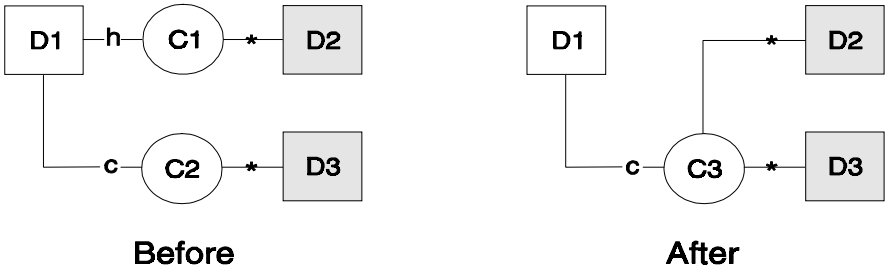
Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
clearedCall	[mandatory] specifies the callID of the call which has been cleared.
localConnectionInfo	[optional - supported] always specifies a null state (CS_NULL).
cause	[optional - supported] specifiesthe cause of the call termination. This cause is always set to EC_NONE.

Conferenced Event

Function: ***CSTAConferencedEvent***
Direction: C ← S
Functional Description:

The Conference Event Report indicates that two calls are conferenced (merged) into one, and no parties are removed from the resulting call in the process. The event may include up to four parties on the resulting call.



The Conferenced Event Report is generated for the following circumstances:

- When DBS extension completes a conference by pressing the "conference" button.
- When an application processor successfully completes a ***cstaConferenceCall*** request.
- When busy override is successfully activated.

Service Parameters:

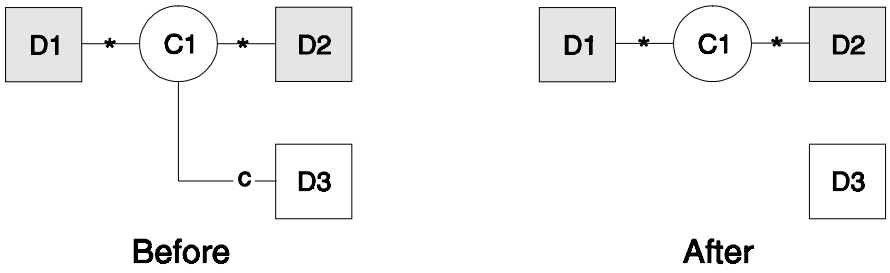
monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.

primaryOldCall	[mandatory] specifies the callID of the call that was conferenced. This is usually the held call before the conference. This call ended as a result of the conference.
secondaryOldCall	[mandatory] specifies the callID of the call that was conferenced. This is usually the active call before the conference. This call ended as a result of the conference.
confController	[mandatory] specifies the device which is controlling the conference. This is the device which setup the conference.
addedParty	[mandatory] specifies the new conferenced in device. • If the device is a DBS extension, the extension is specified. • If the party is a DBS trunk, the trunk number is given.
conferenceConnections	[optional - supported] specifies a count of the number of devices and a list of up to four connectionIDs and up to four deviceIDs which resulted from the conference. • If a device is a DBS extension, the extension is specified. • If a device is a DBS trunk, the trunk identifier is given.
localConnectionInfo	[optional - not supported] specifies the connection state of the monitored device for this call.
cause	[optional - not supported] specifies the reason for this event.

Connection Cleared Event

Function: ***CSTAConnectionClearedEvent***
Direction: C ← S
Functional Description:

The Connection Cleared Event Report indicates that a device in a call disconnects or is dropped.



A Connection Cleared Event Report is generated in the following cases:

- When a DBS extension drops from a call.
- When a DBS trunk drops from a call.
- When a monitored extension is involved in a call dissolved by a CSTA Clear Call request or a CSTA Connection Cleared request.

A Connection Cleared Event Report is **not** generated in the following cases:

- A party drops as a result of a transfer operation.

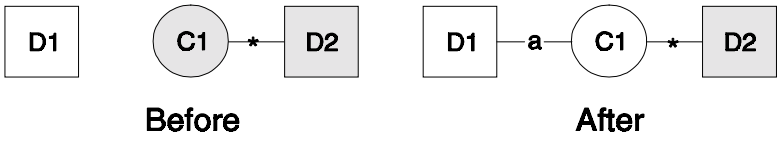
Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
droppedConnection	[mandatory] specifies the connection which has been dropped from the call.
releasingDevice	[mandatory] specifies the dropped device. • If the device is a DBS extension, then the extension is specified. • If a party is a DBS trunk, then the trunk number is specified.
localConnectionInfo	[optional - not supported] specifies the connection state of the monitored device for this call.
cause	[optional - not supported] specifies the cause of the event.

Delivered Event

Function: ***CSTADeliveredEvent***
Direction: C ← S
Functional Description:

The Delivered Event indicates that a DBS extension is alerting. Depending upon the parties monitored and the devices called, delivered events may be sent to the calling party and the called parties.



The DBS generates the Delivered Event Report when the following events occur.

- "Alerting" tone is applied to a DBS extension.
- The originator of a ***cstaMakeCall*** call is a DBS extension and ringback tone is heard or the voice path is active for voice calls.
- A monitored DBS extension makes a intercom call via the phone.

The Delivered Event Report is not sent for calls that connect to outgoing trunks.

Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
connection	[mandatory] specifies the endpoint which is alerting.
alertingDevice	[optional - supportted] specifies the device which is alerting. • If the device being alerted is a DBS extension, then the extension of the device is specified.
callingDevice	[optional - supported] specifies the calling device. The following rules apply: • For internal calls - the originator's extension. • For incoming calls from trunks, the trunk number is given unless it is a Caller ID trunk and the CID number is avail-

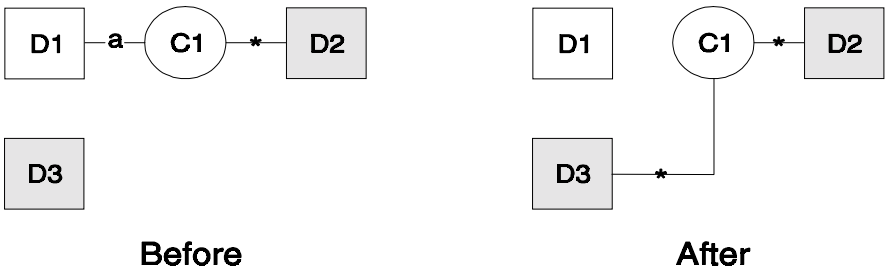
able.

calledDevice	[optional - partially supported] specifies the originally called device. In the case of diverted calls, this value is supplied.
lastRedirectionDevice	[optional - not supported] specifies the previously alerted device in case where the call was redirected or diverted to the ***alertingDevice***.
localConnectionInfo	[optional - supported] specifies the connection state of the monitored device for this call. A CS_NONE means the local connection state is unknown.
cause	[optional - supported] specifies the reason for this event.

Diverted Event

Function: ***CSTADivertedEvent***
Direction: C ← S
Functional Description

The Diverted Event Report indicates that a call has been deflected or diverted from a monitored device. It indicates that the call is no longer present at the device.



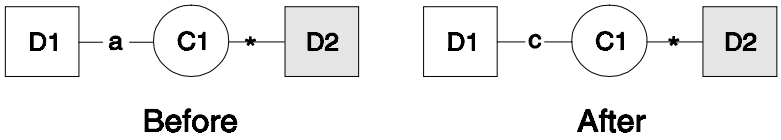
The Diverted Event Report is sent to notify the client application that event reports for a call will no longer be provided. This event report is sent to a ***cstaMonitorDevice*** monitored station when a call leaves the station, without the call having been dropped/disconnected. Examples of this are call forwarding and call pickup.

Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
connection	[mandatory] specifies the connection which was alerting.
divertingDevice	[optional - supported] specifies the device from which the call was diverted.
newDestination	[optional - supported] specifies the device to which the call was diverted.
localConnectionInfo	[optional - supported] specifies the connection state of the monitored device for this call.
cause	[optional - supported] specifies the reason for this event.

Established Event

Function: ***CSTAEstablishedEvent***
Direction: C ← S
Functional Description:
The Established Event Report indicates that the DBS detects that a device answers or connects to a call.



The Established Event Report is sent as follows:

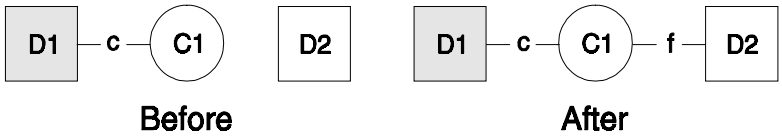
- When a call is delivered to a DBS extension and the party has answered the call (CSTA Answer Call request, picked up handset or pressed the on/off key).

Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
establishedConnection	[mandatory] specifies the endpoint which joined the call.
answeringDevice	[mandatory] specifies the device which joined the call.
callingDevice	[mandatory] specifies the calling device. The following rules apply: • For internal calls originated at a DBS station - the station's extension is specified. • For incoming calls over a trunk the trunk number is specified.
calledDevice	[optional - partially supported] specifies the originally called device.
lastRedirectionDevice	[optional - not supported] specifies the previously alerted device in case where the call was redirected or diverted to the ***answeringDevice***.
localConnectionInfo	[optional - not supported] specifies the connection state of the monitored device for this call.
cause	[optional - not supported] specifies the reason for this event.

Failed Event

Function: ***CSTAFailedEvent***
Direction: C ← S
Functional Description:
The Failed Event Report indicates that a call cannot be completed or the monitored device is the last remaining party in the call (i.e., listening to internal busy tone).



The DBS TSAPI driver has the ability to intercept failed events due to a party disconnecting from a call.

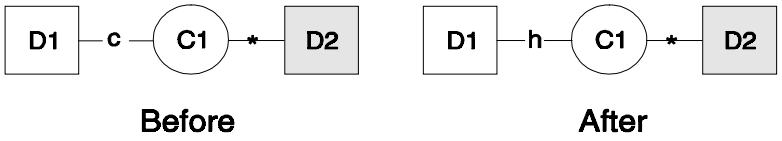
Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
failedConnection	[mandatory - supported] specifies the callID that failed.
failingDevice	[mandatory - supported] specifies the device that failed.
calledDevice	[mandatory- partially supported] specifies the called device.
localConnectionInfo	[optional - supported] specifies the connection state of the monitored device for this call.
cause	[optional - supported] specifies the reason for this event.

Held Event

Function: ***CSTAHeldEvent***
Direction: C ← S
Functional Description:

The Held Event Report indicates that a DBS extension places a call on hold. This includes the hold for conference and transfer.



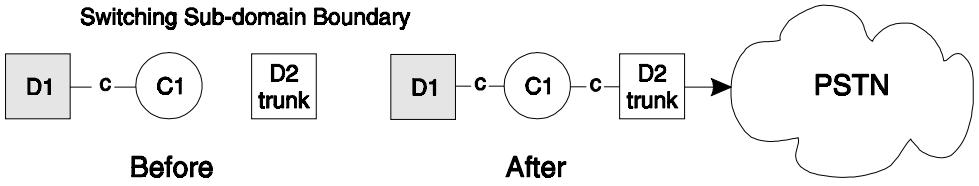
Placing a call on hold can be done either manually at the station or via a Hold Service request.

Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
heldConnection	[mandatory] specifies the endpoint where hold was activated.
holdlingDevice	[mandatory] specifies the extension that placed the call on hold.
localConnectionInfo	[optional - supported] specifies the connection state of the monitored device for this call.
cause	[optional - not supported] specifies the reason for this event.

Network Reached Event

Function: ***CSTANetworkReachedEvent***
Direction: C ← S
Functional Description:
This event indicates that a call has reached an outgoing trunk.



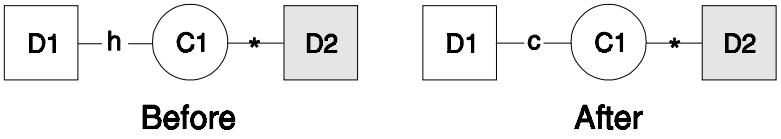
This event report implies that there will be no additional device feedback, except disconnect/drop, provided for this party in the call. A Network Reached Event Report is never sent for calls made to devices connected directly to the DBS.

Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
connection	[mandatory] specifies the endpoint for the outbound connection to another network.
trunkUsed	[mandatory] specifies the trunk identifier that was used to establish the connection.
calledDevice	[mandatory - not supported] specifies the destination device of the call. The deviceID is not supported (ID_NOT_REQUIRED).
localConnectionInfo	[optional - supported] specifies the connection state of the monitored device for this call.
cause	[optional - supported] specifies the reason for this event.

Retrieved Event

Function: ***CSTARetrievedEvent***
Direction: C ← S
Functional Description:
The Retrieved Event Report indicates that the DBS detects a previously held call has been retrieved.



It is generated when a DBS extension connects to a call that has been previously placed on hold. Retrieving a held call can be done either manually at the station or via a ***cstaRetrieveCall*** Service request from a client application.

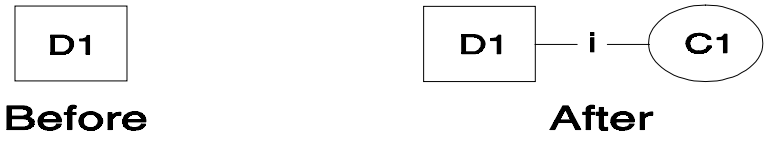
Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
retrievedConnection	[mandatory] specifies the connection for which the call has been taken off the hold state.
retrievingDevice	[mandatory] specifies the device which connected the call from the hold state. This is the extension that has been connected the call.
localConnectionInfo	[optional - supported] specifies the connection state of the monitored device for this call.
cause	[optional - not supported] specifies the reason for this event.

Service Initiated Event

Function: ***CSTAServiceInitiatedEvent***
Direction: C ← S
Functional Description:

The Service Initiated Event Report indicates that telecommunication service is initiated.



This event is generated as follows:

- When a DBS extension begins to receive intercom dial tone.
- When a station is forced off-hook because a ***cstaMakeCall*** is requested on that station.
- When a DBS extension receives intercom dial tone after placing a call on hold.
- After transferring a call via the "prog" button on the phone.
- After parking a trunk call.
- Initiating a call via a FF-key or a softkey on the phone.

Service Parameters:

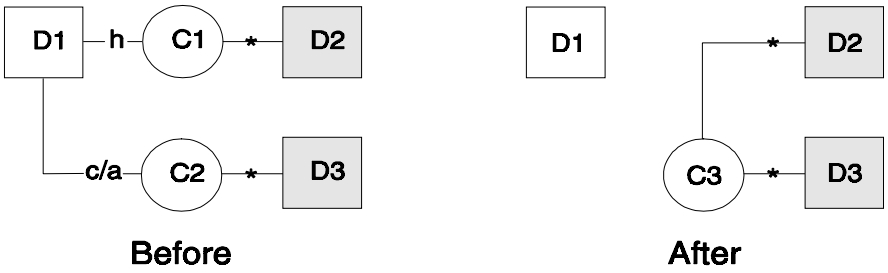
monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
initiatedConnection	[mandatory] specifies the connection for which the service (dial tone) has been initiated.
localConnectionInfo	[optional supported] specifies the connection state of the monitored device for this call.
cause	[optional - not supported] specifies the reason for this event.

Transferred Event

Function: ***CSTATransferredEvent***
Direction: C ← S

Functional Description:

The Transferred Call Event Report indicates that an existing call was transferred to another device and the device requesting the transfer has been dropped from the call. The ***transferringDevice*** will not appear in any future events for the call.



The Transferred Event Report is generated for the following circumstances:

- When a DBS extension completes a transfer by pressing the "transfer" button on the phone.
- When a DBS extension places a call on hold, initiates a new call and then hangs up. (On-Hook Transfer when enabled.)
- When an application successfully completes a ***cstaTransferCall*** request.

Service Parameters:

monitorCrossRefID	[mandatory] contains the handle to the monitor request for which this event is reported.
primaryOldCall	[mandatory] specifies the callID of the call that was transferred. This is usually the held call before the transfer. This call ended as a result of the transfer.
secondaryOldCall	[mandatory] specifies the callID of the call that was transferred. This is usually the active call before the transfer. This call is ended as a result of the transfer.
transferringDevice	[mandatory] specifies the device which is controlling the transfer. This is the device which did the transfer.
transferredDevice	[mandatory] specifies the new transferred-to device. • If the device is an on-PBX station, the extension is specified. • If the party is an off DBS device, then the trunk number is given.
transferredConnections	[optional - supported] specifies a count of the number of devices and a list of two connectionIDs and two deviceIDs which resulted from the transfer. • If a device is a DBS , the extension is specified. • The static deviceID of a queued endpoints is set to the split extension of the queue. • If a party is a trunk, then the caller id information or trunk identifier is given..
localConnectionInfo	[optional - partially supported] specifies the connection state of the monitored device for this call.
cause	[optional - not supported] specifies the reason for this event.

Chapter 7. Driver Application Interface Events

Call Cleared Event Report

Event Report	Option	Data Type	Data Type ...	Value
clearedCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	controlling extension
			ConnectionID_device_t deviceIDType	STATIC_ID
localConnectionInfo	optional - supported	LocalConnectionState_t	enum	CS_NULL
cause	optional - supported	CSTAEventCause_t	enum	cause_value or EC_NONE for normal call clearing

Conferenced Event Report

Event Report	Option	Data Type	Data Type ...	Value
primaryOldCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	controlling extension
			ConnectionID_device_t deviceIDType	STATIC_ID
secondaryOldCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	controlling extension
			ConnectionID_device_t deviceIDType	STATIC_ID
confController	mandatory	SubjectDeviceID_t	DeviceID_t device ID	controlling extension
			ConnectionID_device_t deviceIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
addedParty	mandatory	SubjectDeviceID_t	DeviceID_t device ID	extension or trunk from active call
			ConnectionID_device_t deviceIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED

Event Report	Option	Data Type	Data Type ...	Value
conferencedConnections	optional	ConnectionList_t	int count	count - up to 4
			Connection_t *connection	array of Connection_t structures
		Connection_t	ConnectionID_t party	connection[i]
			SubjectDeviceID_t staticDevice	
		ConnectionID_t	long callid	call_id
			DeviceID_t devID	extension or trunk
			ConnectionID_Device_t devIDType	STATIC_ID (extension or DYNAMIC_ID (trunk)
		SubjectDeviceID_t	DeviceID_t deviceID	extension or trunk
			Connection ID_Device_t devIDType	DEVICE_IDENTIFIER or TRUNK_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
localConnectionInfo	optional	LocalConnectionState_t	enum	CS_CONNECT
cause	optional	CSTAEventCause_t	enum	EC_KEY_CONFERENCE

Connection Cleared Event Report

Event Report	Option	Data Type	Data Type ...	Value
droppedConnection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	initiator of call - extension or turnk
			ConnectionID_device_t deviceIDType	STATIC_ID - extension DYNAMIC_ID - trunk
releasingDevice	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	extension or trunk that dropped off the call
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER - extension TRUNK IDENTIFIER - trunk
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
localConnectionInfo	optional	LocalConnectionState_t	enum	CS_CONNECT - or CS_FAIL if not the releasing Device CS_NULL - if the releasing Device
cause	optional	CSTAEventCause_t	enum	EC_NONE

Delivered Event Report

Event Report	Option	Data Type	Data Type ...	Value
connection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	initiator of call - extension or trunk
			ConnectionID_device_t deviceIDType	STATIC_ID - extension DYNAMIC_ID - trunk
alertingDevice	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	extension where the call is alerting or ringing
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
callingDevice	mandatory	CallingDeviceID_t	DeviceID_t deviceID	extension or trunk that initiated the call
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER - extension TRUNK_IDENTIFIER - trunk
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
calledDevice	mandatory	CalledDeviceID_t	DeviceID_t deviceID	extension or trunk that was initially called
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER - extension TRUNK_IDENTIFIER - trunk
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
lastRedirectionDevice	optional - not supported	CalledDeviceID_t	DeviceID_t deviceID	NULL
			ConnectionID_device_t devIDtype	EXPLICIT_PUBLIC_UNKNOWN
			DeviceIDStatus_t deviceIDStatus	ID_NOT_REQUIRED
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_CONNECT for calling device CS_ALERTING for the alerting device
cause	optional	CSTAEventCause_t	enum	EC_NEWCALL

Established Event Report

Event Report	Option	Data Type	Data Type ...	Value
establishedconnection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	initiator of call - extension or trunk
			ConnectionID_device_t deviceIDType	STATIC_ID - extension DYNAMIC_ID - trunk
answeringDevice	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	extension that answered the call
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
callingDevice	mandatory	CallingDeviceID_t	DeviceID_t deviceID	extension or trunk that initiated the call
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER - extension TRUNK_IDENTIFIER - trunk
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
calledDevice	mandatory	CalledDeviceID_t	DeviceID_t deviceID	extension that was initially called
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED extension where the call is alerting or ringing
lastRedirectionDevice	optional - not supported	CalledDeviceID_t	DeviceID_t deviceID	NULL
			ConnectionID_device_t devIDtype	EXPLICIT_PUBLIC_UNKNOWN
			DeviceIDStatus_t deviceIDStatus	ID_NOT_REQUIRED
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_CONNECT
cause	optional	CSTAEventCause_t	enum	EC_NONE

Failed Event Report

Event Report	Option	Data Type	Data Type ...	Value
failedConnection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	initiating extension
			ConnectionID_device_t deviceIDType	STATIC_ID - extension
failingDevice	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	initiating extension
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
calledDevice	mandatory	CalledDeviceID_t	DeviceID_t deviceID	extension or trunk that was originally called
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER - extension TRUNK_IDENTIFIER - trunk
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_FAIL
cause	optional	CSTAEventCause_t	enum	cause_value

Held Event Report

Event Report	Option	Data Type	Data Type ...	Value
heldconnection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	initiator of call - extension or trunk
			ConnectionID_device_t deviceIDType	STATIC_ID - extension DYNAMIC_ID - trunk
holdingDevice	mandatory	CalledDeviceID_t	DeviceID_t deviceID	extension where the call was put on hold
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_HELD for holding device CS_CONNECT for device being held
cause	optional	CSTAEventCause_t	enum	EC_NONE

Network Reached Event

Event Report	Option	Data Type	Data Type ...	Value
connection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	trunk_id
			ConnectionID_device_t deviceIDType	DYNAMIC_ID
trunkUsed	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	trunk_id
			ConnectionID_device_t devIDType	TRUNK_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
calledDevice	mandatory - not supported	CalledDeviceID_t	DeviceID_t deviceID	NULL
			DonnectionID_device_t devIDType	EXPLICIT_PUBLIC_UNKNOWN
			DeviceIDStatus_t deviceIDStatus	ID_NOT_REQUIRED
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_CONNECT
cause	optional	CSTAEventCause_t	enum	EC_NEWCALL

Retrieved Event Report

Event Report	Option	Data Type	Data Type ...	Value
retrievedconnection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	extension retrieving the call
			ConnectionID_device_t deviceIDType	STATIC_ID
retrievingDevice	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	extension retrieving the call
			DonnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_CONNECT
cause	optional	CSTAEventCause_t	enum	EC_NONE

Service Initiated Report

Event Report	Option	Data Type	Data Type ...	Value
initiatedConnection	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	extension initiating the call
			ConnectionID_device_t deviceIDType	STATIC_ID
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_INITIATE
cause	optional	CSTAEventCause_t	enum	EC_NEW_CALL

Transferred Event Report

Event Report	Option	Data Type	Data Type ...	Value
primaryOldCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extension
			ConnectionID_device_t deviceIDType	STATIC_ID
secondaryOldCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extensio
			ConnectionID_device_t deviceIDType	STATIC_ID
transferringDevice	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	controlling extension
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
transferredDevice	mandatory	SubjectDeviceID_t	DeviceID_t deviceID	extension or trunk from active call
			ConnectionID_device_t devIDType	DEVICE_IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED

Event Report	Option	Data Type	Data Type ...	Value
transferredConnections	optional	ConnectionList_t	int count	count - 2
			Connection_t *connection	array of connection_t
		Connection_t	ConnectionID_t party	connection[i]
			SubjectDeviceID_t staticDevice	
		ConnectionID_t	long callid	call_id
			DeviceID_t devID	extension or trunk
			ConnectionID_Device_t devIDType	STATIC_ID (extension) or DYNAMIC_ID (trunk)
		SubjectDeviceID_t	DeviceID_t deviceID	extension or trunk
			ConnectionID_Device_t devIDType	DEVICE_IDENTIFIER or TRUNK _IDENTIFIER
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED
LocalConnectionInfo	optional	LocalConnectionState_t	enum	CS_CONNECT or CS_ALERTING
cause	optional	CSTAEventCause_t	enum	EC_TRANSFER

Monitor Ended Event Report

Event Report	Option	Data Type	Data Type ...	Value
monitorCrossRefID	mandatory	CSTAMonitorCrossRefID	long	cross_ref_id
cause	optional	CSTAEventCause_t		EC_NONE

Chapter 8. Driver Application Interface Services

Universal Failure Confirmation

Parameter	Option	Data Type	Data Type ...	Value
eventHeader	mandatory	ACSEventHeader_t	ACSHandle_t acsHandle	
			EventClass_t eventClass	
			EventType_t eventType	
		ACSHandle_t	unsigned short	handle_id
		EventClass_t	unsigned short	CSTACONFIRMATION
		EventType_t	unsigned short	CSTA_UNIVERSAL_FAILURE_CONF
invokeID	mandatory	InvokeID_t	unsigned long	invoke_id
universalFailure	mandatory	CSTAUniversalFailureConfEvent_t	CSTAUniversalFailure_t error	
		CSTAUniversalFailure_t	enum	error_code
*privateData	optional not supported			

Answer Call Service

Service Request	Option	Data Type	Data Type ...	Value
*alertingCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	answering extension
			ConnectionID_device_t devIDType	STATIC_ID

Confirmation	Option	Data Type	Data Type ...		Value
answerCallConf	mandatory	CSTAAnswerCallConfEvent_t	Nulltype null	char	NULL

Clear Call Service

Service Request	Option	Data Type	Data Type ...	Value
*call	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	not used
			ConnectionID_device_t devIDType	not used

Confirmation	Option	Data Type	Data Type ...		Value
clearCall	mandatory	CSTAClearCallConfEvent_t	Nulltype null	char	NULL

Clear Connection Service

Service Request	Option	Data Type	Data Type ...	Value
*call	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	extension dropping from the call
			ConnectionID_device_t devIDType	STATIC_ID

Confirmation	Option	Data Type	Data Type ...		Value
clearConnection	mandatory	CSTAClearConnectionConfEvent_t	Nulltype null	char	NULL

Conference Call Service

Service Request	Option	Data Type	Data Type ...	Value
*heldCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extension
			ConnectionID_device_t devIDType	STATIC_ID
*activeCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extension
			ConnectionID_device_t devIDType	STATIC_ID

Confirmation	Option	Data Type	Data Type ...	Value
conferenceCall	mandatory	CSTAConferenceCallConfEvent_t	ConnectionID_t newcall	
			ConnectionList_t connList	
newCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	controlling extension
			ConnectionID_Device_t devIDType	STATIC_ID
connList	optional	ConnectionList_t	int count	count - up to 4
			Connection_t *connection	array of Connection_t structures
		Connection_t	ConnectionID_t party	connection[i]
			DeviceID_t staticDevice	
		ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	extension or trunk_id on the call
			ConnectionID_Device_t devIDType	STATIC_ID - extension DYNAMIC_ID - trunk
		SubjectDeviceID_t	DeviceID_t deviceID	extension or trunk_id on the call
			ConnectionID_Device_t devIDType	DEVICE_IDENTIFIER - extension TRUNK_IDENTIFIER - trunk
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED

Hold Call Service

Service Request	Option	Data Type	Data Type ...	Value
*activeCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extension
			ConnectionID_device_t devIDType	STATIC_ID
reservation	optional not supported	Boolean		ON

Confirmation	Option	Data Type	Data Type ...		Value
heldCall	mandatory	CSTAHoldConnectionConfEvent_t	Nulltype null	char	NULL

Make Call Service

Service Request	Option	Data Type	Data Type ...	Value
*callingDevice	mandatory	DeviceID_t	char[64]	source ext
*calledDevice	mandatory	DeviceID_t	char[64]	dest.digits

Confirmation	Option	Data Type	Data Type ...	Value
makeCall	mandatory	CSTAMakeCallConfEvent_t	ConnectionID_t newCall	
newCall	mandatory	ConnectionID_t	long callID DeviceID_t deviceID ConnectionID_Device_t devIDType	call_id device_num STATIC_ID

Retrieve Call Service

Service Request	Option	Data Type	Data Type ...	Value
*heldCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extension
			ConnectionID_Device_t devIDType	STATIC_ID

Confirmation	Option	Data Type	Data Type ...		Value
retrieveCall	mandatory	CSTARetrieveCallConfEvent_t	Nulltype null	char	NULL

Transfer Call Service

Service Request	Option	Data Type	Data Type ...	Value
*heldCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extension
			ConnectionID_Device_t devIDType	STATIC_ID
*activeCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	controlling extension
			ConnectionID_Device_t devIDType	STATIC_ID

Confirmation	Option	Data Type	Data Type ...	Value
transferCall	mandatory	CSTATransferCallConfEvent_t	ConnectionID_t newcall	
			ConnectionList_t connList	
newCall	mandatory	ConnectionID_t	long callID	call_id
			DeviceID_t device ID	controlling extension
			ConnectionID_Device_t devIDType	STATIC_ID

Confirmation	Option	Data Type	Data Type ...	Value
connList	optional	ConnectionList_t	int count	count - 2
			Connection_t *connection	array of Connection_t structures
		Connection_t	ConnectionID_t party	connection[i]
			DeviceID_t staticDevice	
		ConnectionID_t	long callID	call_id
			DeviceID_t deviceID	extension or trunk_id on the call
			ConnectionID_Device_t devIDType	STATIC_ID - extension DYNAMIC_ID - trunk
		SubjectDeviceID_t	DeviceID_t deviceID	extension or trunk_id on the call
			ConnectionID_Device_t devIDType	DEVICE_IDENTIFIER - extension TRUNK_IDENTIFIER - trunk
			DeviceIDStatus_t deviceIDStatus	ID_PROVIDED

Set Do Not Disturb Feature Service

Service Request	Option	Data Type	Data Type ...	Value
*device	mandatory	DeviceID_t	char[64]	extension
doNotDisturb	mandatory	Boolean		TRUE or FALSE

Confirmation	Option	Data Type	Data Type ...		Value
setDnd	mandatory	CSTASetDndConfEvent_t	Nulltype null	char	NULL

Set Forwarding Feature Service

Service Request	Option	Data Type	Data Type ...	Value
*device	mandatory	DeviceID_t	char[64]	extension
forwardingType	mandatory	ForwardingType_t	enum	FWD_IMMEDIATE, FWD_BUSY or FWD_NO_ANS
forwardingOn	mandatory	Boolean		TRUE or FALSE
*forwardingDN	mandatory	DeviceID_t	char[64]	forwarding destination (ignored if FALSE)

Confirmation	Option	Data Type	Data Type ...		Value
setFwd	mandatory	CSTASetFwdConfEvent_t	Nulltype null	char	NULL

Set Message Waiting Indicator Feature Service

Service Request	Option	Data Type	Data Type ...	Value
*device	mandatory	DeviceID_t	char[64]	extension where light is to be turned on or off
messages	mandatory	Boolean		TRUE or FALSE

Confirmation	Option	Data Type	Data Type ...		Value
setMWI	mandatory	CSTASetMwiConfEvent_t	Nulltype null	char	NULL

Monitor Calls via Device and Monitor Device Service

Service Request	Option	Data Type	Data Type ...	Value
*deviceID	mandatory	DeviceID_t	char[64]	extension to monitor
monitorFilter	optional not supported	CSTAMonitorFilter_t	CSTACallFilter_t call	
			CSTAFeatureFilter_t feature	
			CSTAAgentFilter_t agent	
			CSTAMaintenanceFilter_t maintenance	
		CSTACallFilter_t	unsigned short	ignored
		CSTAFeatureFilter_t	unsigned short	ignored
		CSTAAgentFilter_t	unsigned short	ignored
		CSTAMaintenance Filter_t	unsigned short	ignored

Confirmation	Option	Data Type	Data Type ...	Value
monitorStart	mandatory	CSTAMonitorCallConfEvent_t	monitorCrossRefID	
			monitorFilter	
monitorCrossRefID	mandatory	CSTAMonitorCrossRefID	long	cross_ref_id
monitorFilter	optional supported	CSTAMonitorFilter_t	CSTACallFilter_t call	
			CSTAFeatureFilter_t feature	
			CSTAAgentFilter_t agent	
			CSTAMaintenanceFilter_t maintenance	
		CSTACallFilter_t	unsigned short	0x0060
		CSTAFeatureFilter_t	unsigned short	0x80
		CSTAAgentFilter_t	unsigned short	0xFC (all filters on)
		CSTAMaintenanceFilter_t	unsigned short	0xC0 (all filters on)

Monitor Stop Service

Service Request	Option	Data Type	Data Type ...	Value
monitorCrossRefID	mandatory	CSTAMonitorCrossRefID	long	cross_ref_id

Confirmation	Option	Data Type	Data Type ...		Value
monitorStop	mandatory	CSTAMonitorStopConfEvent_t	Nulltype null	char	NULL

Query Do Not Disturb Feature Service

Service Request	Option	Data Type	Data Type ...	Value
*device	mandatory	DeviceID_t	char{64}	extension

Confirmation	Option	Data Type	Data Type ...	Value
queryDND	mandatory	CSTAQueryDNDConfEvent_t	Boolean doNotDisturb	TRUE or FALSE

Query Forwarding Feature Service

Service Request	Option	Data Type	Data Type ...	Value
*device	mandatory	DeviceID_t	char{64}	extension

Confirmation	Option	Data Type	Data Type ...	Value
queryFwd	mandatory	CSTAQueryFwdConfEvent_t	ListForwardParameters_t forward	
forward	mandatory	ListForwardParameters_t	short count	count - 1
			ForwardingInfo_t param[7]	
		ForwardingInfo_t	ForwardingType_t forwardingType	FWD_IMMEDIATE FWD_BUSY or FWD_NO_ANS
			forwardingOn	TRUE or FALSE
			ForwardDN	forwarding destination

Query Message Waiting Indicator Feature Service

Service Request	Option	Data Type	Data Type ...	Value
*device	mandatory	DeviceID_t	char{64}	extension

Confirmation	Option	Data Type	Data Type ...	Value
queryMwi	mandatory	CSTAQueryMwiConfEvent_t	Boolean messages	TRUE or FALSE

Query Last Number Dialed Service

Service Request	Option	Data Type	Data Type ...	Value
*device	mandatory	DeviceID_t	char{64}	extension

Confirmation	Option	Data Type	Data Type ...	Value
queryLastNumber	mandatory	CSTAQueryLastNumberConfEvent_t	DeviceID_t lastNumber	digits from this extension's last number dialed buffer

Chapter 9. Callflow Diagrams

Overview

The following callflow diagrams reflect the CSTA Call Event Reports received by an application during call processing. The diagrams use Specification and Description Language (SDL) as defined in CCITT Recommendations Z.101 to Z.104 to depict the relationship of call state/stimulus/event report/local connection state for DBS TSAPI/CSTA events.

The following key should be used to interpret SDL symbols.

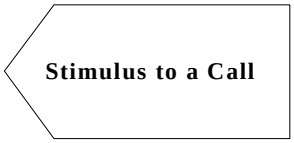
Call State

Description:

A DBS CSTA call state of the described call.

- Point of view is from a monitor request.
- An "unknown" state indicates that the state of the call is no longer of concern to the described device and event reporting has stopped (the device is no longer connected to the call).

Stimulus to a Call



Description:

A stimulus to a call includes the following:

- Button pushes, flash-hook, and other manual operations.
- DBS CSTA Service requests sent from the client application toward the call.
- DBS switch call processing events such as call queuing, call redirected to a coverage point, etc.

CSTA Event Reports

Description:

A single DBS CSTA Event Report is sent to a ***cstaMonitorDevice*** monitor request only.

- Event is sent from the switch to the application.
- Point of view is from a monitor request.
- "No Event" indicates that there is no event report for the situation.

Local Connection State

Description:

Reflects the CSTA Local Connection State of the described object after the CSTA Event Report. Only one connection's Local Connection State is reflected in the following diagrams (this state is not related to the local connection state parameter in a CSTA Event Report). Note the following conventions:

- **Before a call is established**

  During inbound calls, the Local Connection State reflects the Local Connection State of the device that *actually receives the call*. The state is therefore Null until the device is alerted. During call setup, if the call is passing through an ACD device, the Local Connection State does not represent the ACD device's Local Connection State.

  During outbound calls, the Local Connection State reflects the Local Connection State of the *calling device*.

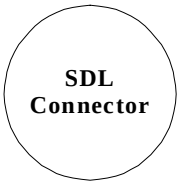
- **After a call is establis**hed

  "This", "other", or "last" is used to indicate the party that is being described.

- **No Change**

  "No Change" is used when the event report does not indicate a Local Connection State change of the monitored device. It may, however, indicate a state change of another device on the call. A state change of one device on a call generates event reports for every device on the call that has a monitor request.

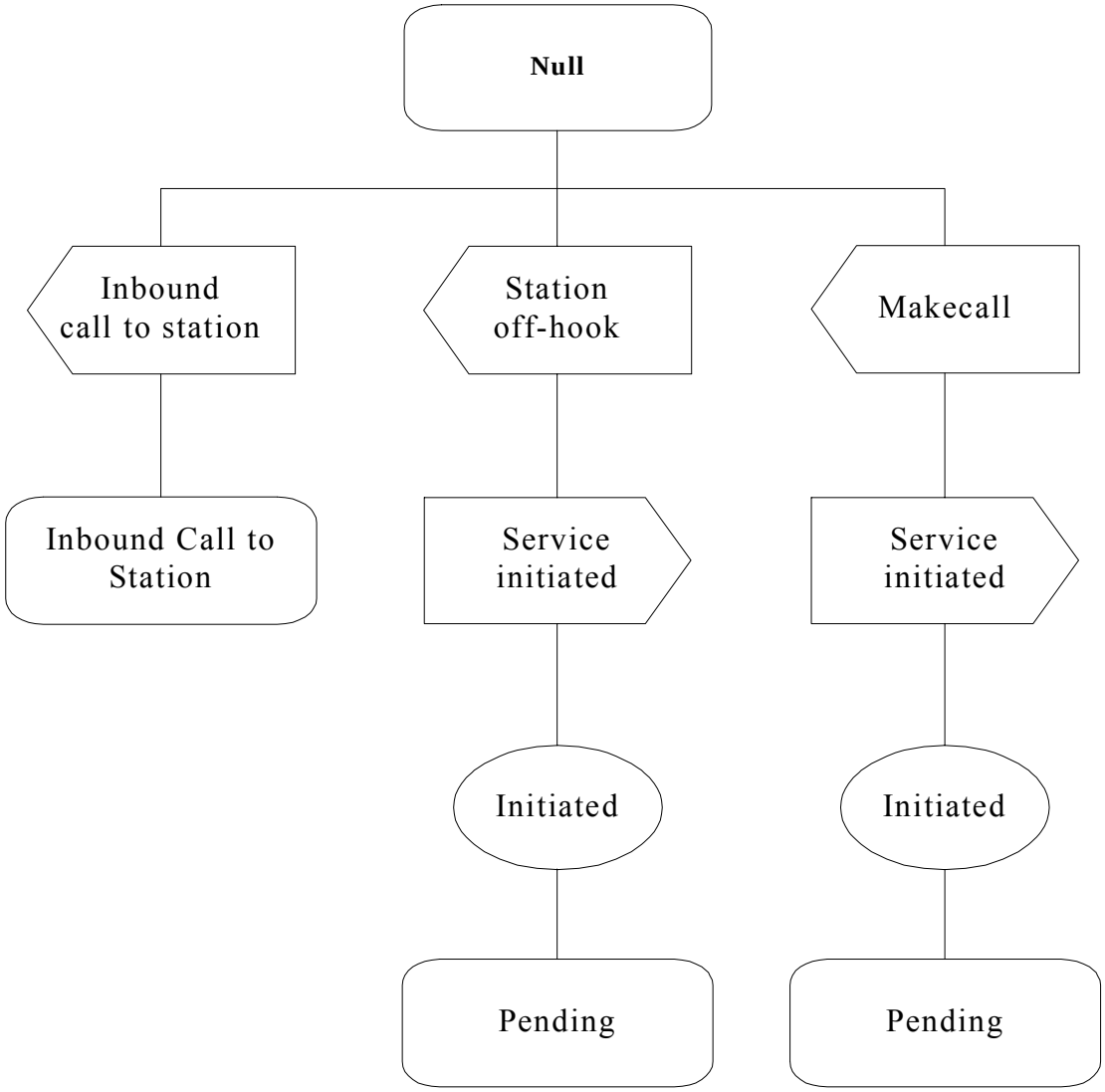
SDL Connector



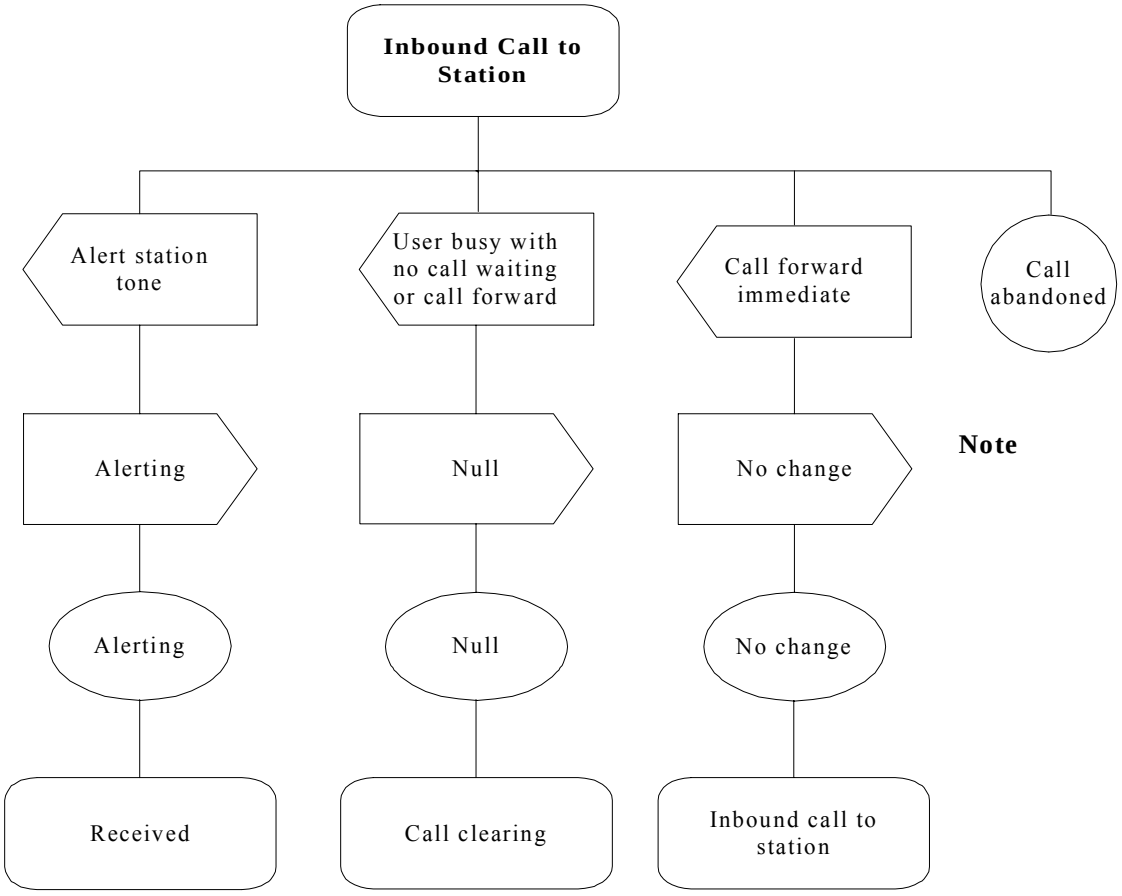
Description:

Connectors are used to reflect an extension of a state diagram. Descriptions that follow the connector are part of the state diagram showing the state to which it is connected.

Null

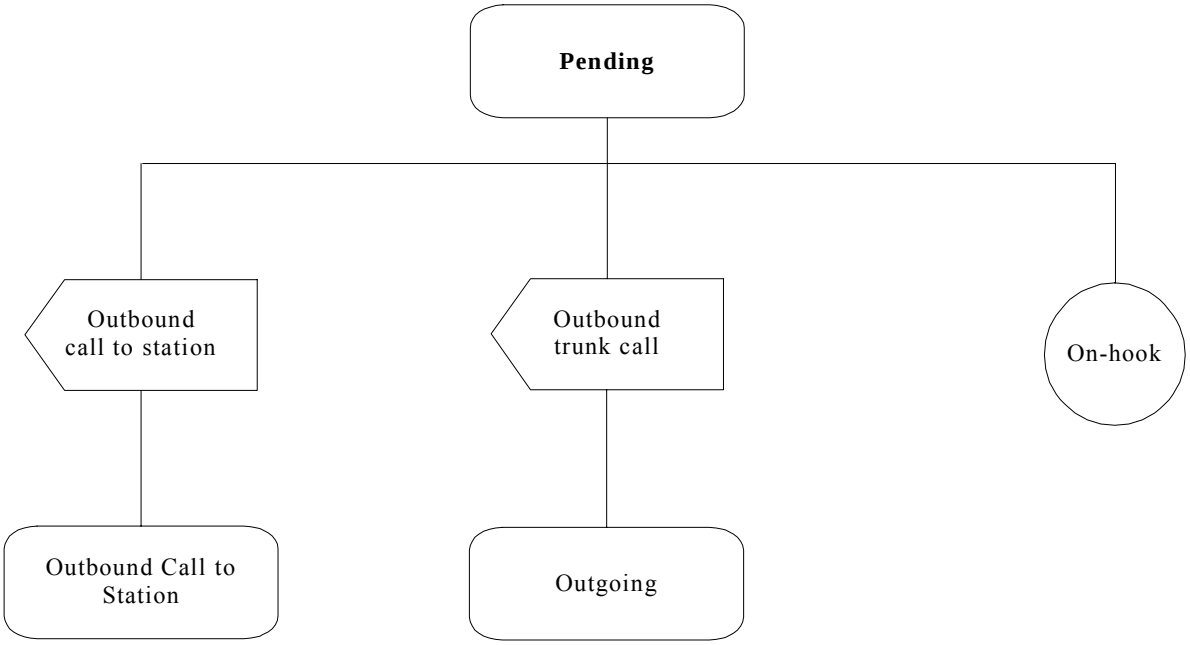

Null
Inbound call to station
Station off-hook
Makecall
Inbound Call to Station
Service initiated
Service initiated
Initiated
Initiated
Pending
Pending

Inbound Call to Station

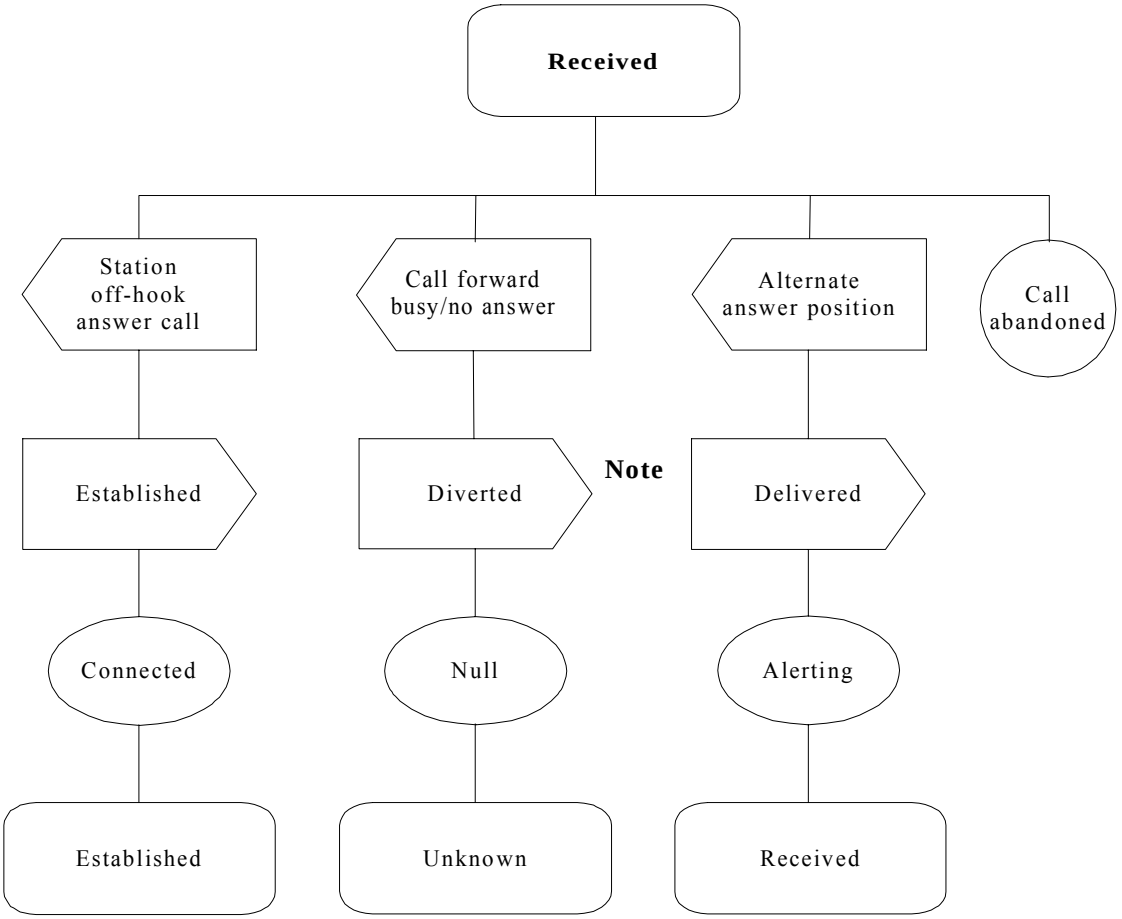


Note: Call forward immediate will not generate any event reports.

Pending

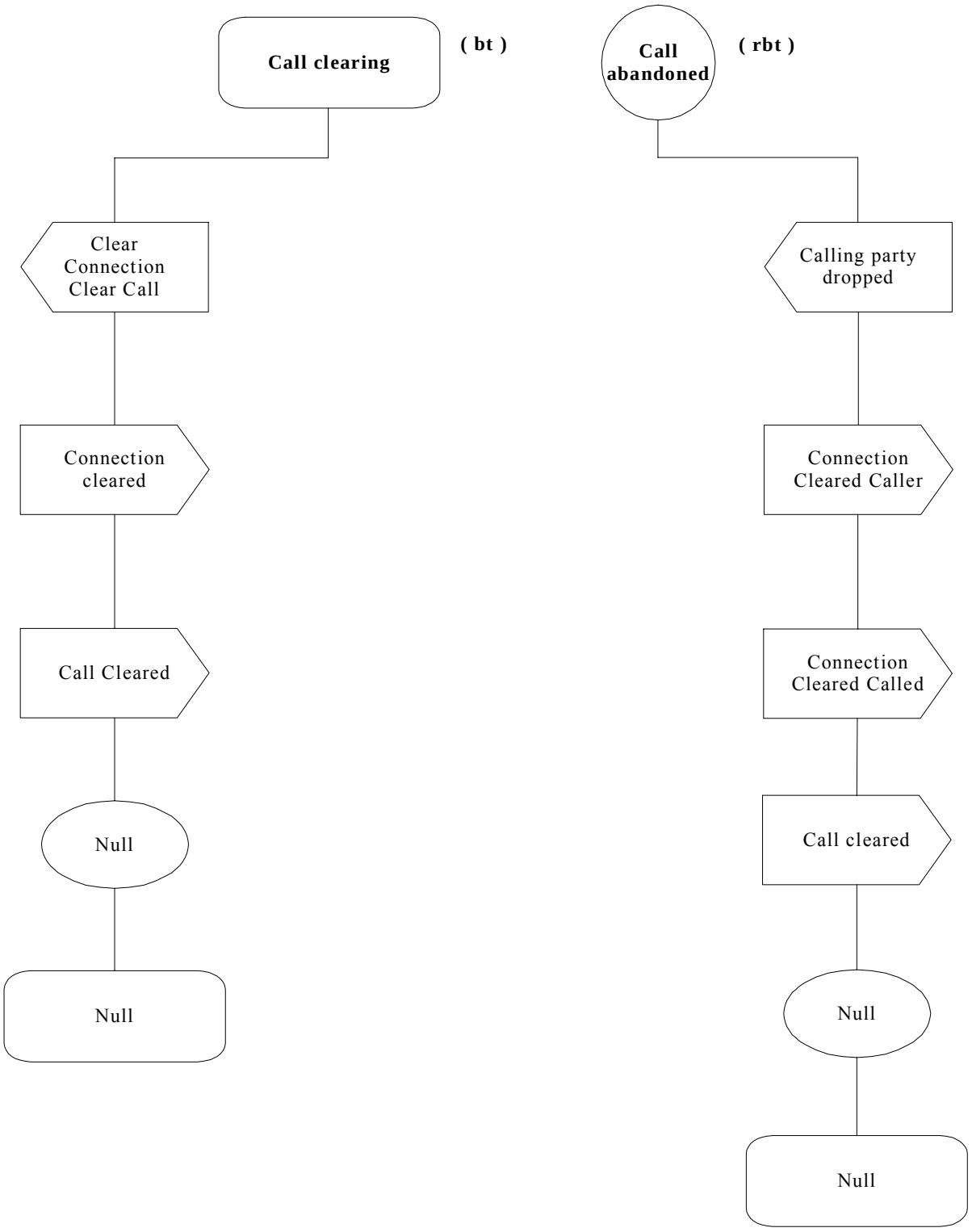

Pending
Outbound call to station
Outbound trunk call
On-hook
Outbound Call to Station
Outgoing

Received

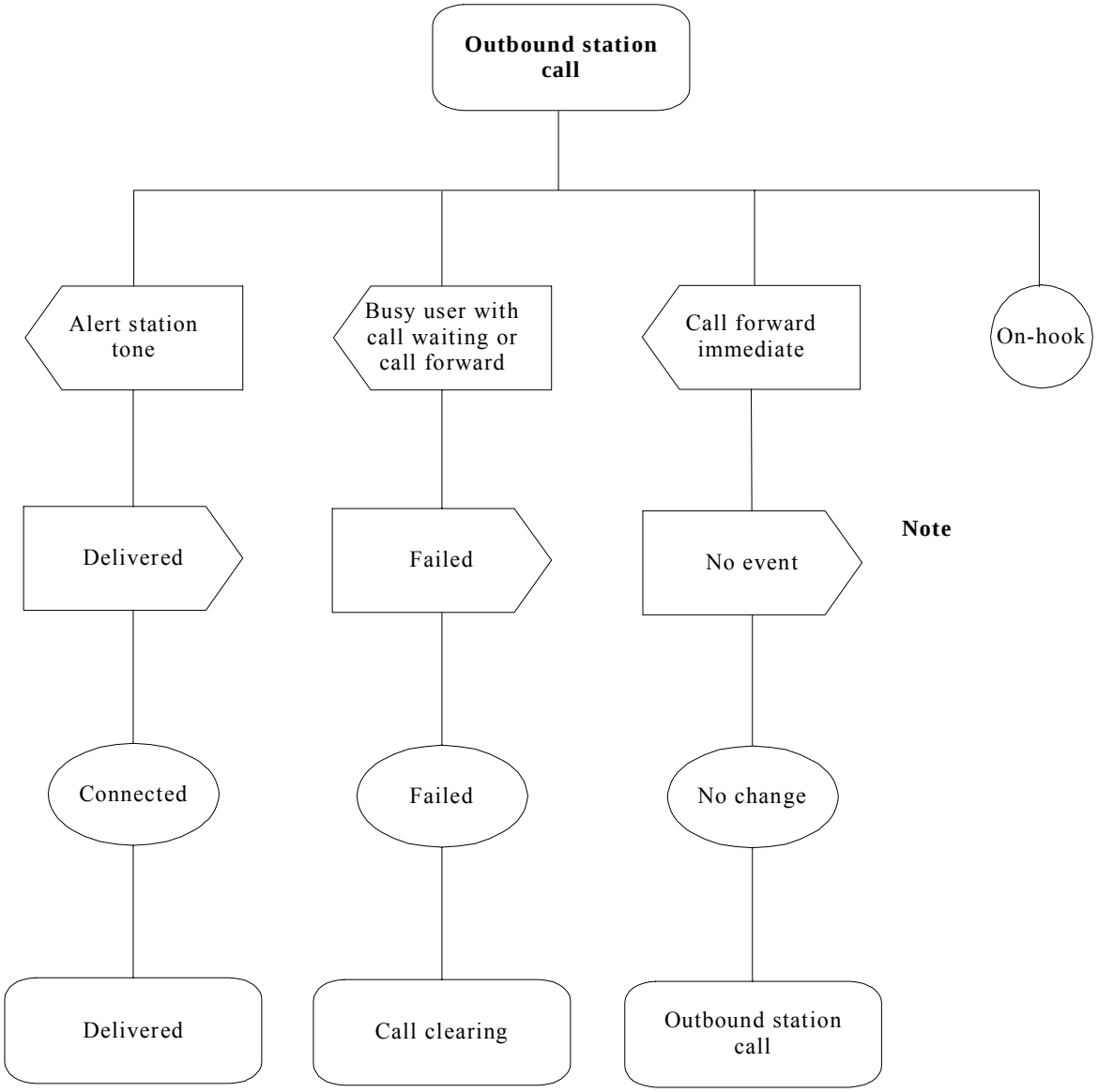


Note: The Diverted Event is sent to the station the call is being diverted away from. No more events for this call are sent to that station.

Call Clear

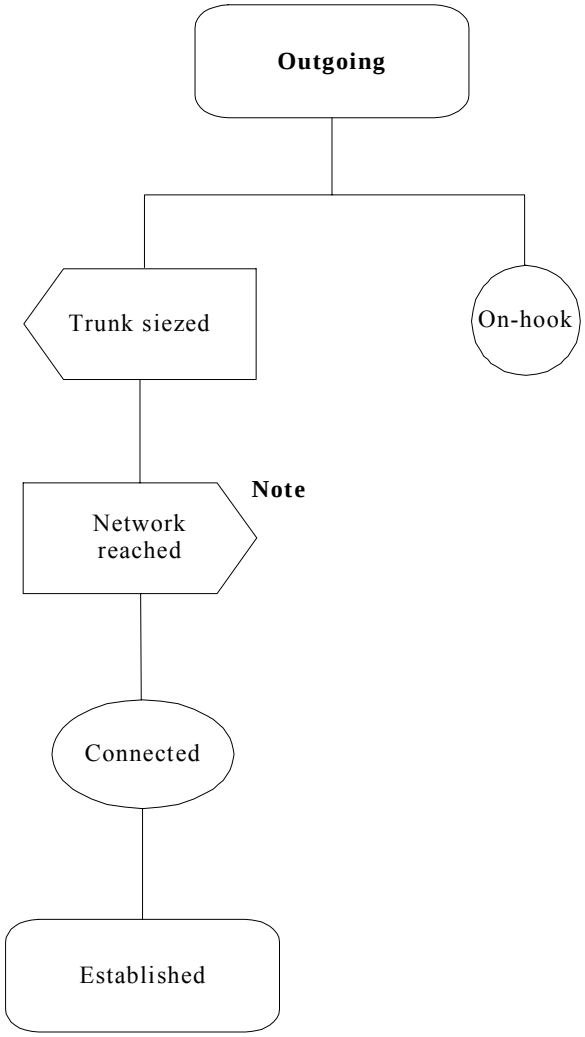

Call clearing
(bt)
Clear Connection Clear Call
Connection cleared
Call Cleared
Null
Null
Call abandoned
(rbt)
Calling party dropped
Connection Cleared Caller
Connection Cleared Called
Call cleared
Null
Null

Outbound Station Call



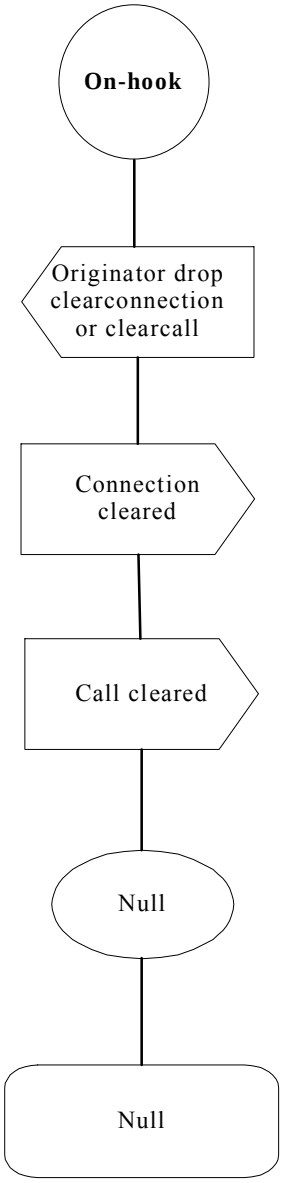
Note: For call forward immediate, no events are sent.

Outgoing

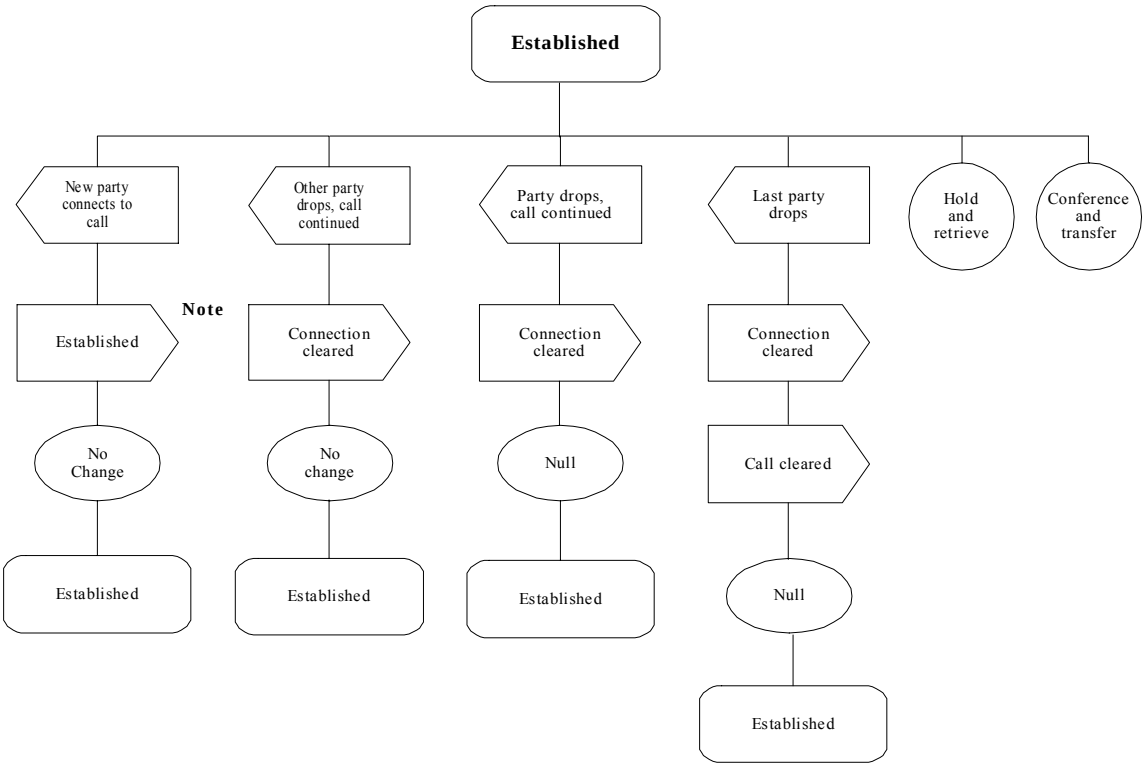


Note: There will be no established event report for this call after the network reached event.

On-hook

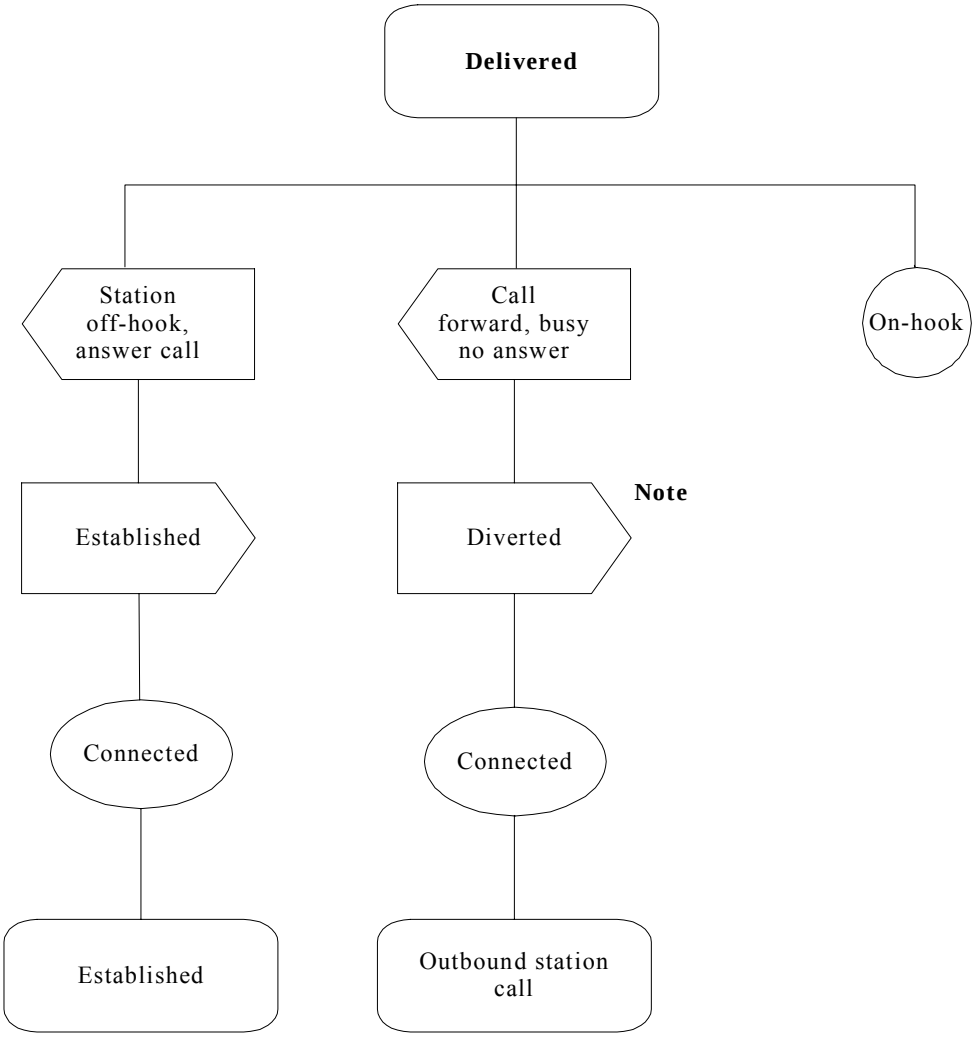

On-hook
Originator drop clearconnection or clearcall
Connection cleared
Call cleared
Null
Null

Established



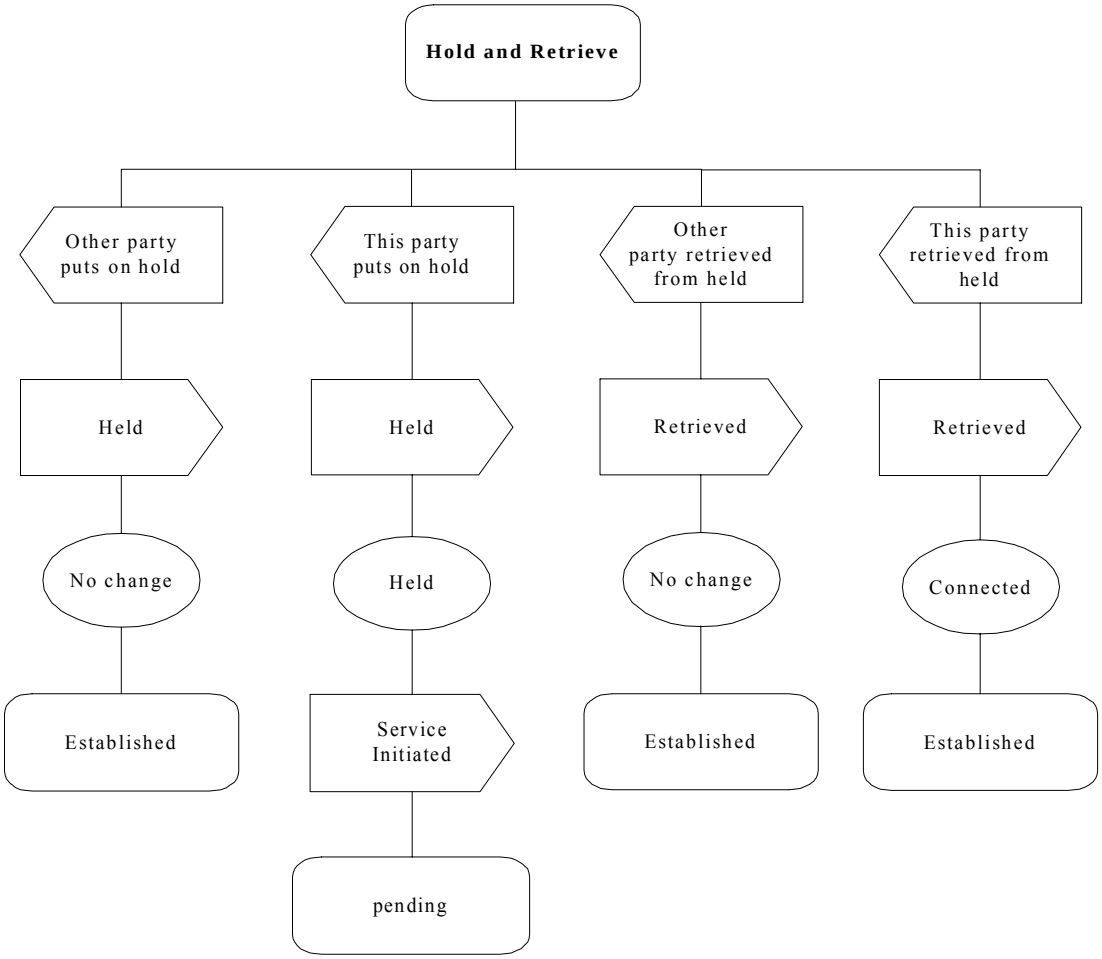
Note: Every party on the call will receive this event report.

Delivered



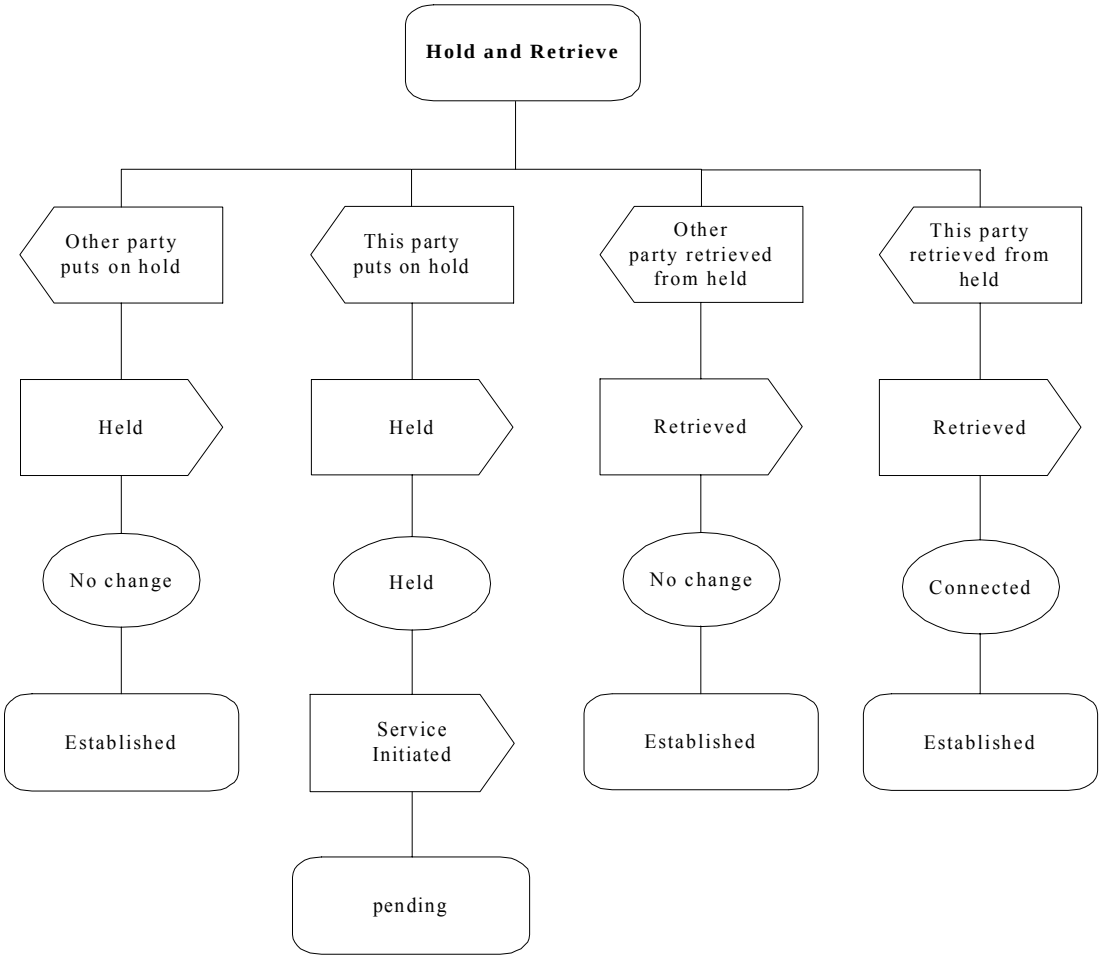
Note: The Diverted Event is sent to the station the call is being diverted away from. No more events for this call are sent to that station.

Hold and Retrieve



Note: The Held and Retrieved event reports are sent to every device that is on the call.

Conference and Transfer

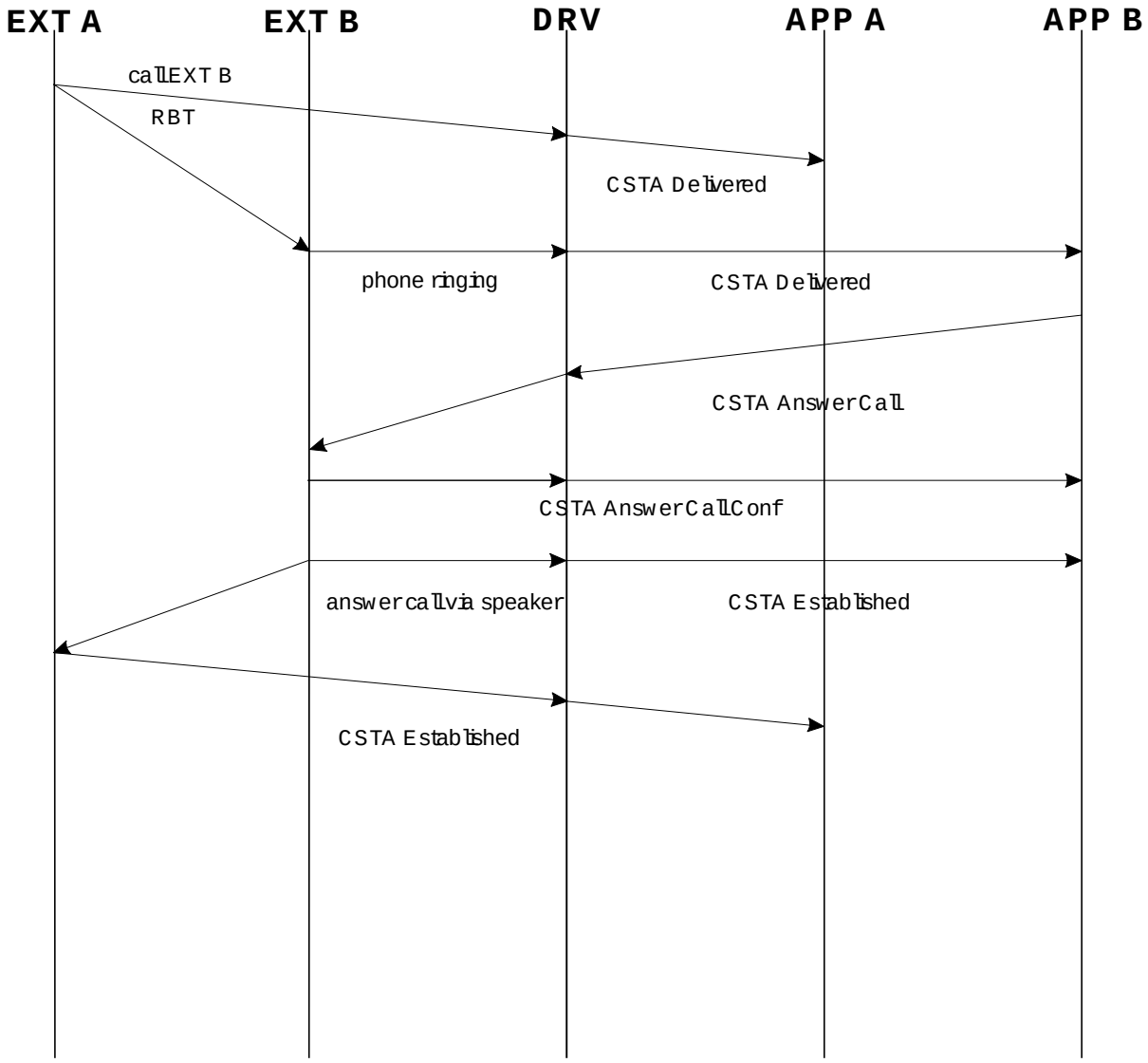


Note: The Held and Retrieved event reports are sent to every device that is on the call.

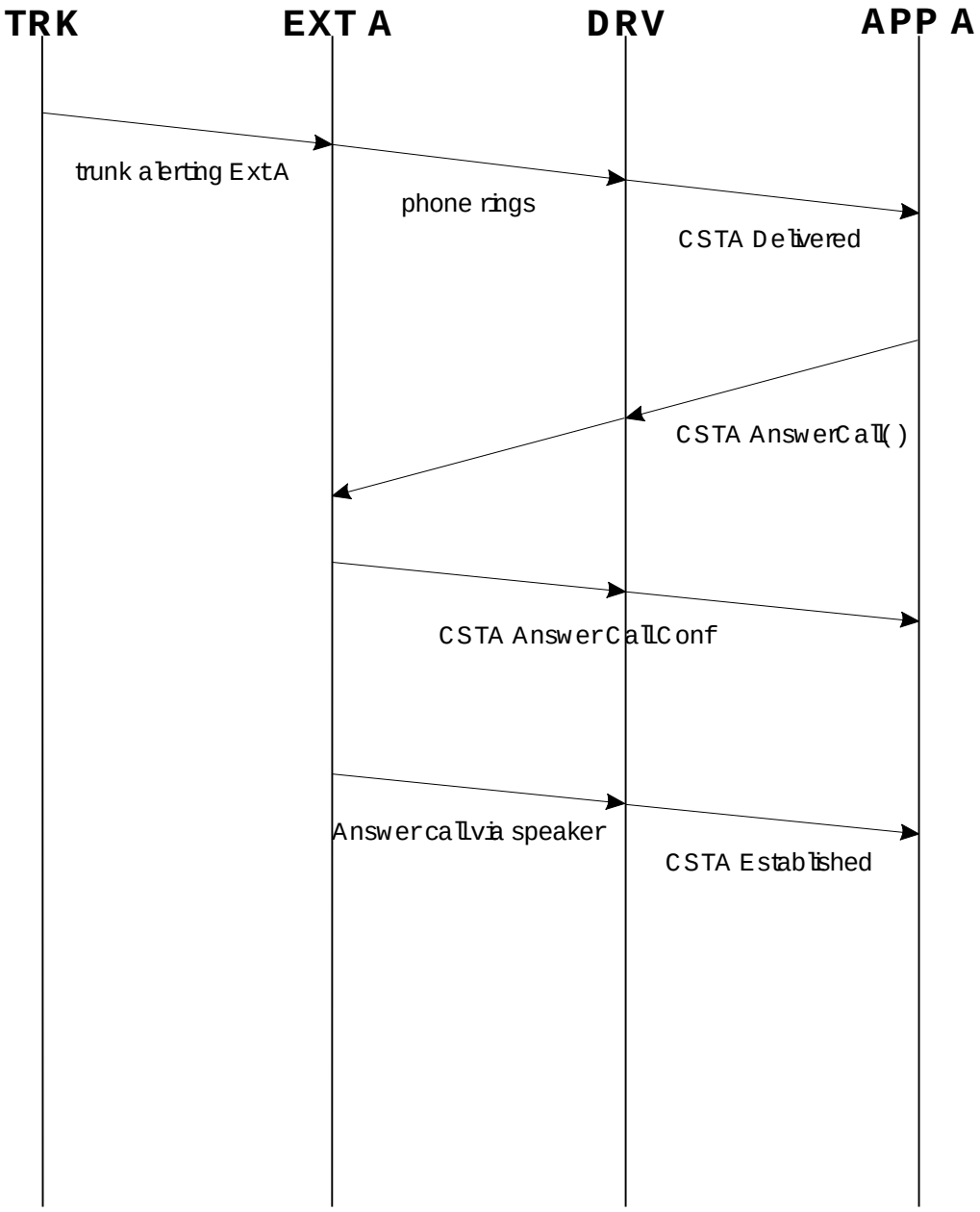
Chapter 10. CSTA Timing Diagrams

Call Control Service Group

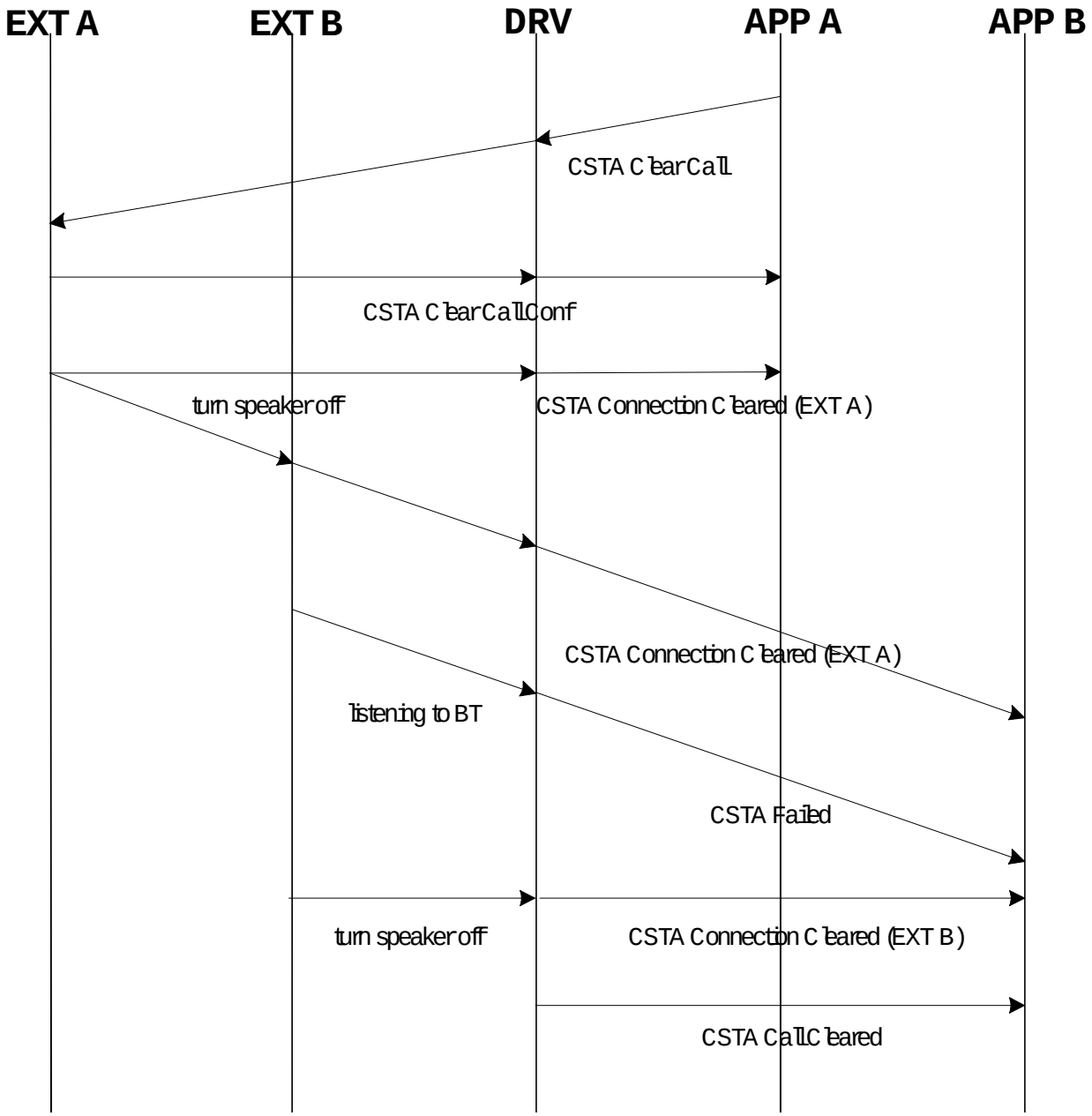
cstaAnswerCall - intercom call



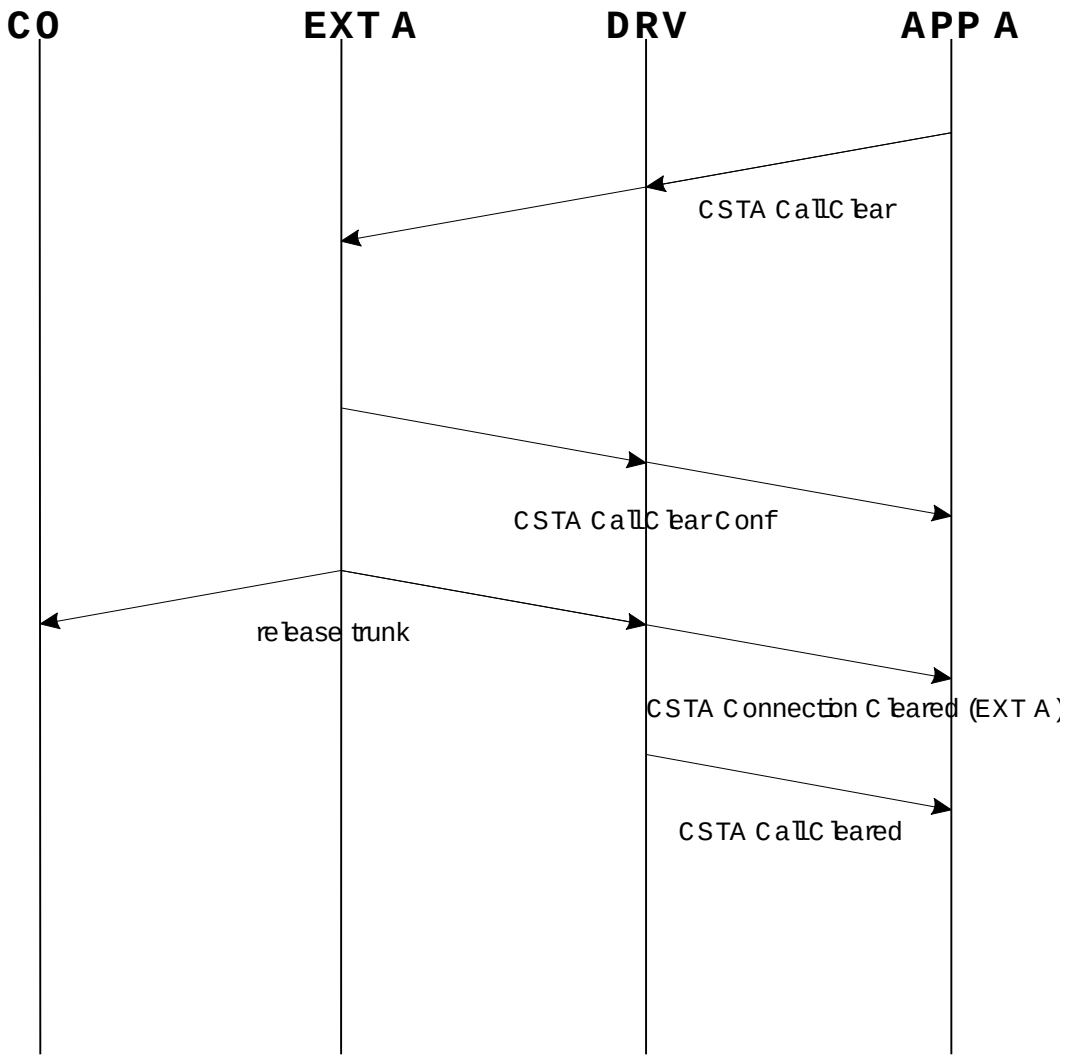
cstaAnswerCall - trunk call



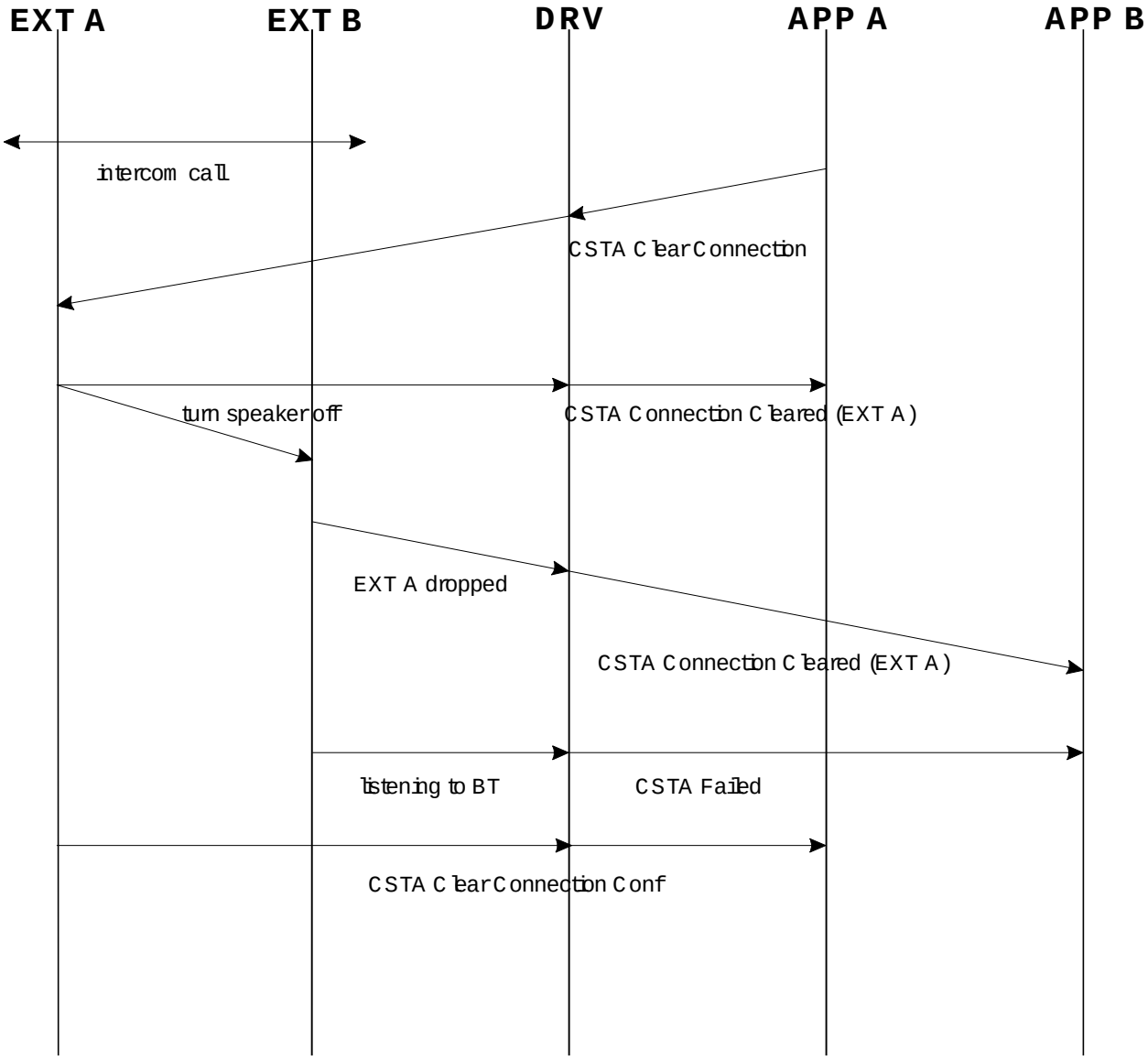
cstaClearCall - intercom call



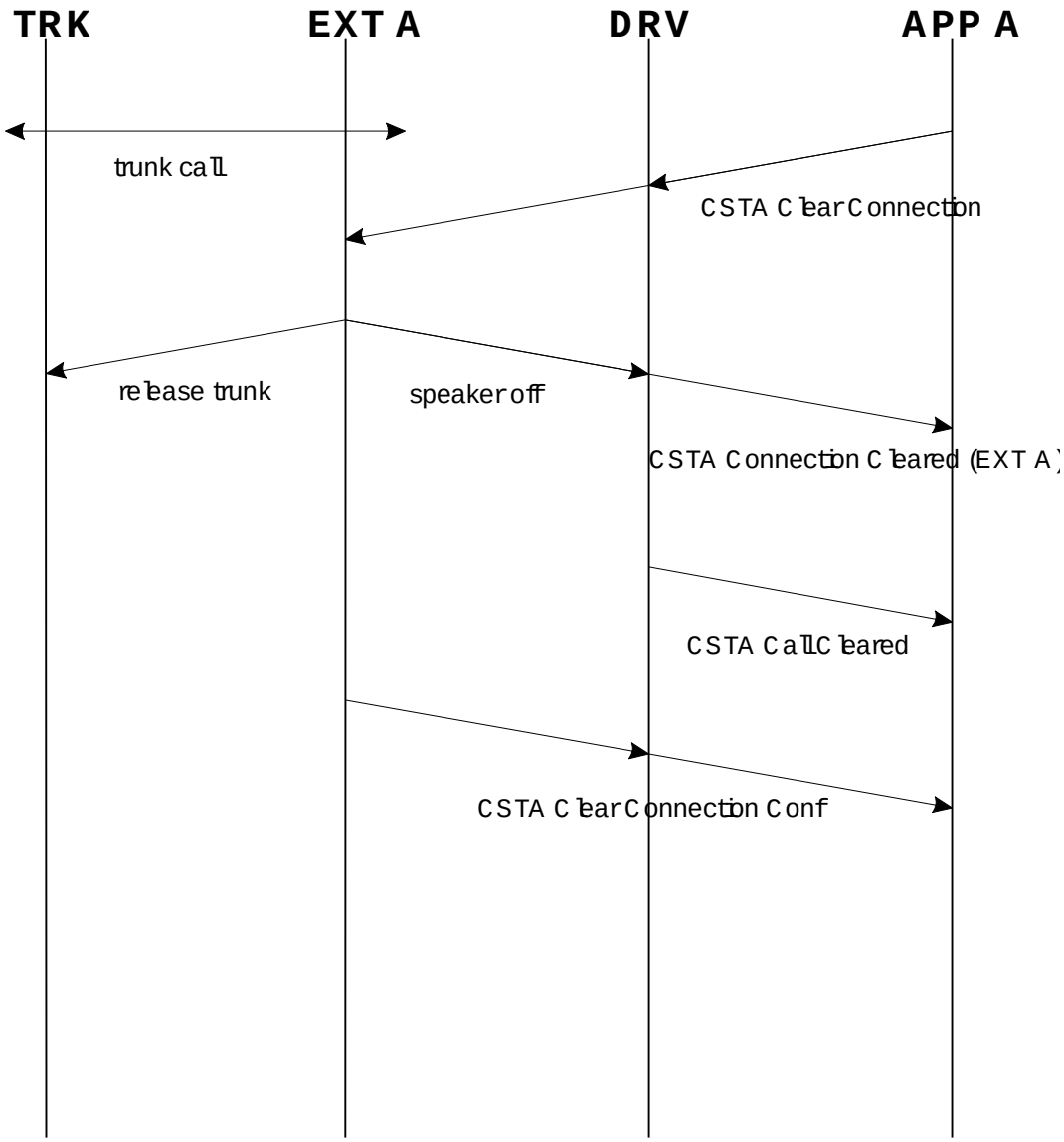
cstaClearCall - trunk call



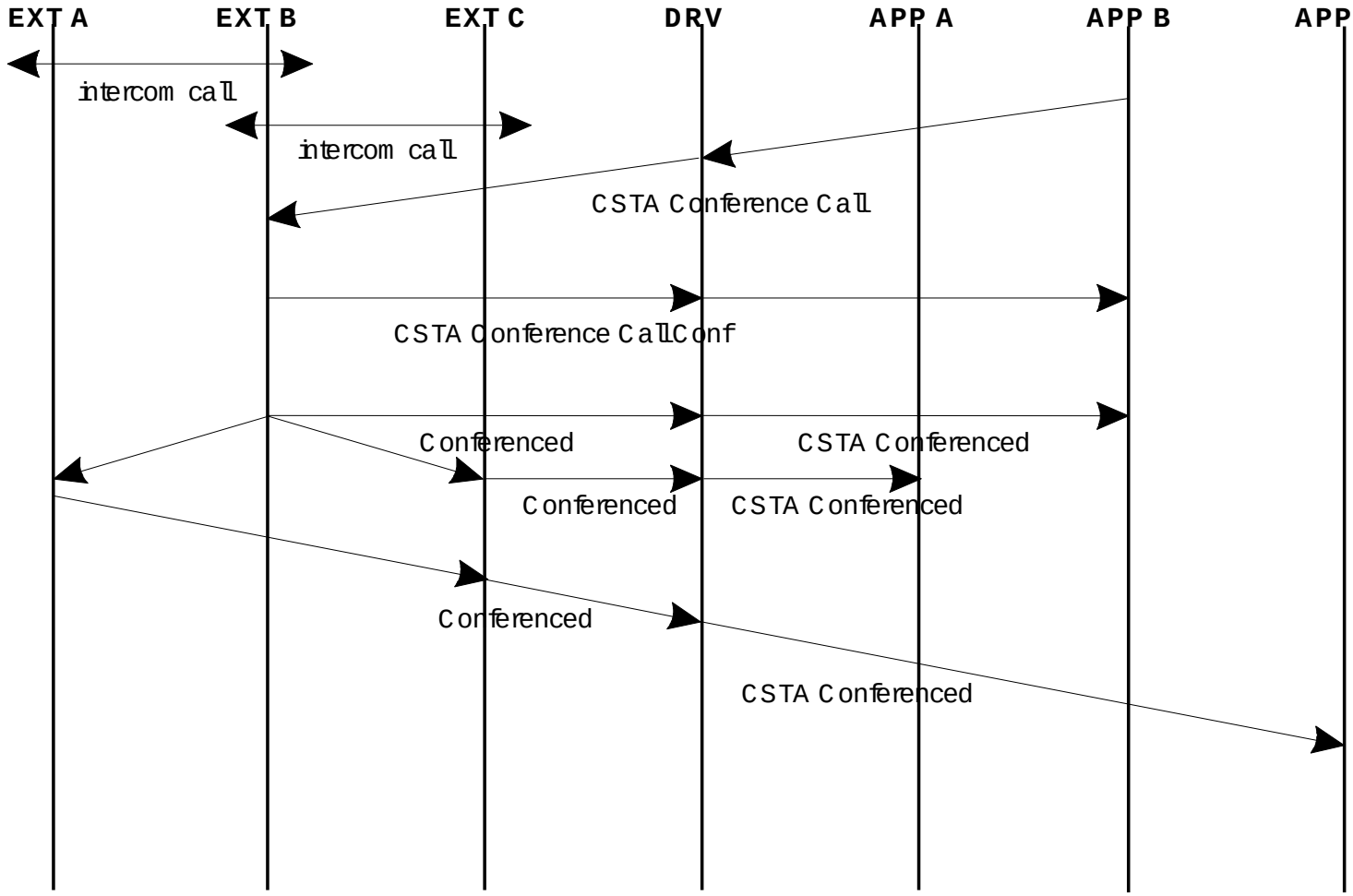
cstaClearConnection - intercom call



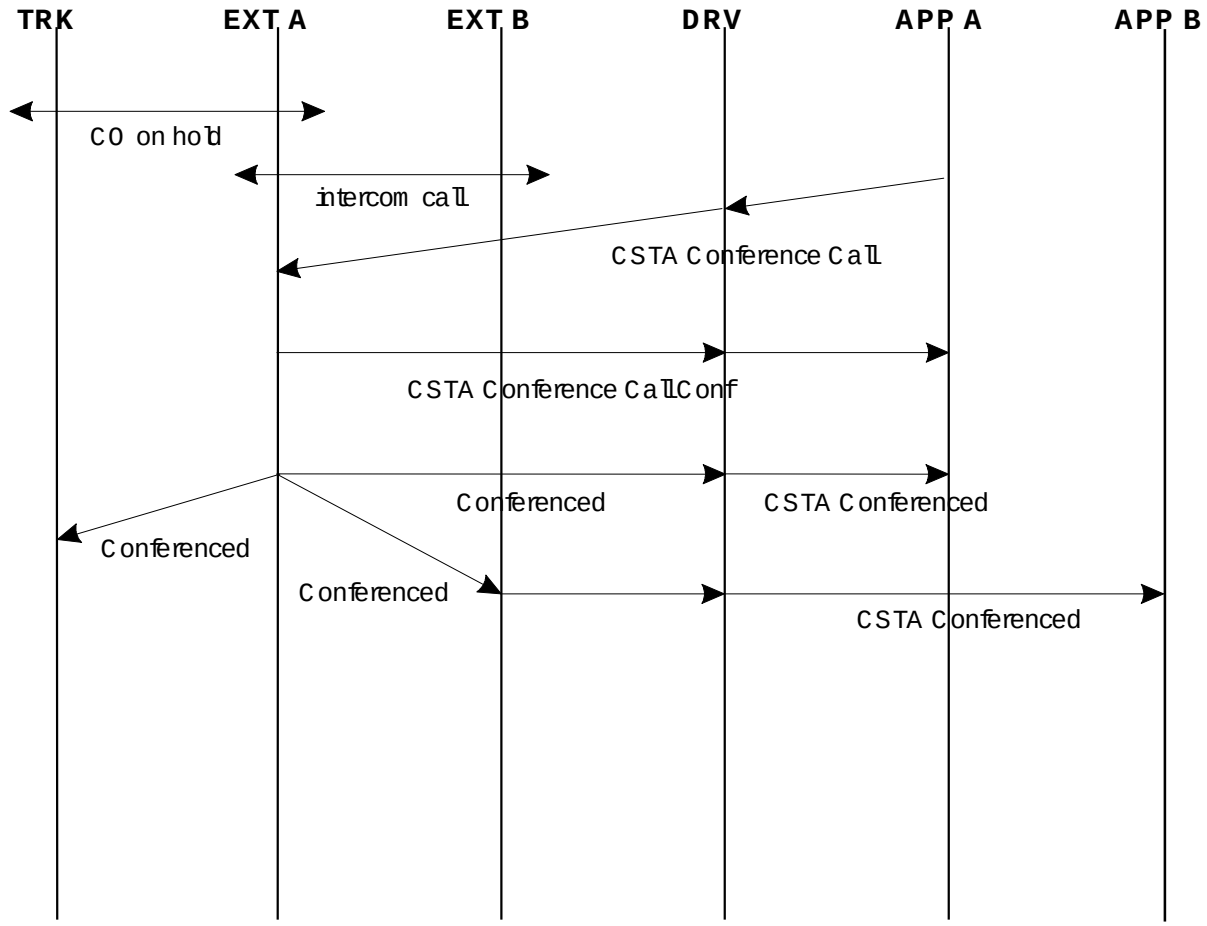
cstaClearConnection - trunk call



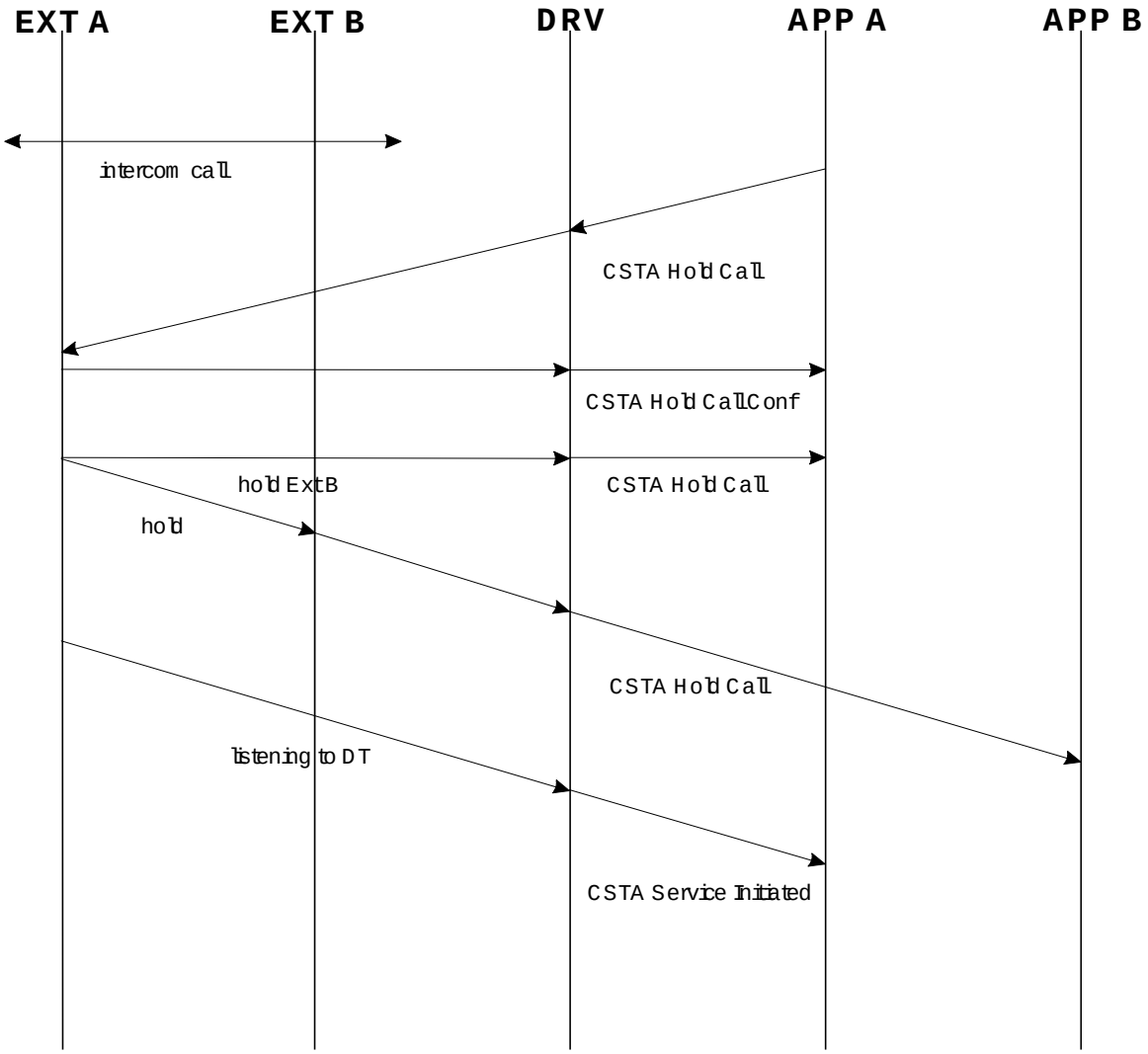
cstaConferenceCall - intercom call



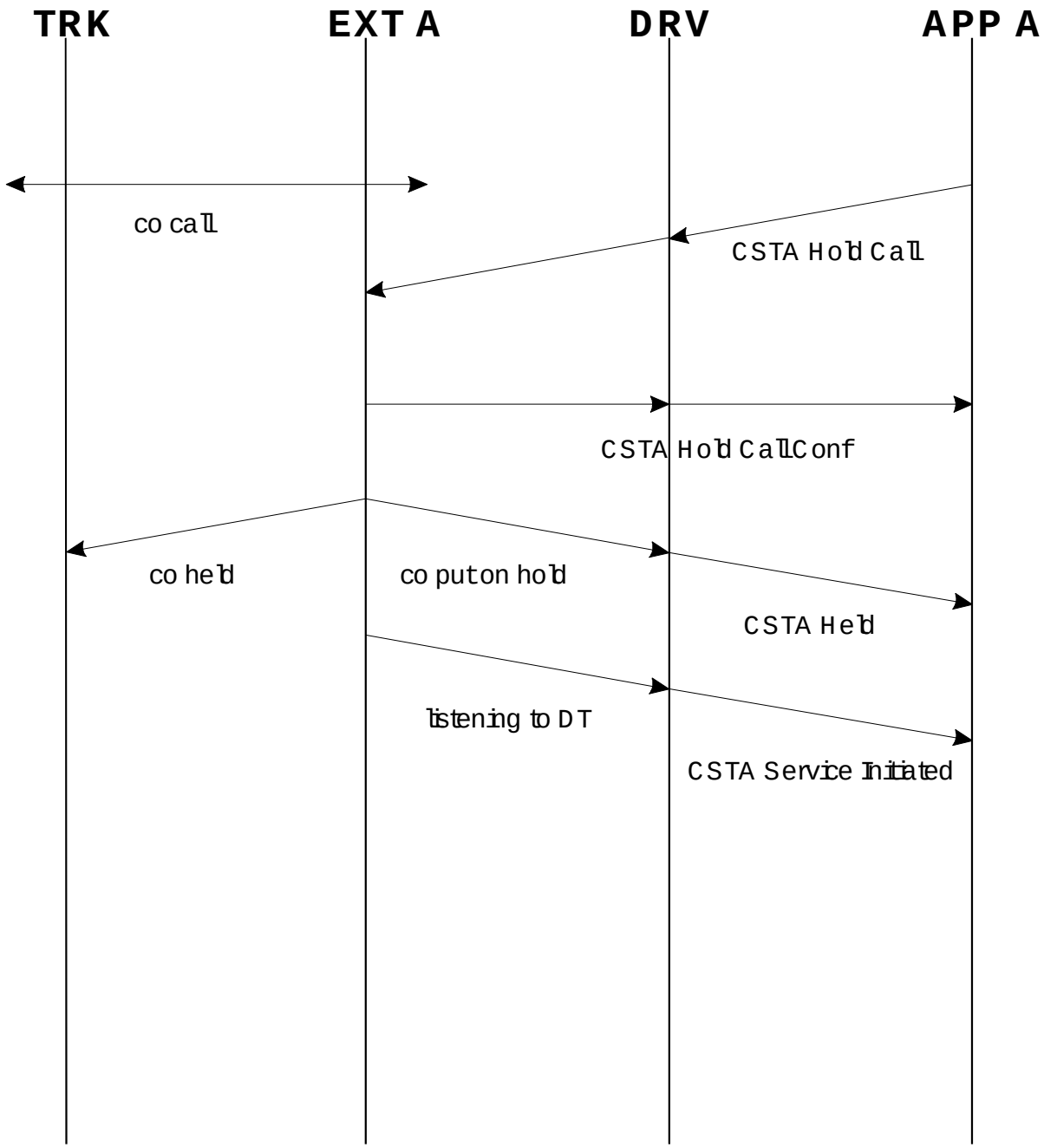
cstaConferenceCall - trunk & intercom call



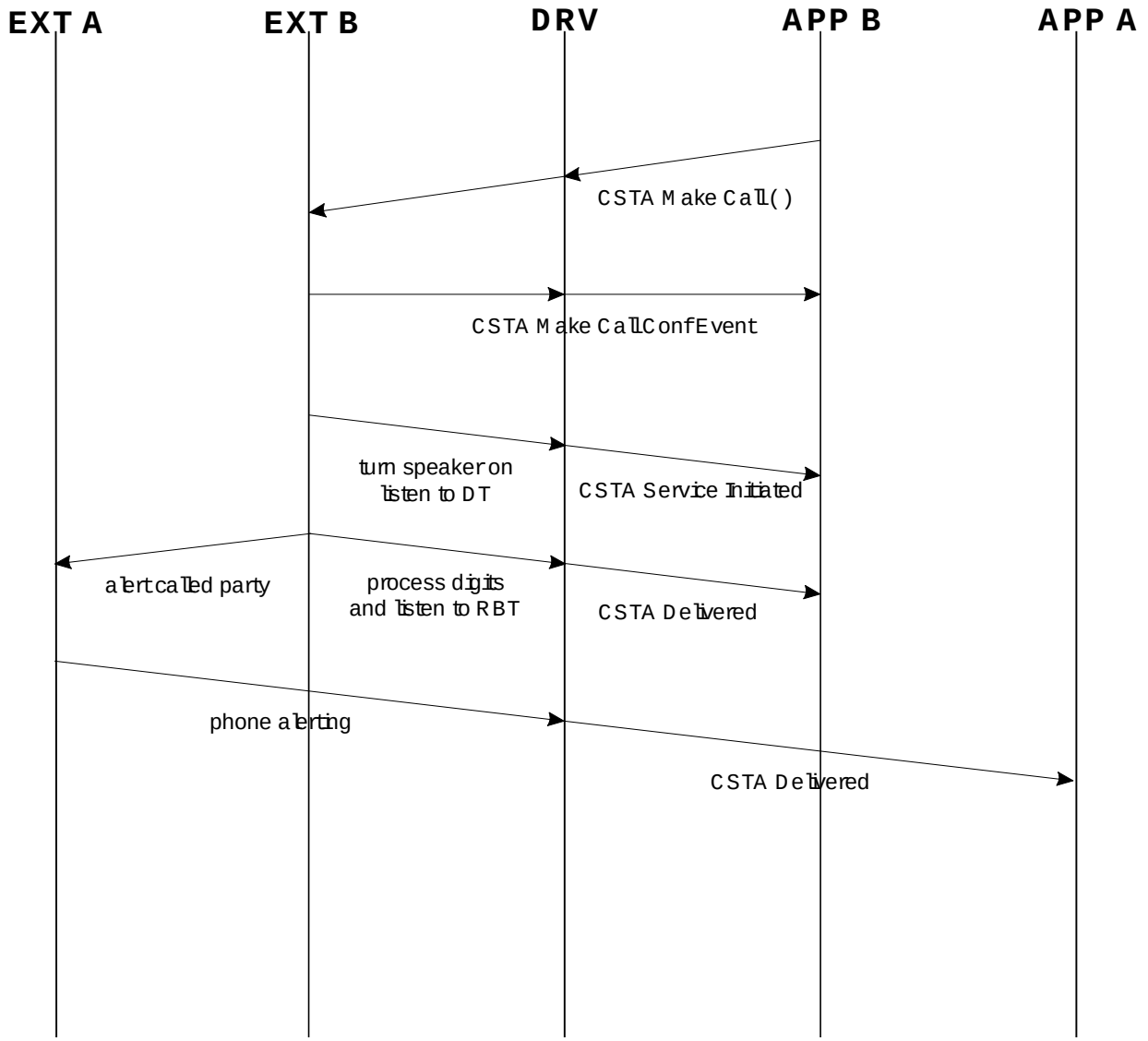
cstaHoldCall - intercom call



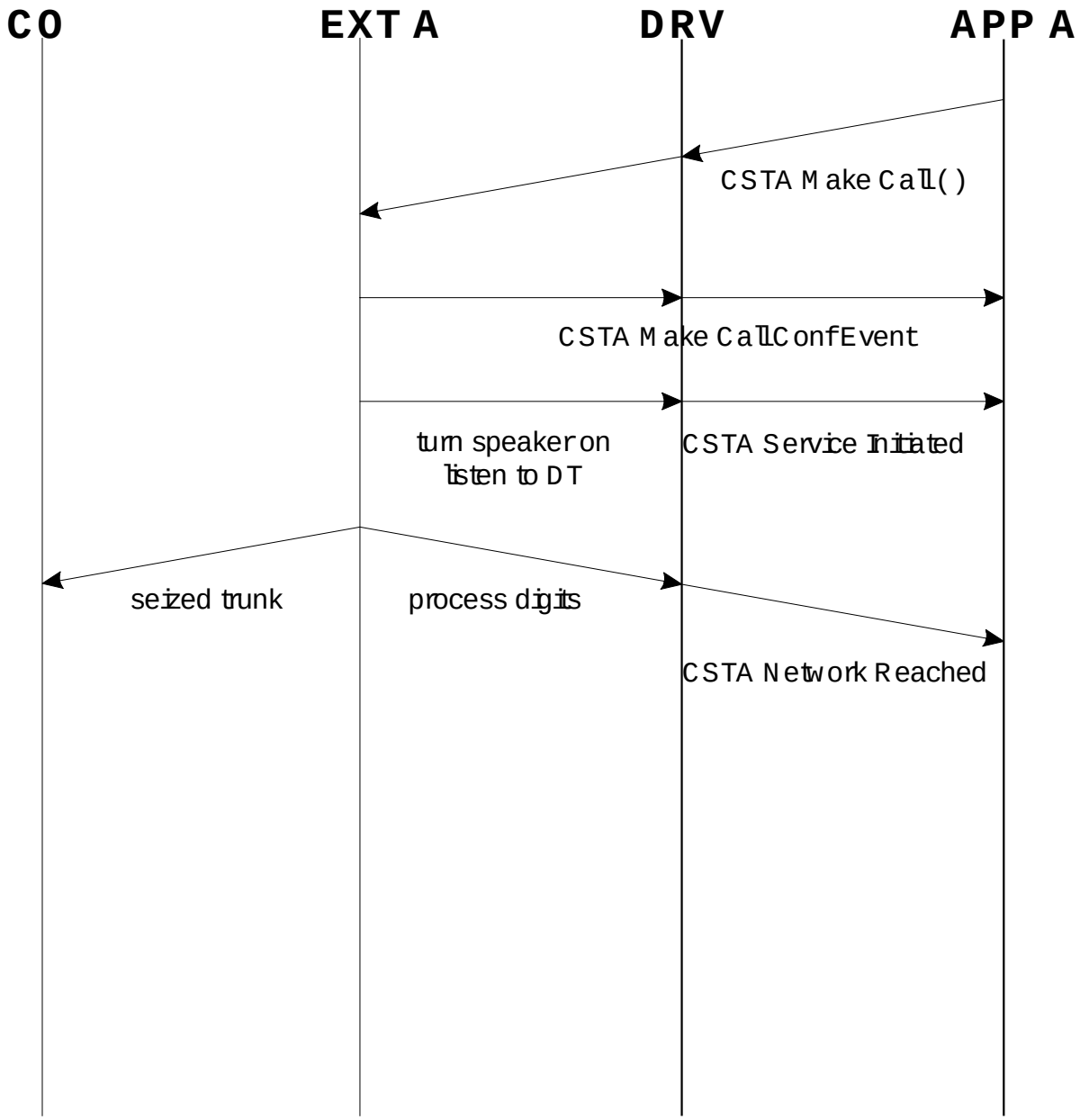
cstaHoldCall - trunk call



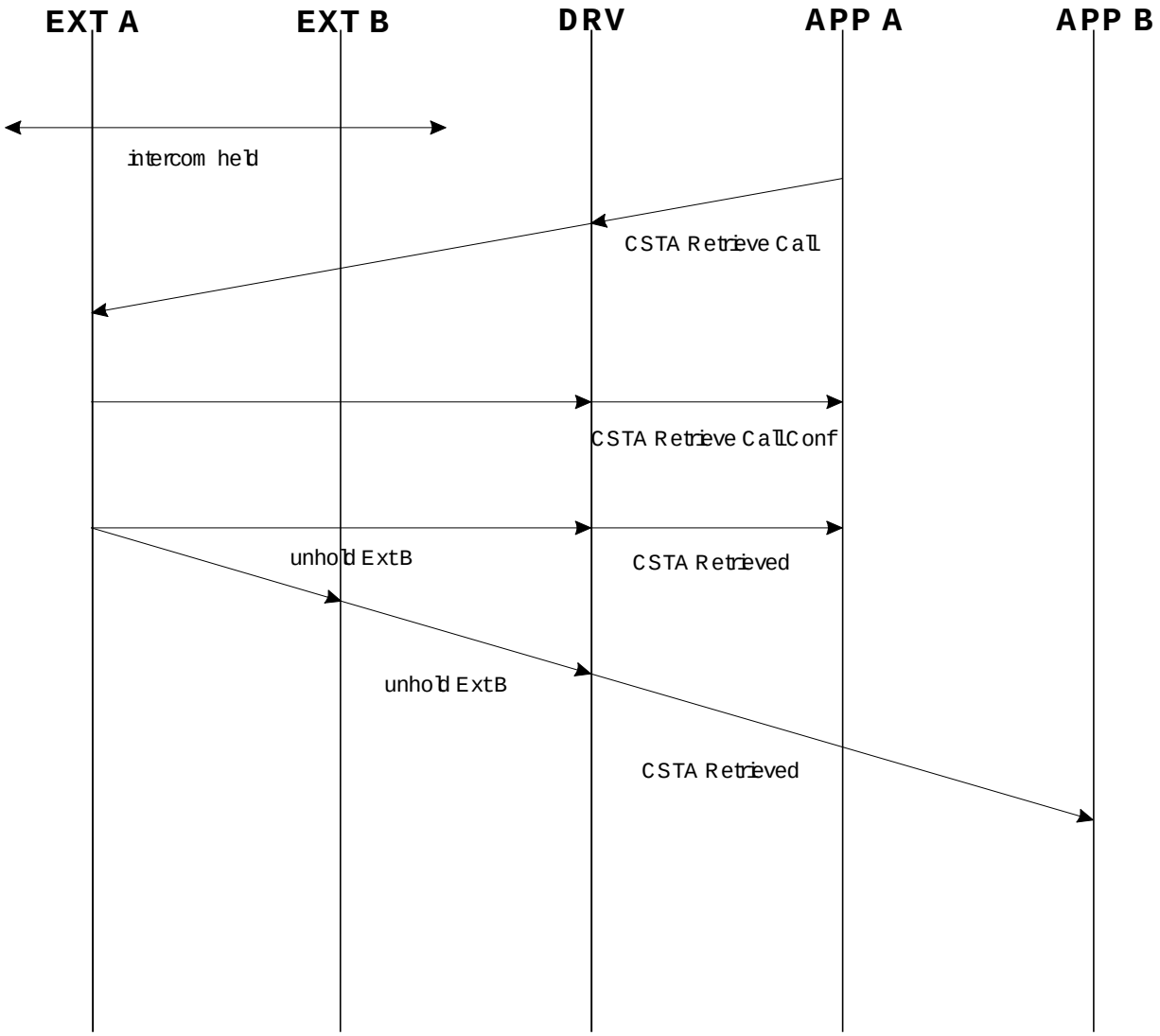
cstaMakeCall - intercom call



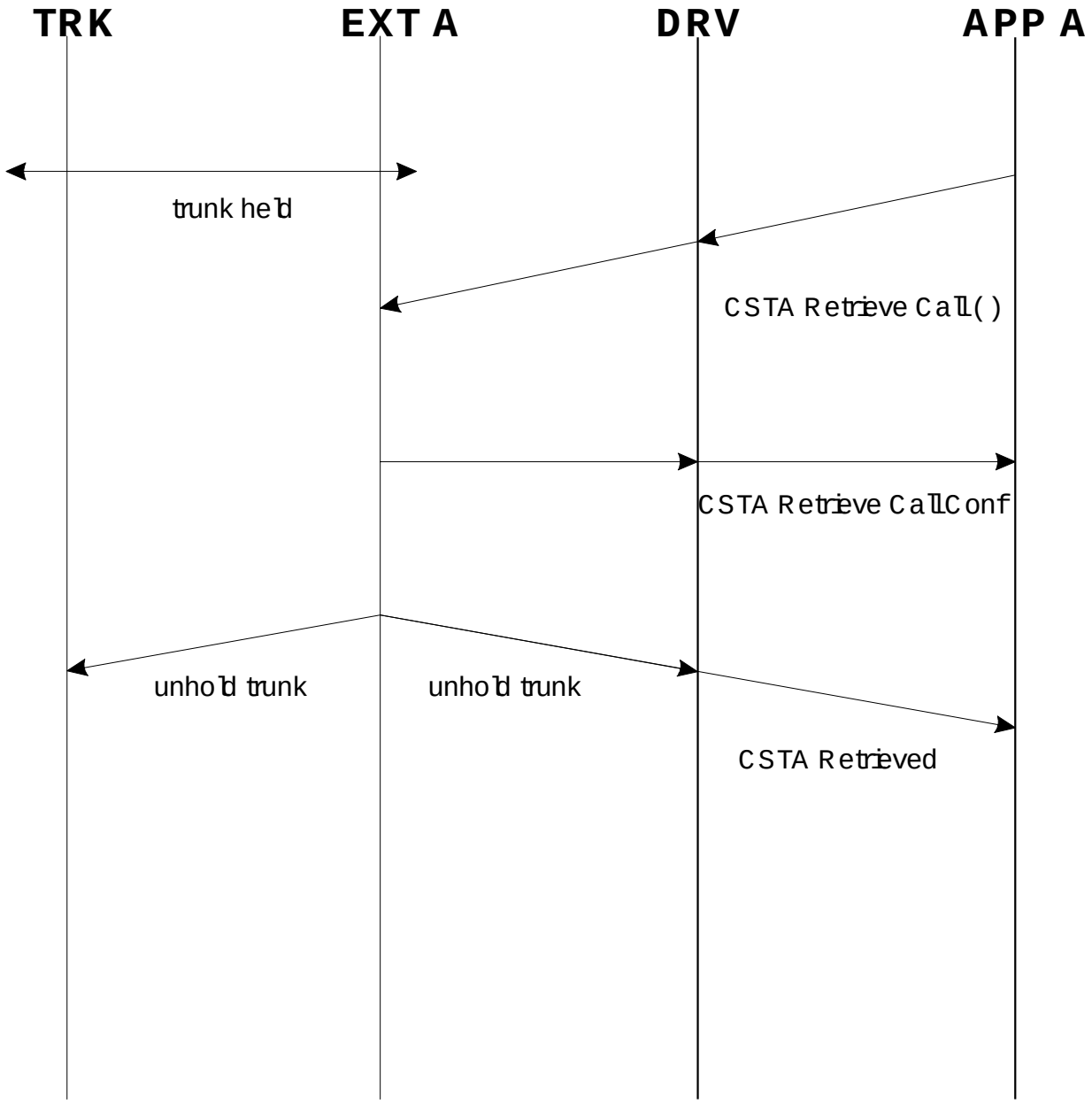
cstaMakeCall - trunk call



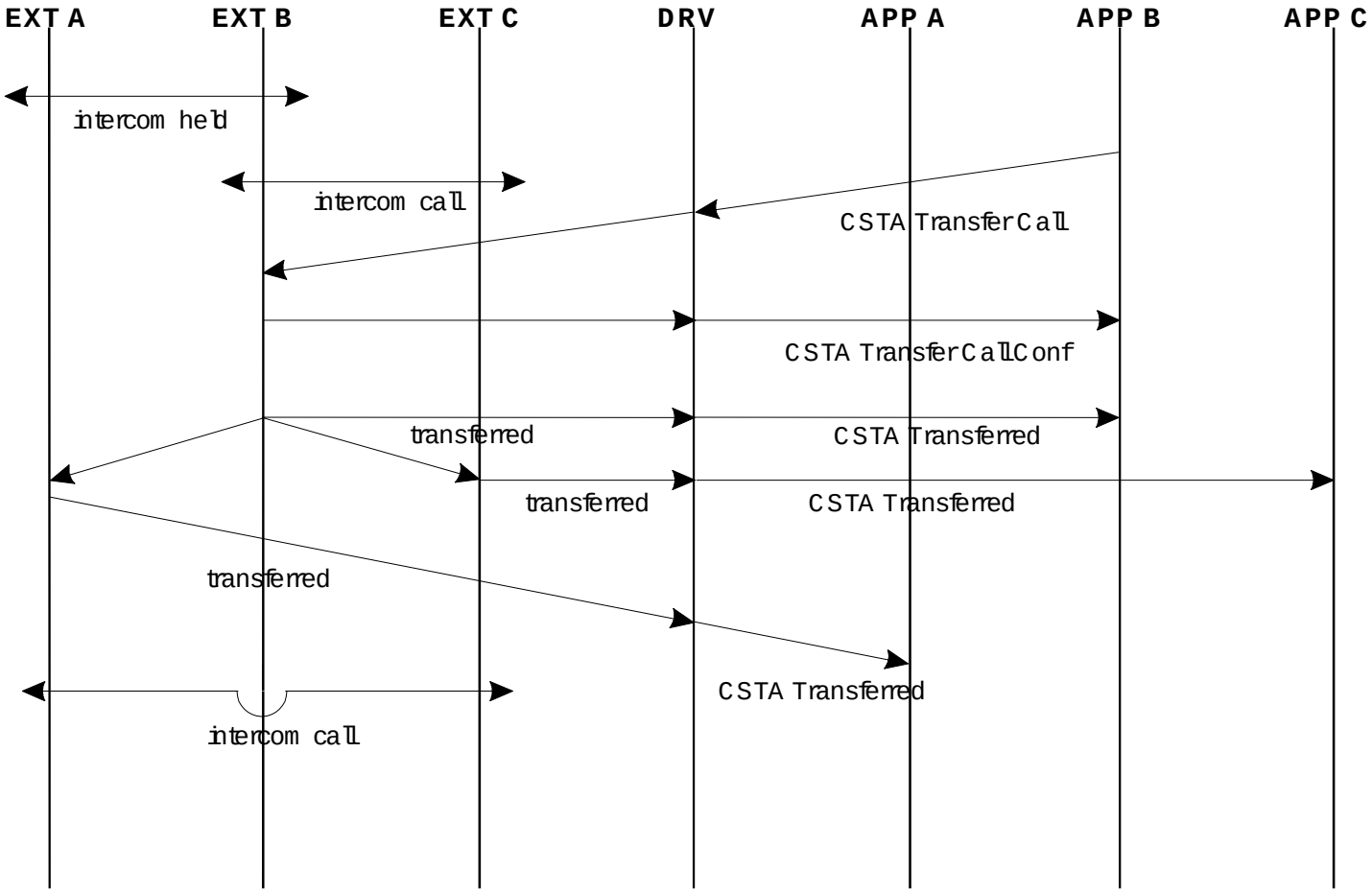
cstaRetrieveCall - intercom held



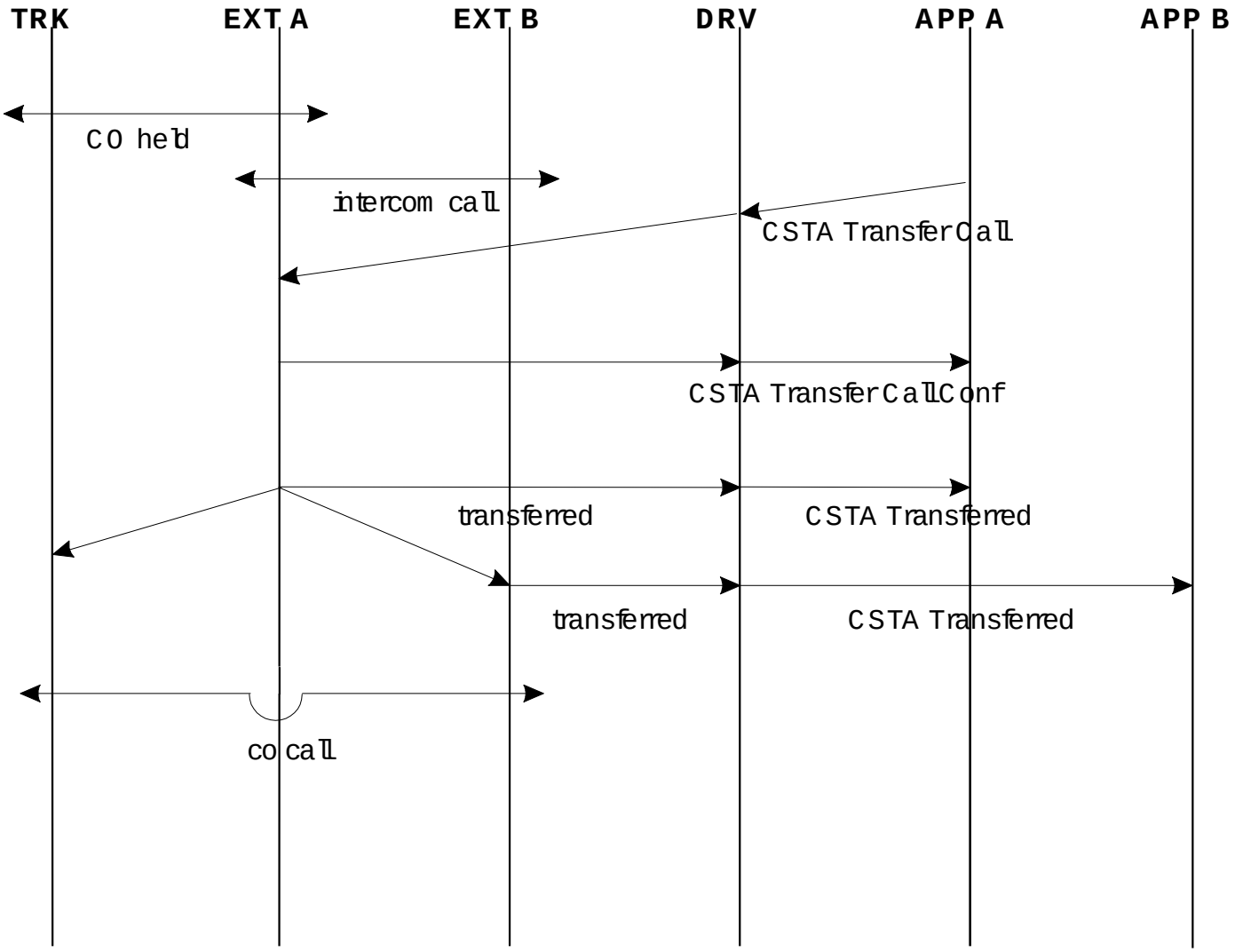
cstaRetrieveCall - CO held



cstaTransferCall - intercom call

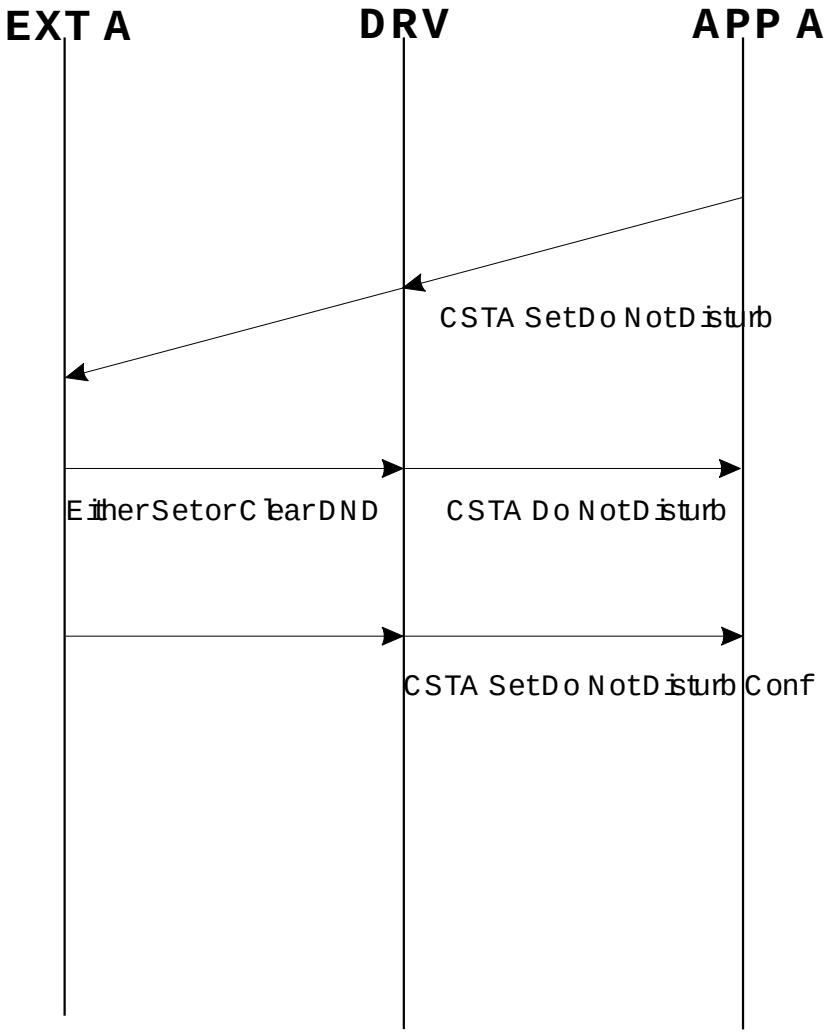


cstaTransferCall - CO Transfer to Extension

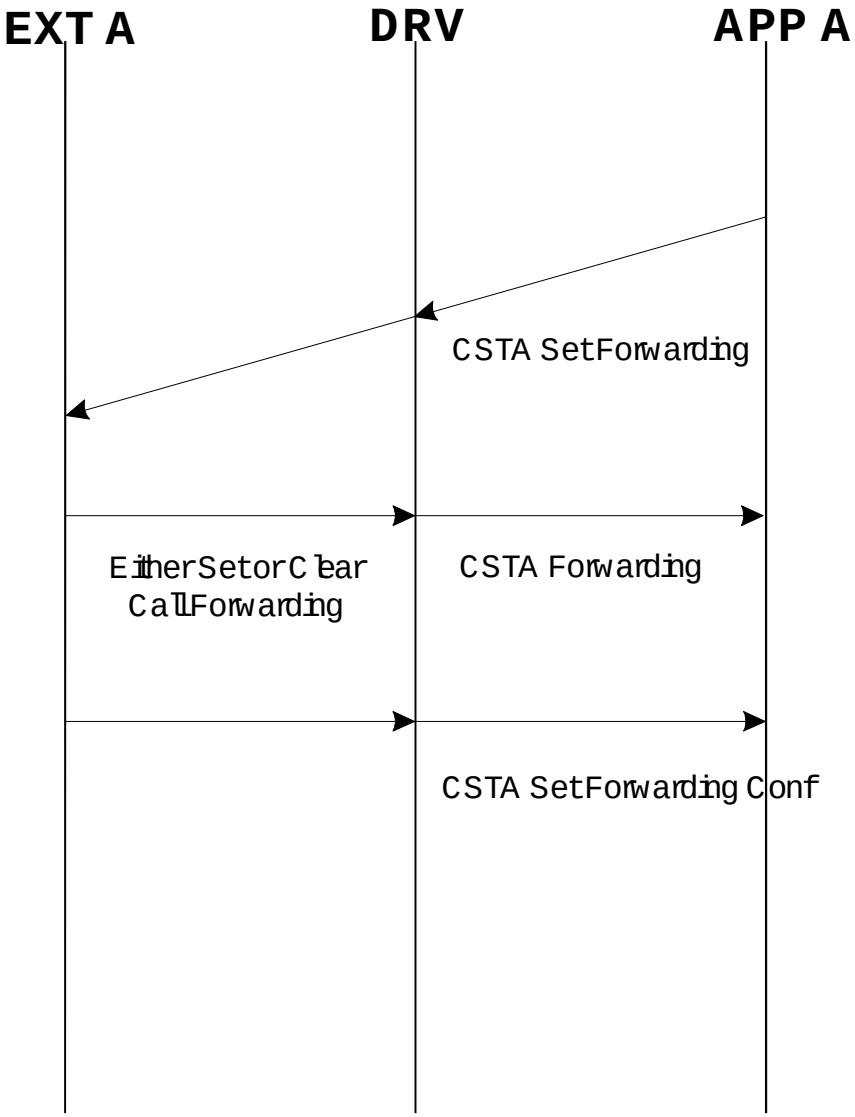


Set Feature Service Group

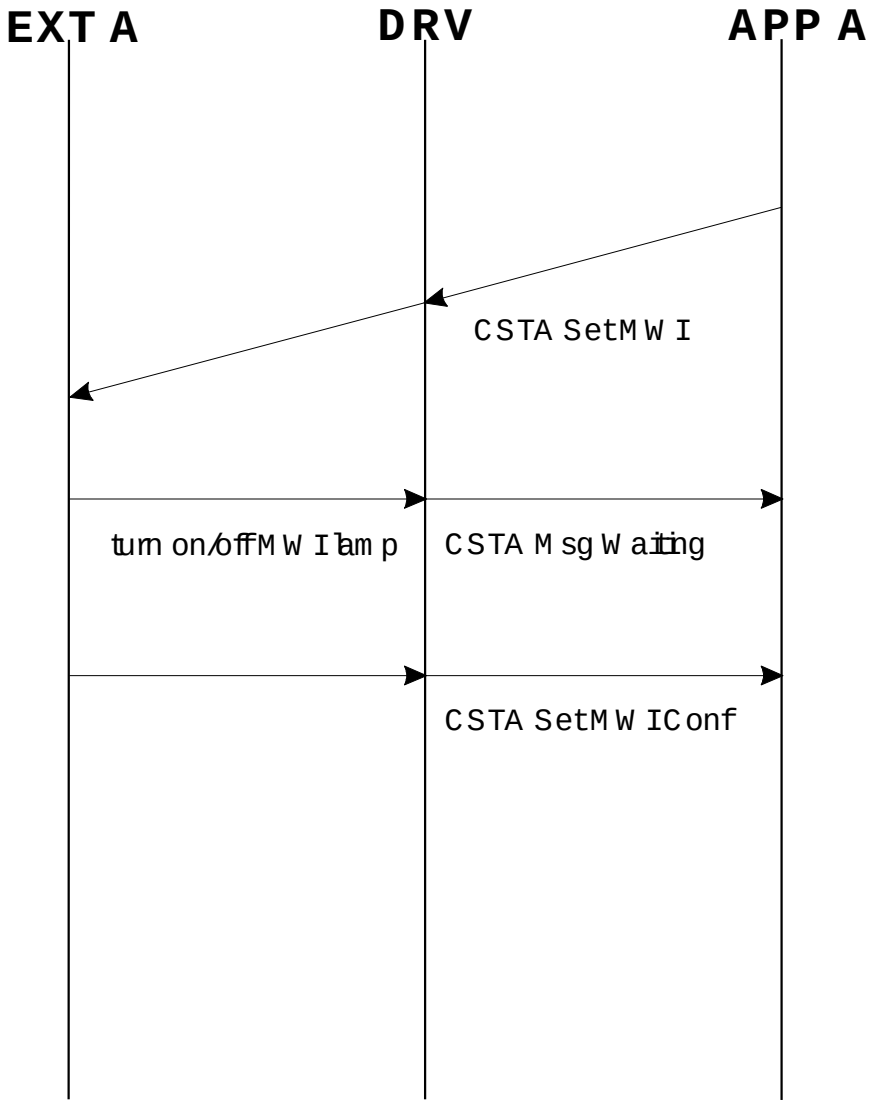
cstaSet Do Not Disturb



cstaSet Forwarding

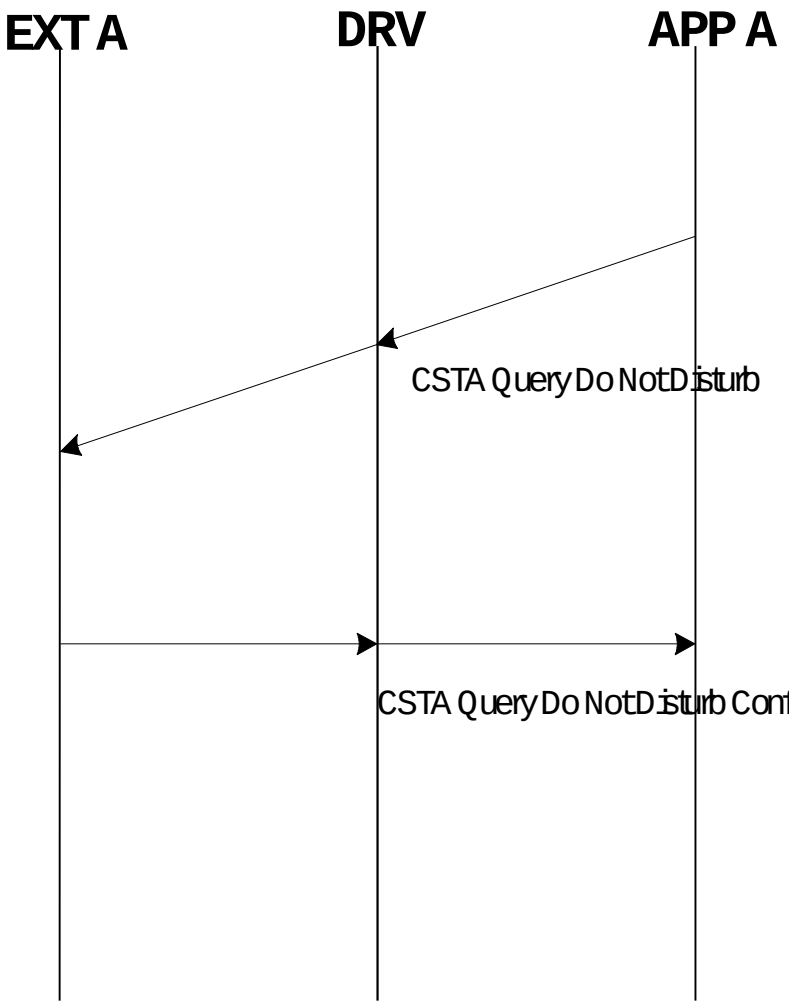


cstaSet Message Waiting Indication

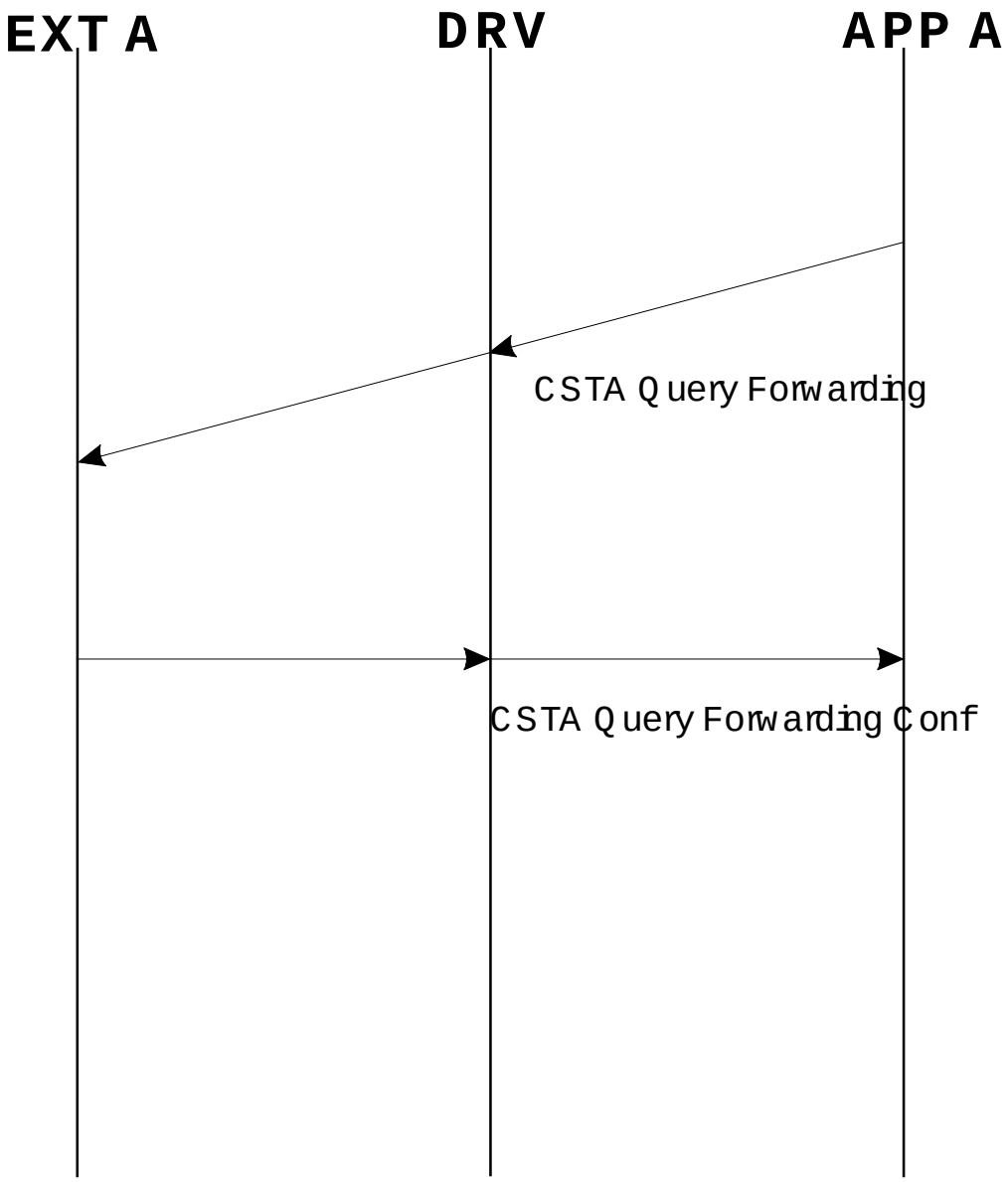


Query Service Group

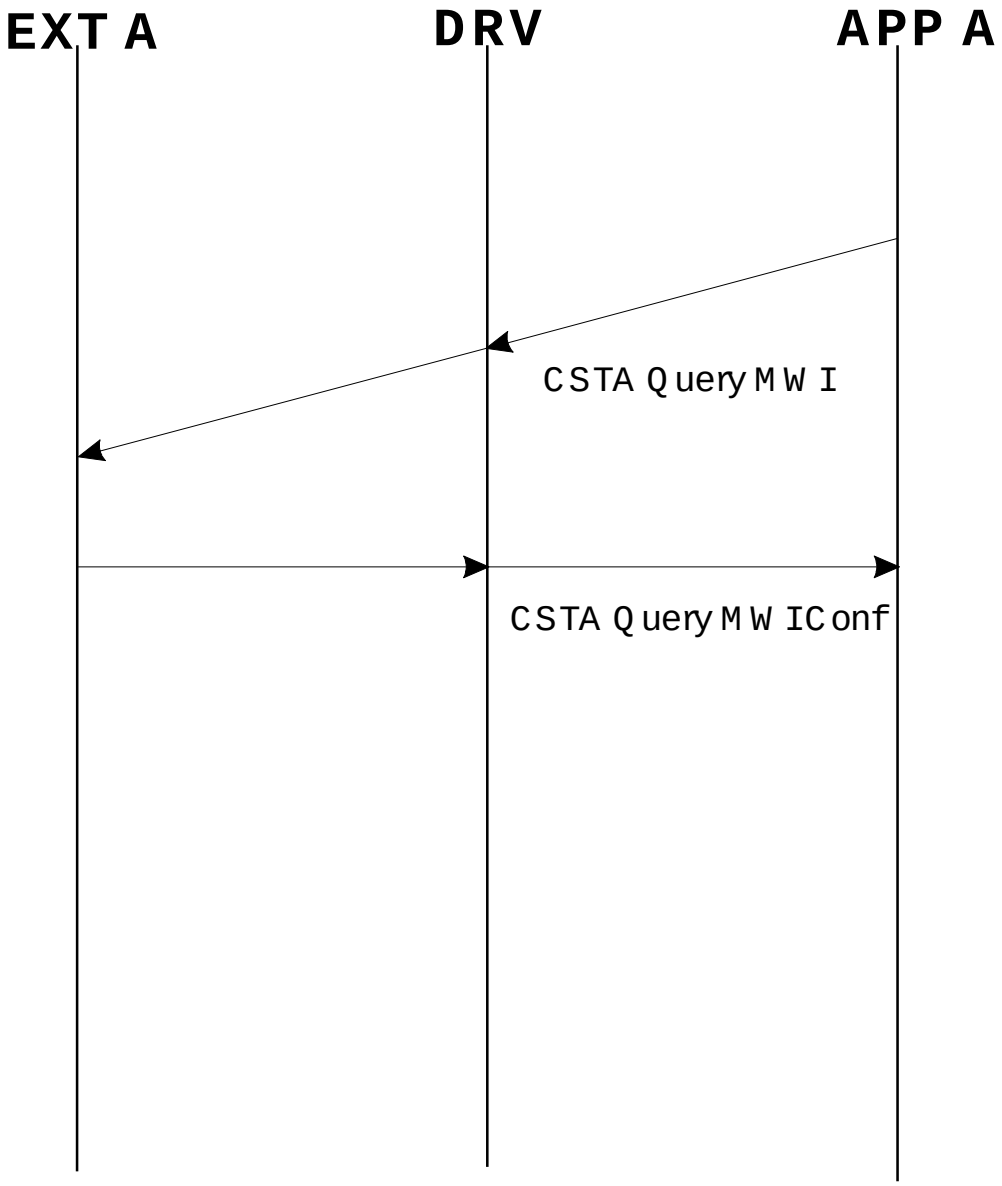
cstaQuery Do Not Disturb



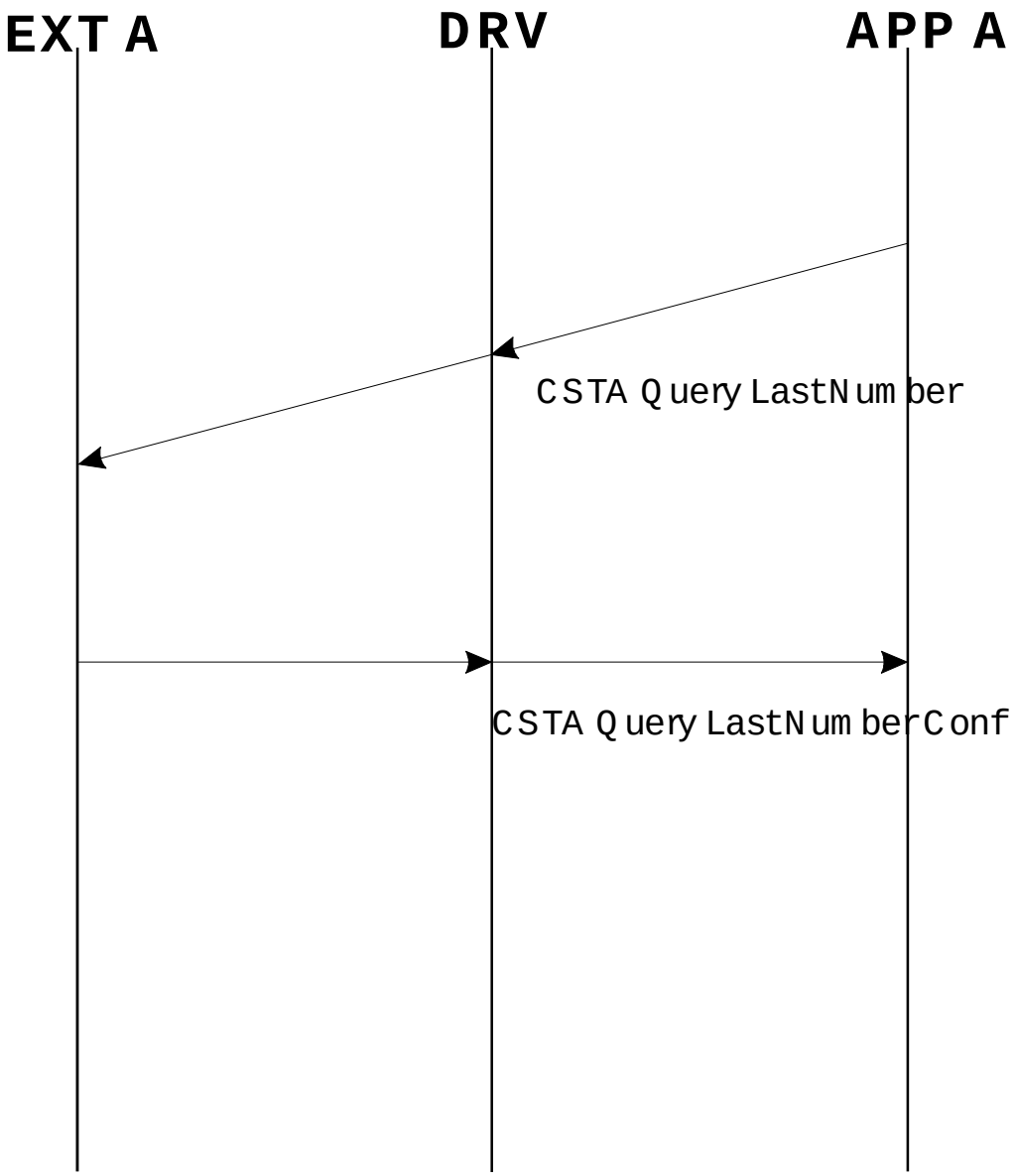
cstaQuery Forwarding



cstaQuery Message Waiting Indication

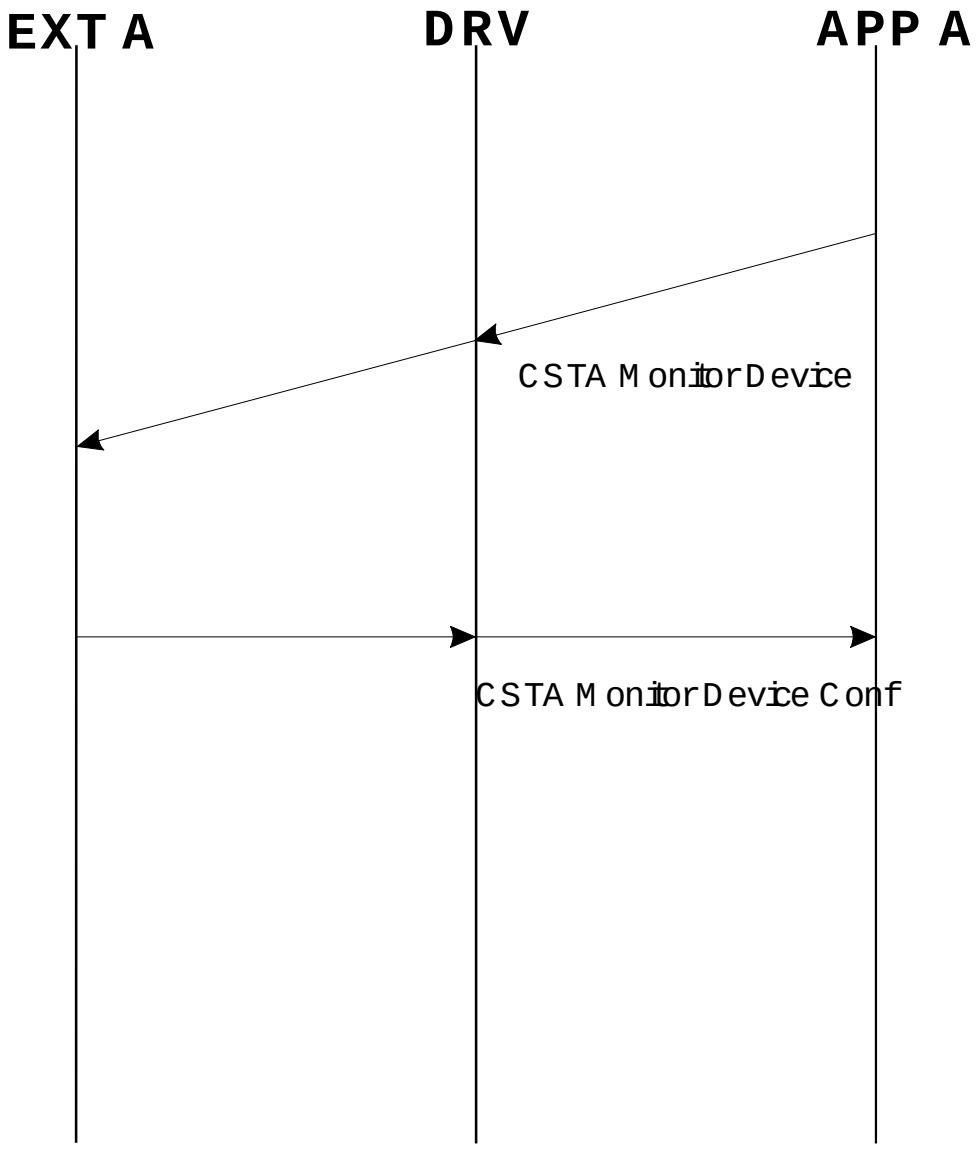


cstaQuery Last Number



Monitor Service Group

cstaMonitor Device

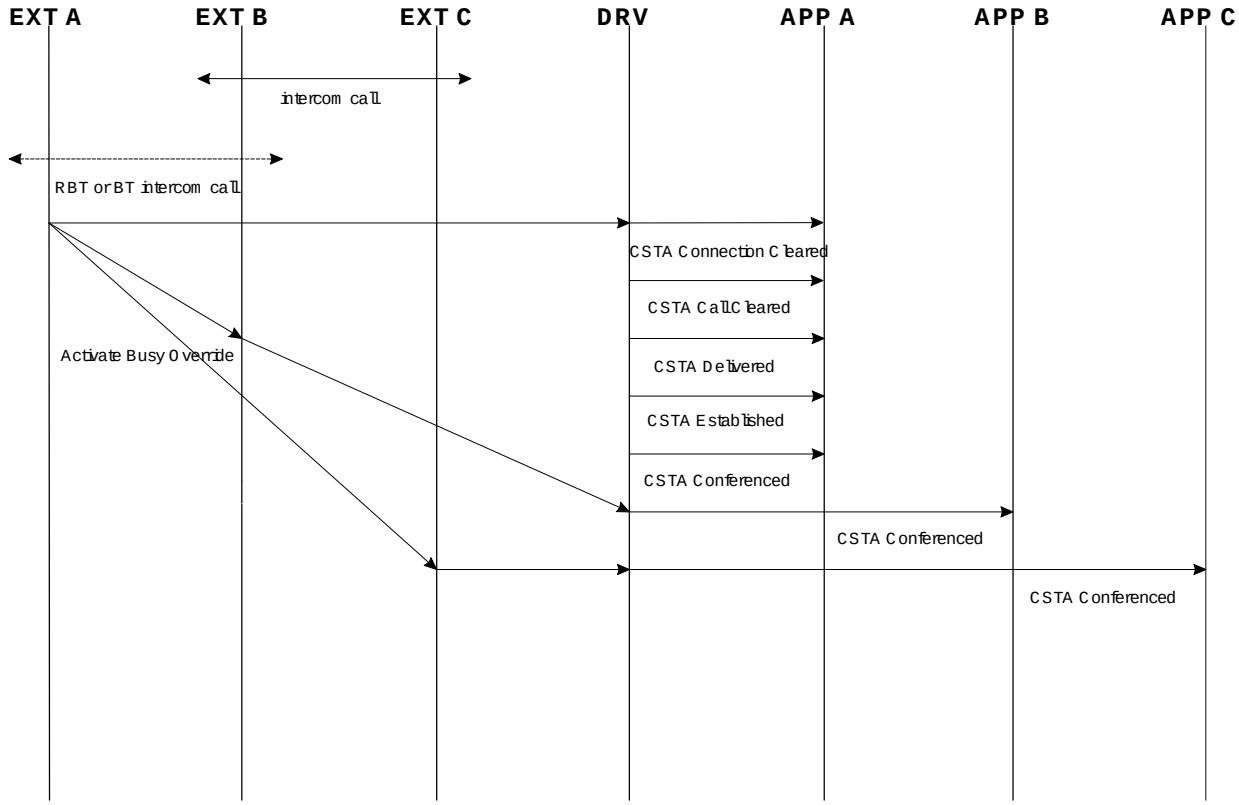


Chapter 11. DBS System Features

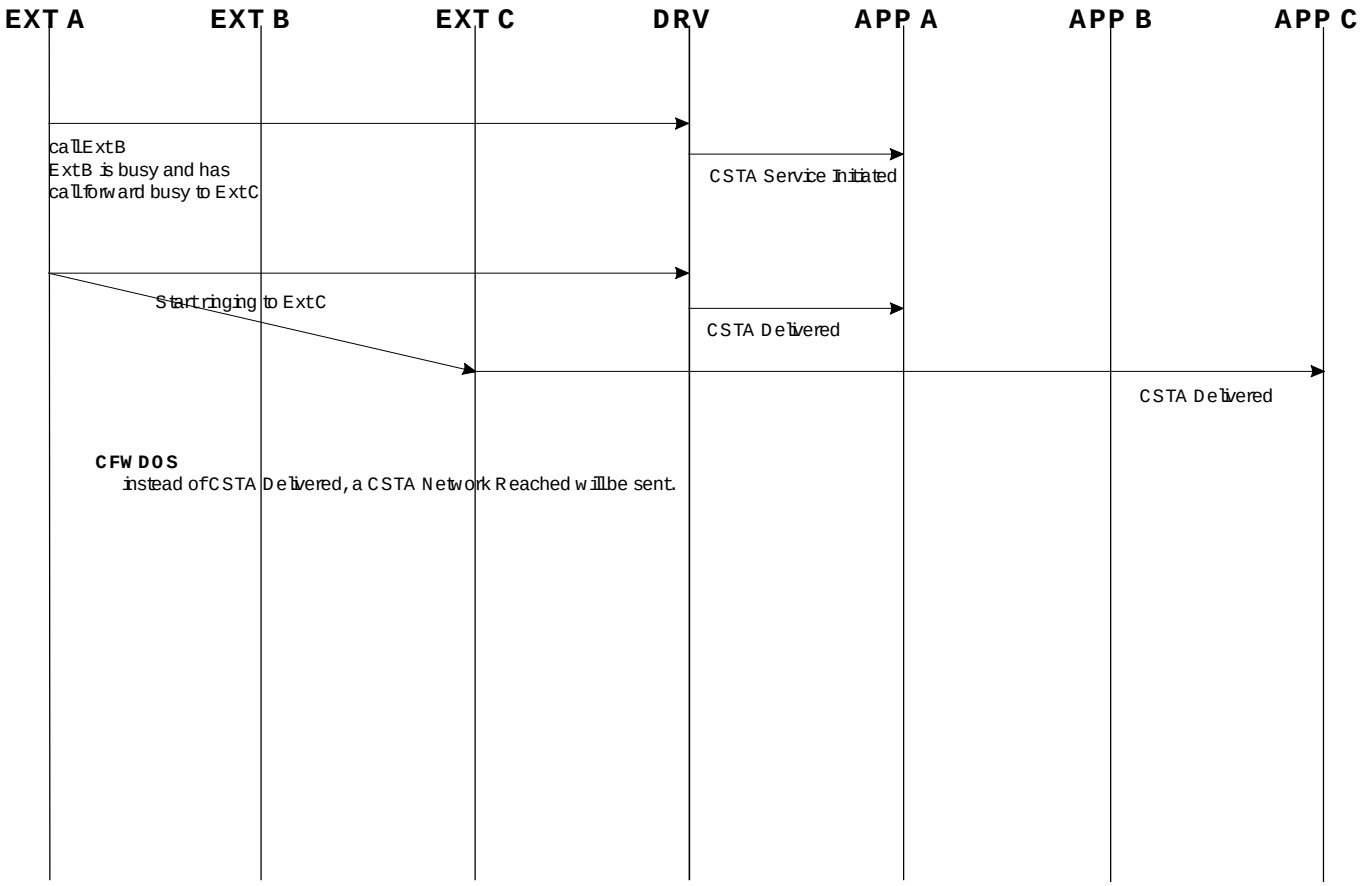
Timing Diagrams

The following illustrations depict event timing for common DBS system features.

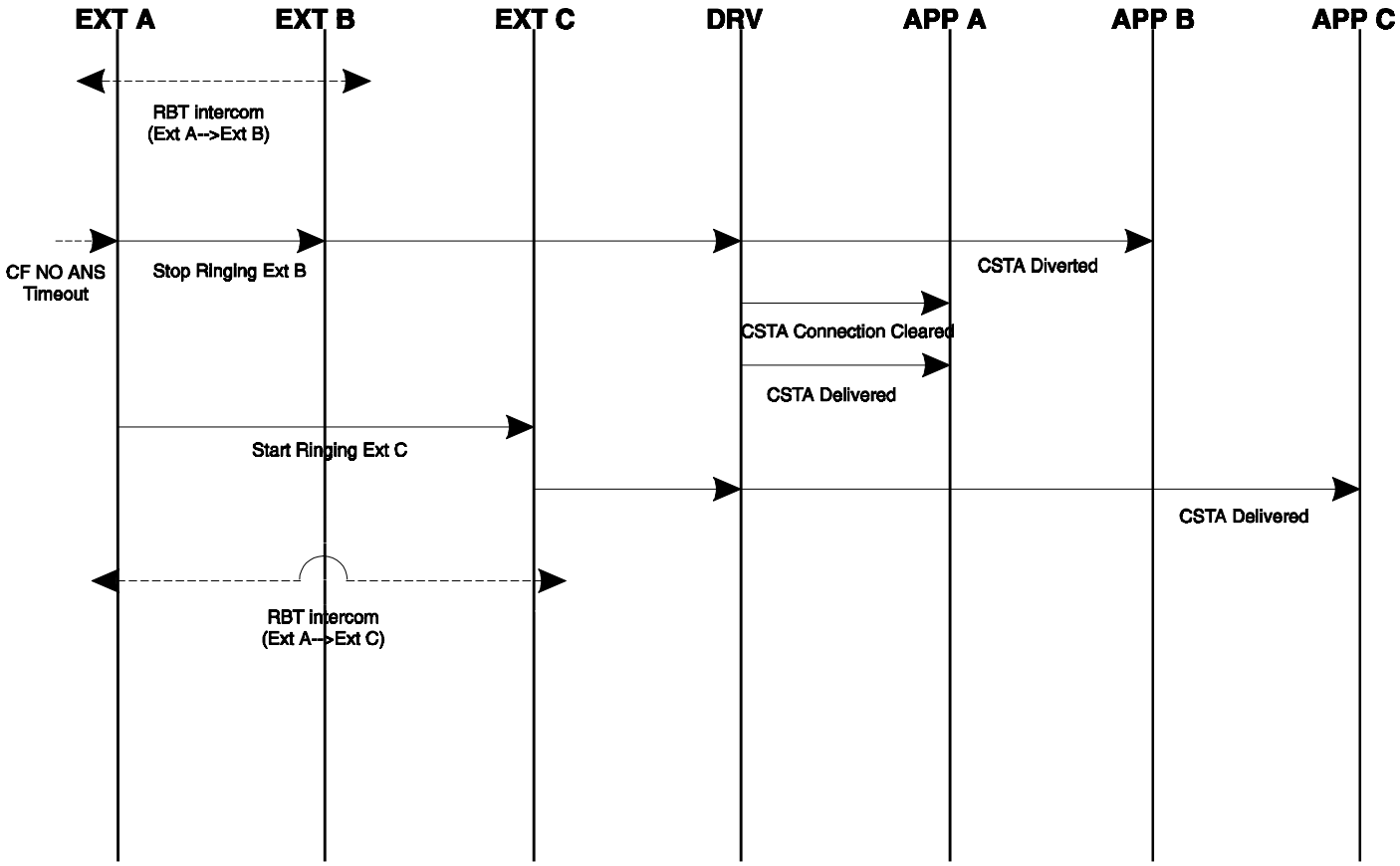
Busy Override



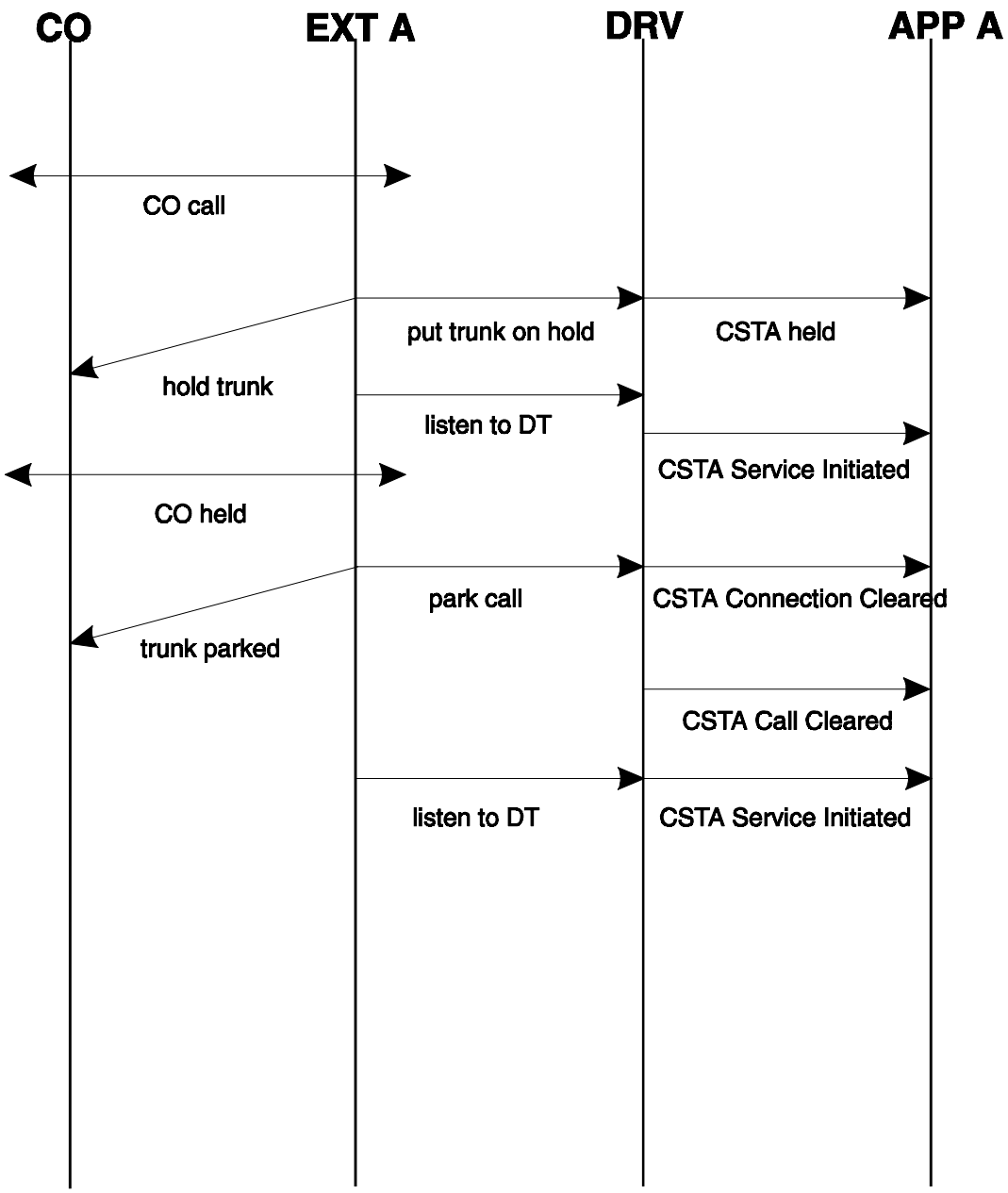
Call Forward - Busy & Immediate



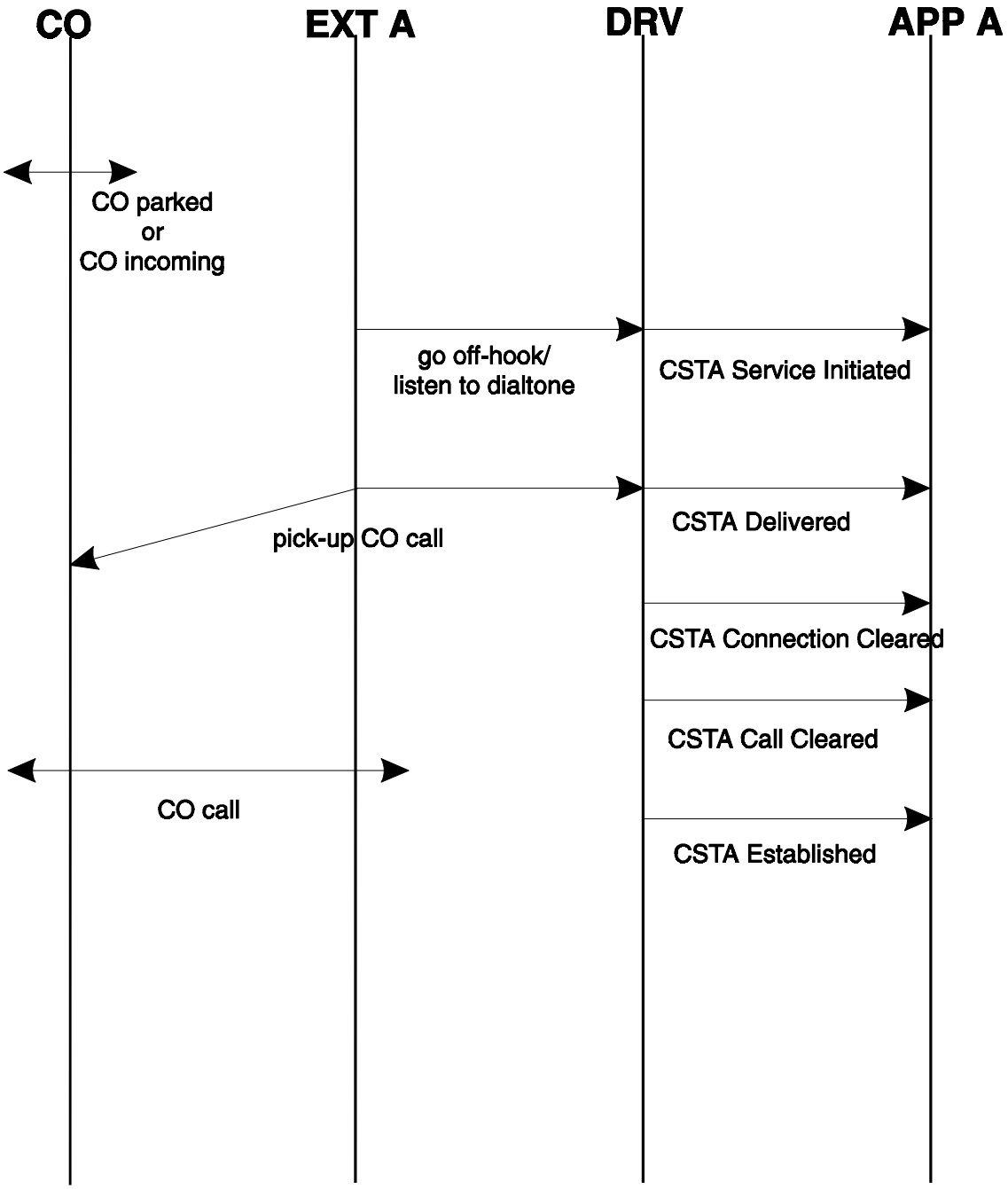
Call Forwarding - No Answer



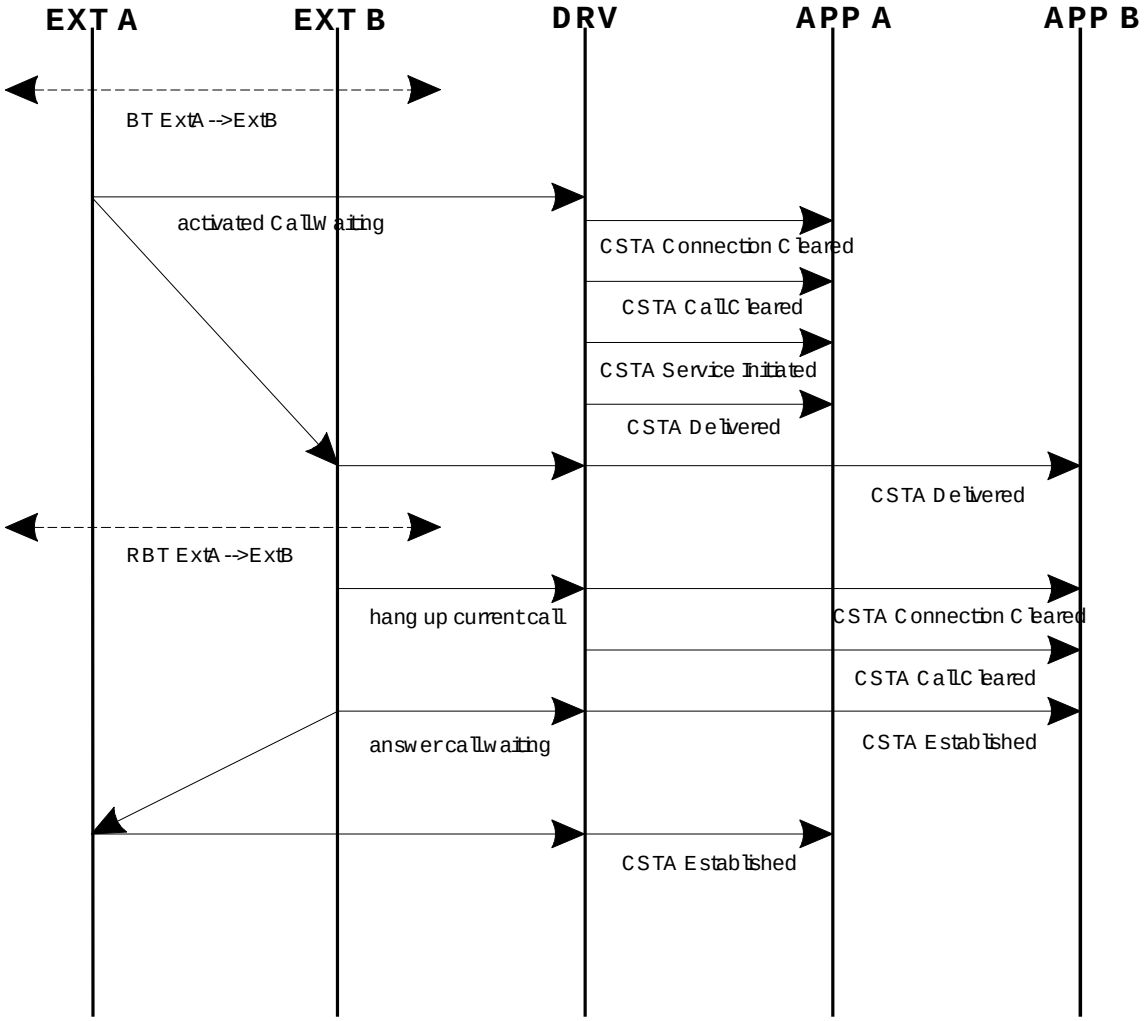
Call Park


CO
EXT A
DRV
APP A
CO call
put trunk on hold
CSTA held
hold trunk
listen to DT
CSTA Service Initiated
CO held
park call
CSTA Connection Cleared
trunk parked
CSTA Call Cleared
listen to DT
CSTA Service Initiated

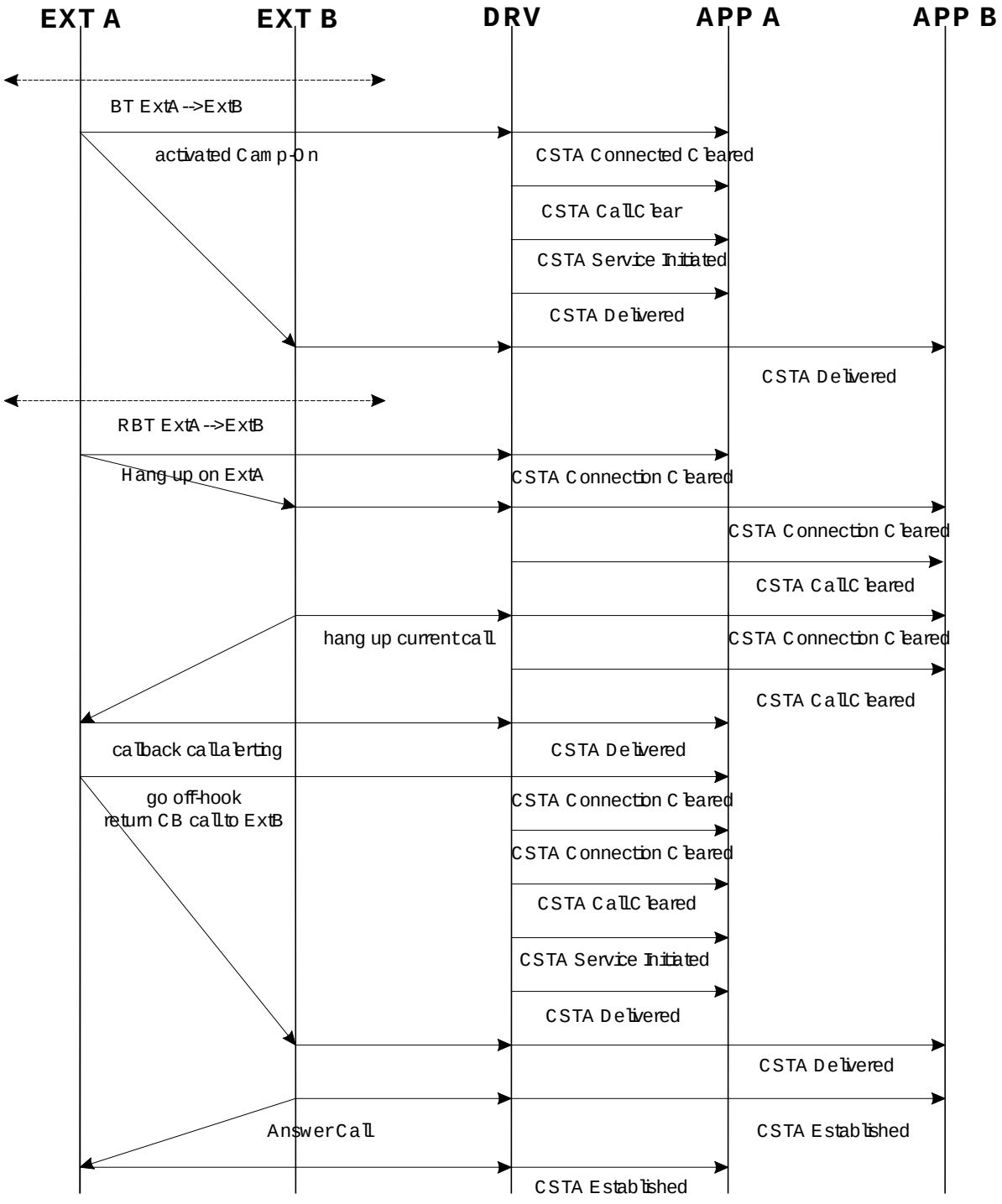
Call Pickup


CO
EXT A
DRV
APP A
CO parked
or
CO incoming
go off-hook/
listen to dialtone
CSTA Service Initiated
CSTA Delivered
pick-up CO call
CSTA Connection Cleared
CSTA Call Cleared
CO call
CSTA Established

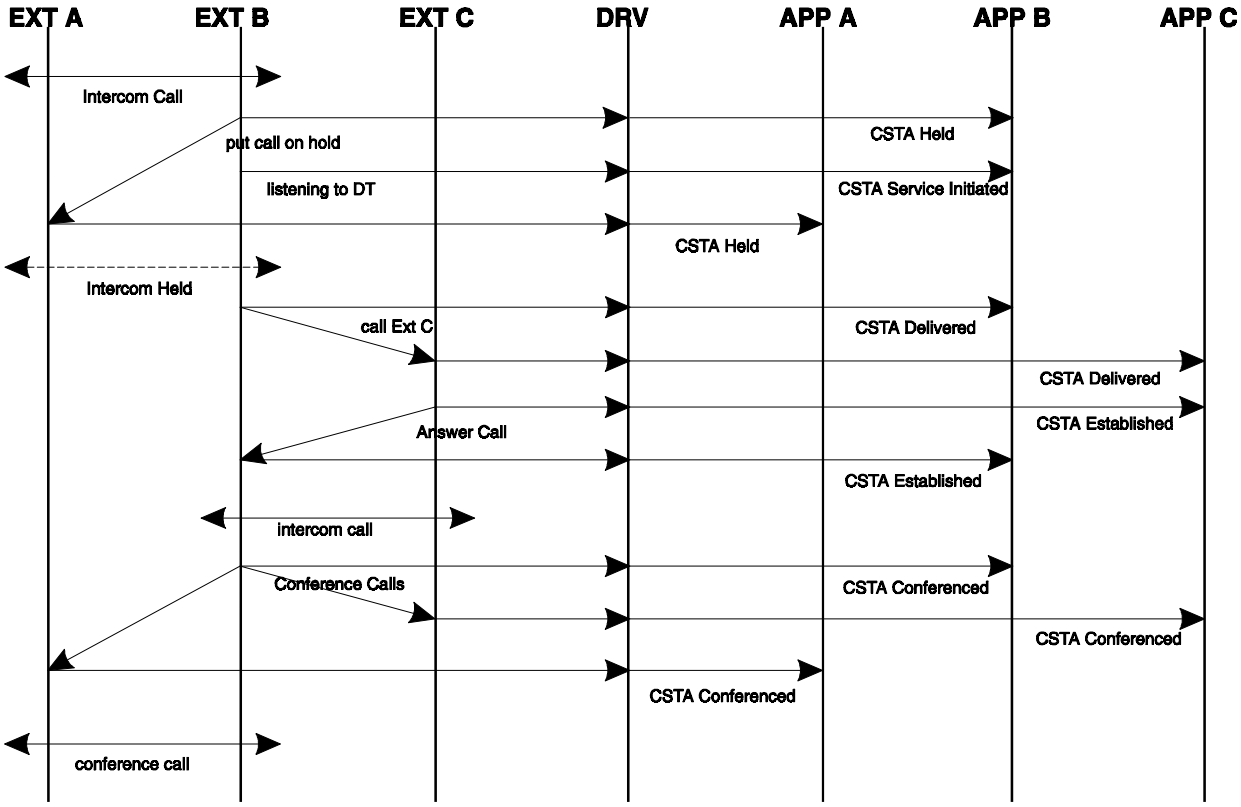
Call Waiting



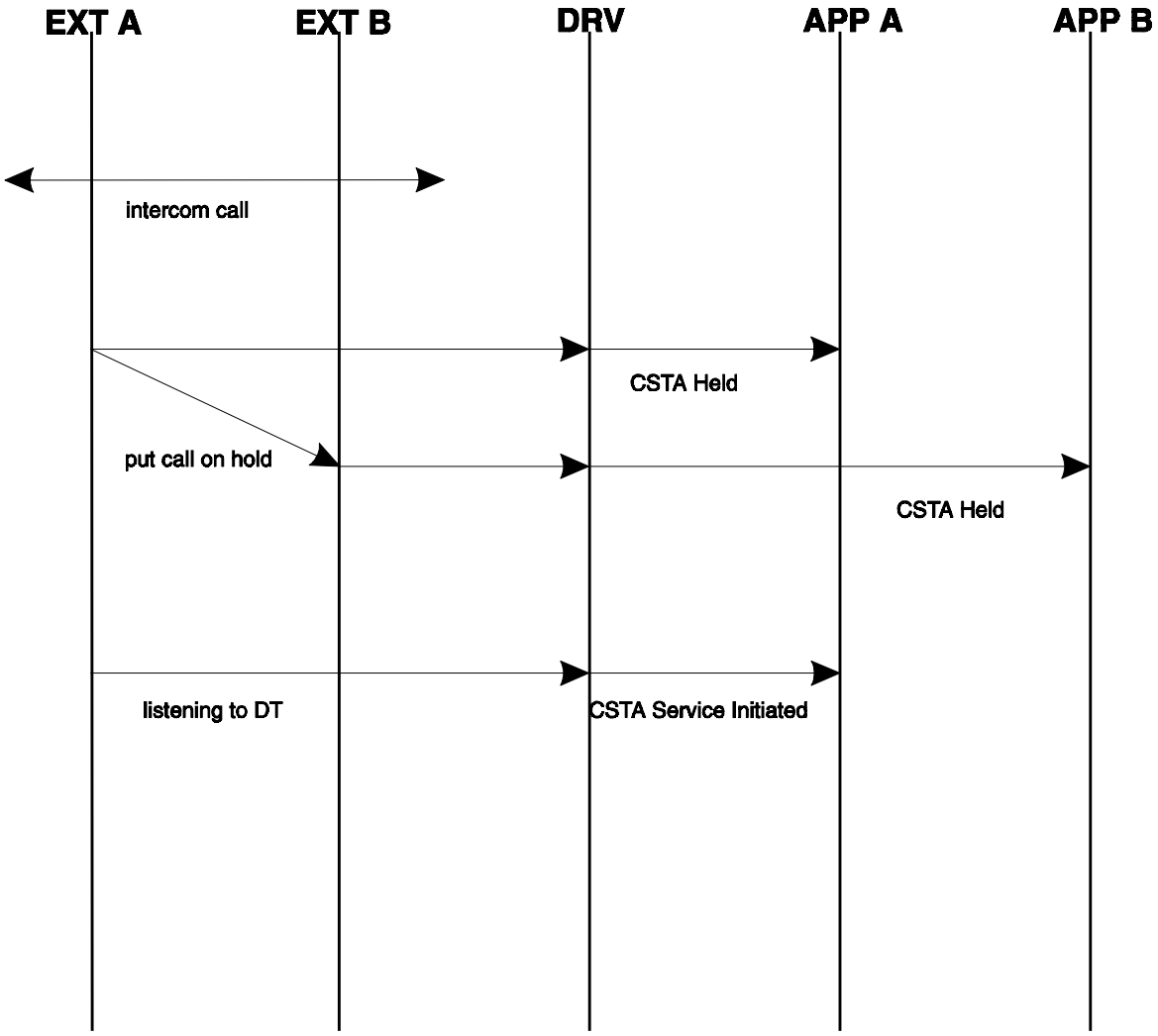
Camp-On


EXT A
EXT B
DRV
APP A
APP B
BT ExtA -->ExtB
activated Camp-On
CSTA Connected Cleared
CSTA CallClear
CSTA Service Initiated
CSTA Delivered
CSTA Delivered
RBT ExtA -->ExtB
Hang up on ExtA
CSTA Connection Cleared
CSTA Connection Cleared
CSTA CallCleared
hang up current call
CSTA Connection Cleared
CSTA CallCleared
callback call alerting
CSTA Delivered
go off-hook
return CB call to ExtB
CSTA Connection Cleared
CSTA Connection Cleared
CSTA CallCleared
CSTA Service Initiated
CSTA Delivered
CSTA Delivered
AnswerCall
CSTA Established
CSTA Established

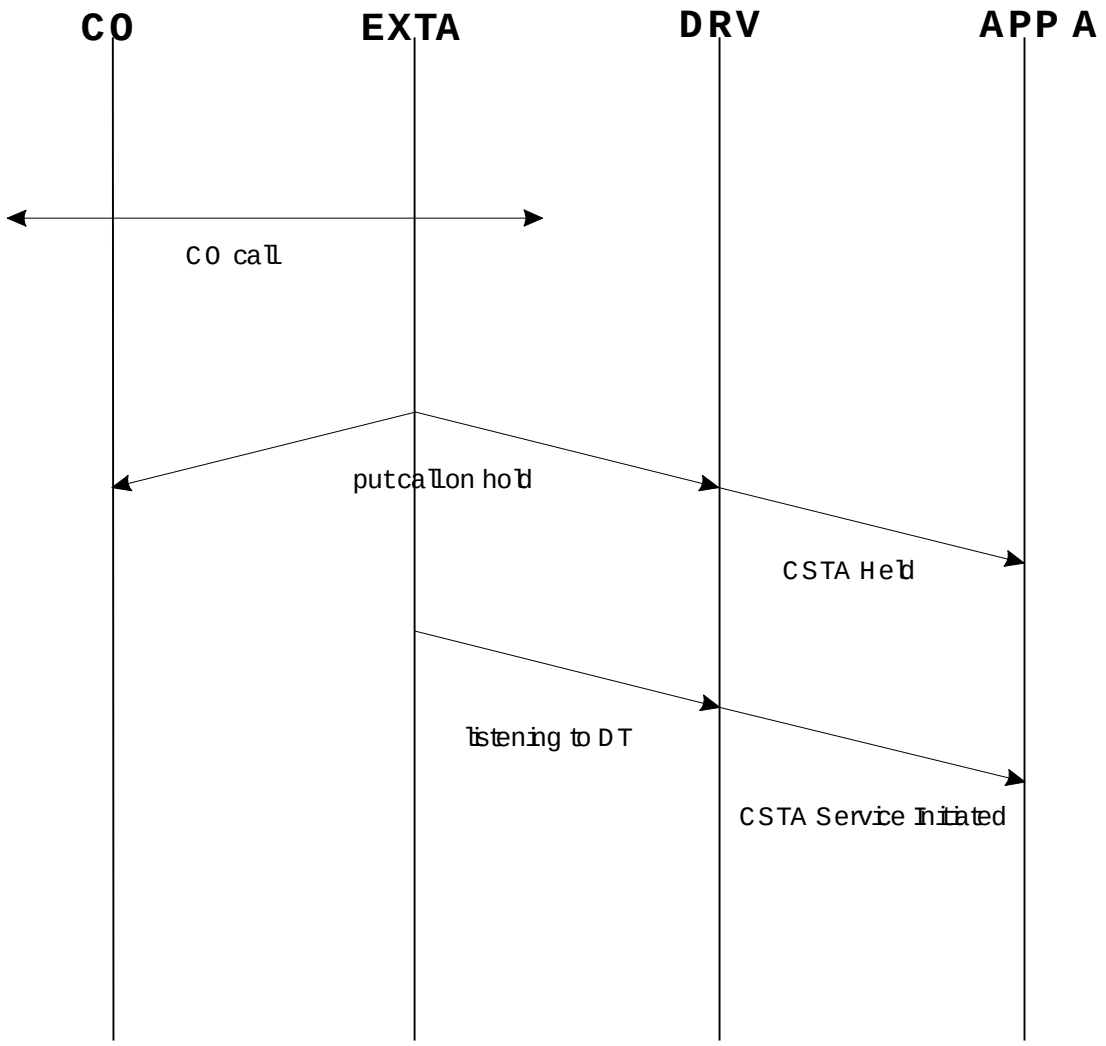
3-Way Conference



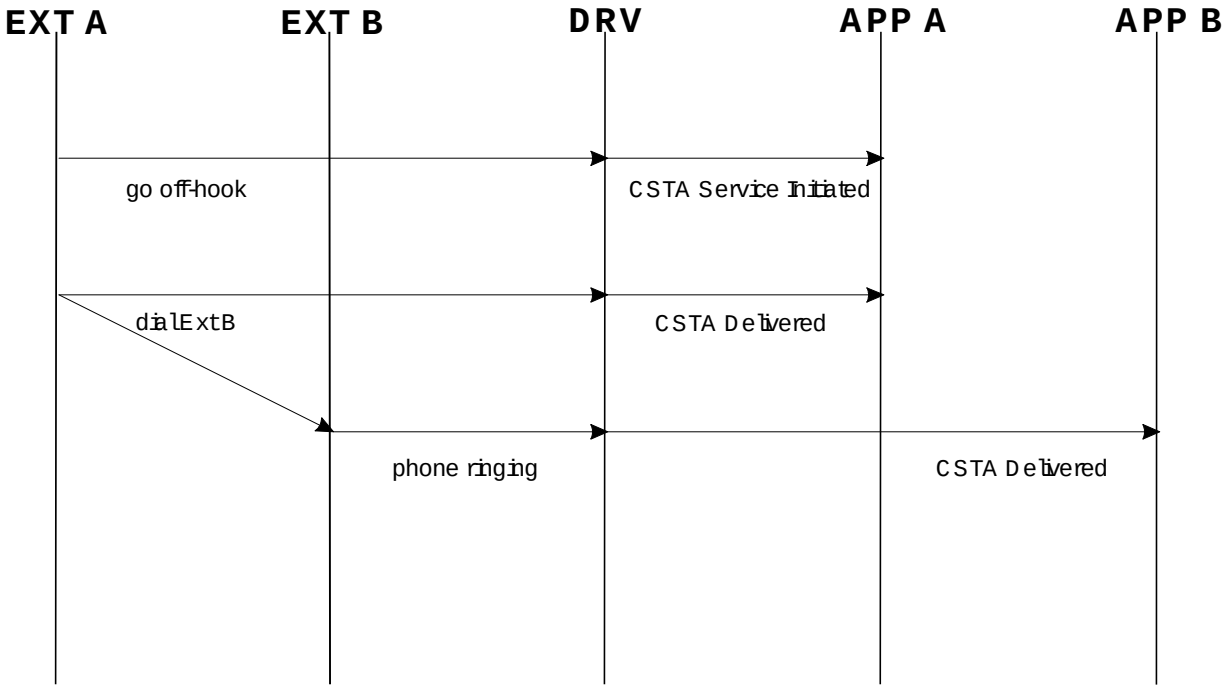
Hold Intercom



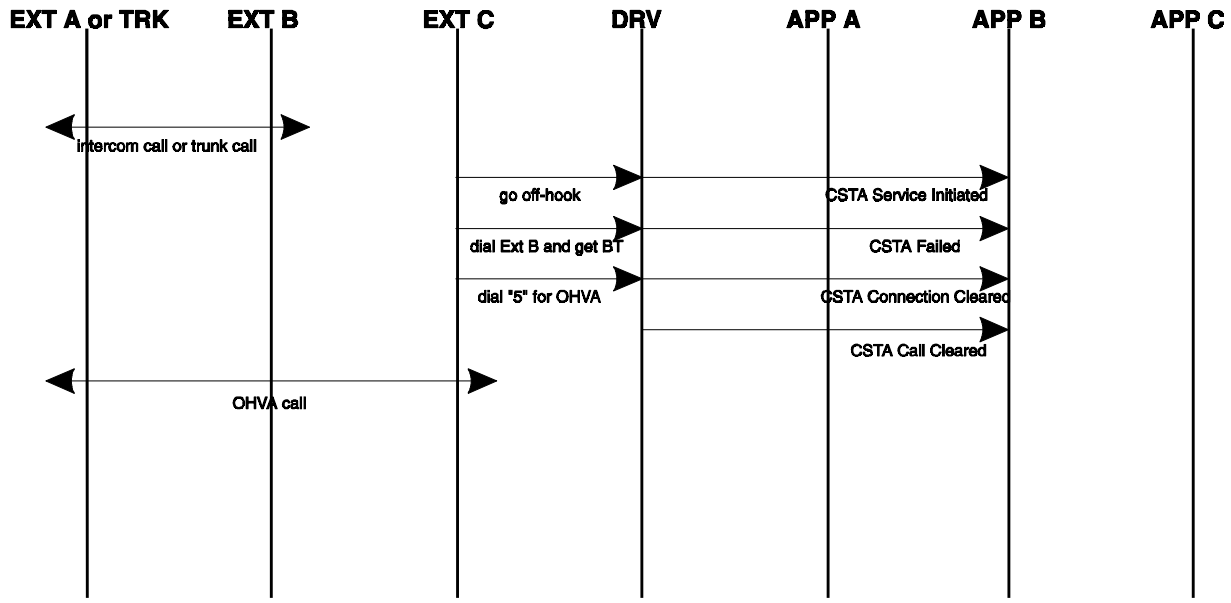
Hold CO


CO
EXTA
DRV
APP A
CO call
put call on hold
CSTA Held
listening to DT
CSTA Service Initiated

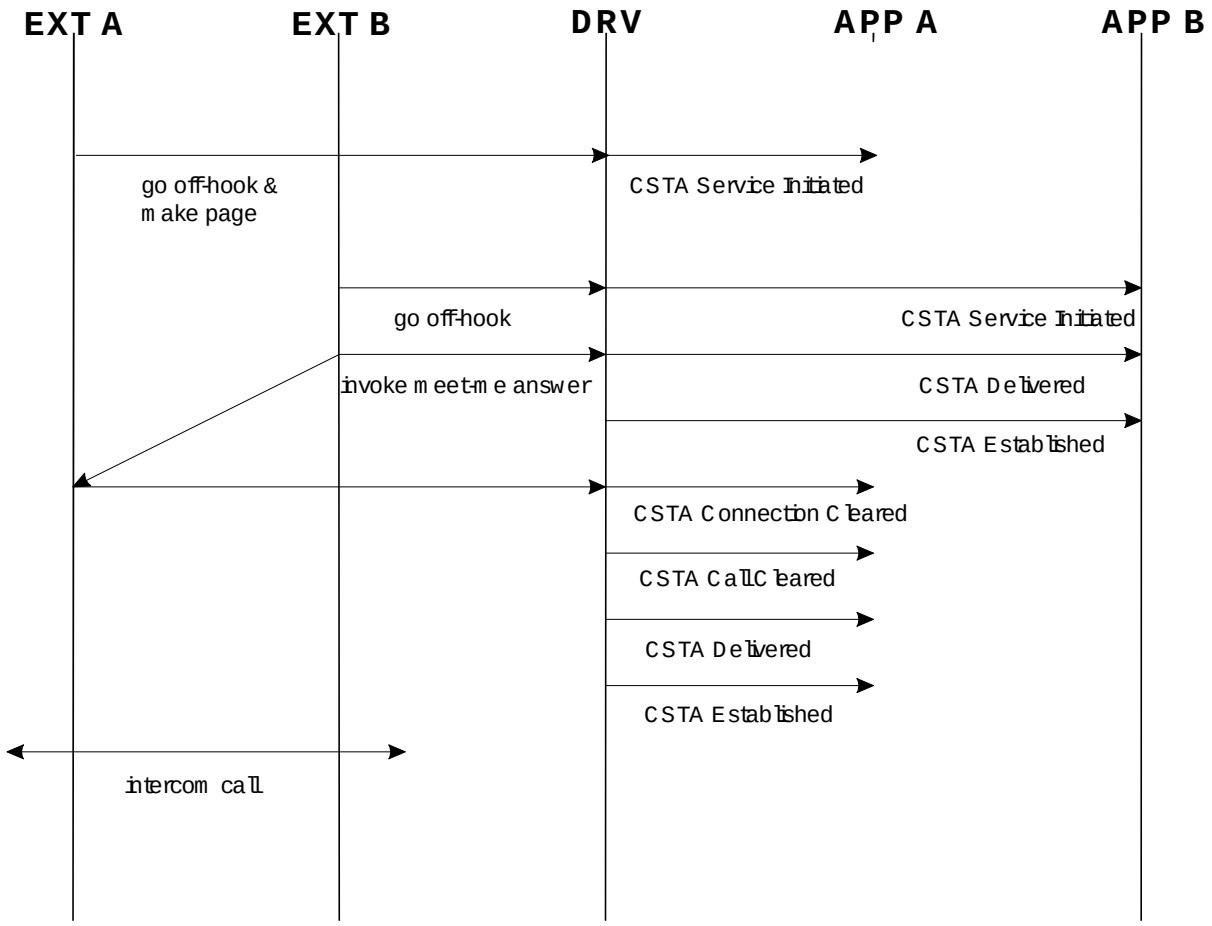
Intercom Call



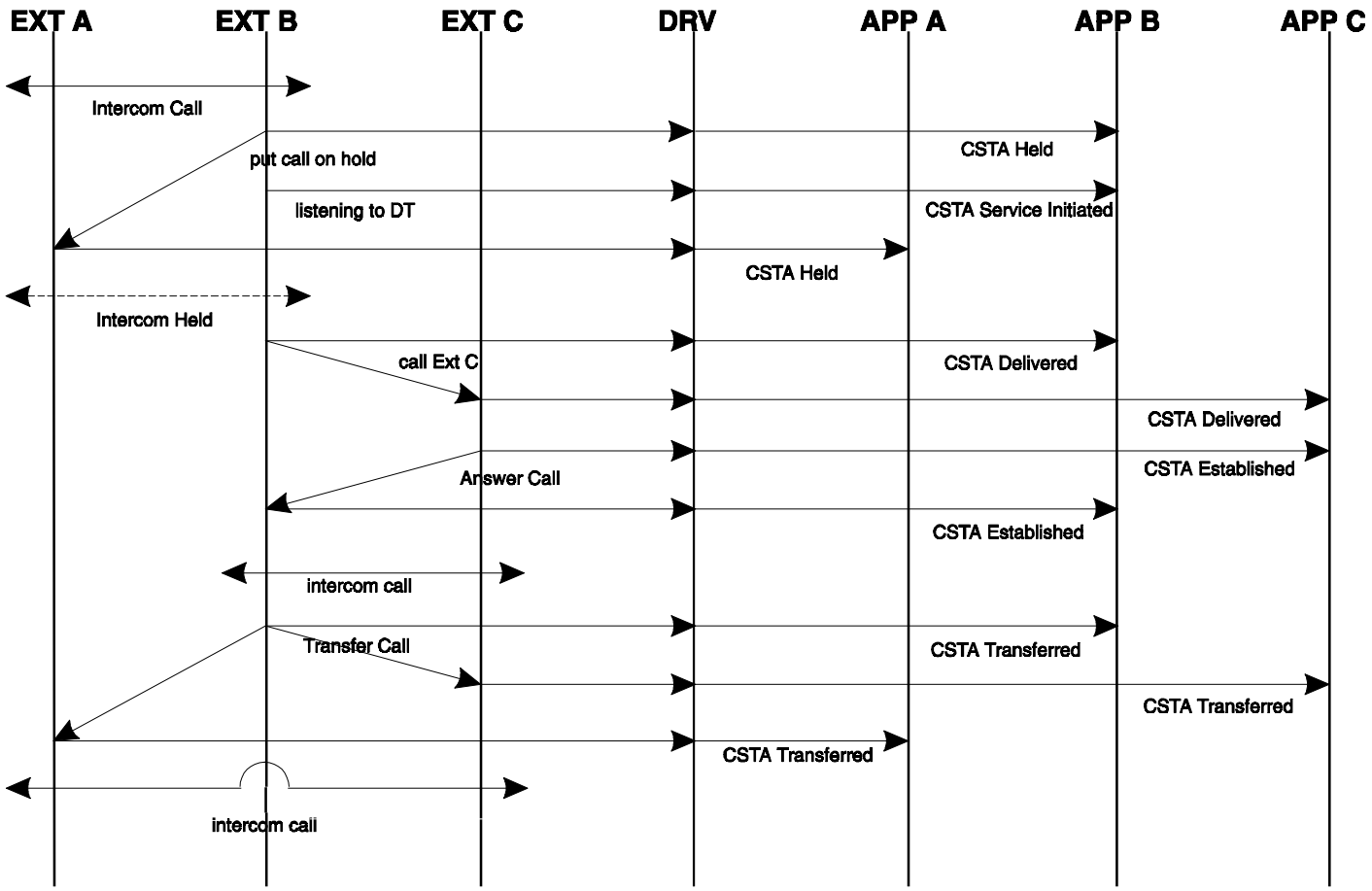
Off-Hook Voice Announce



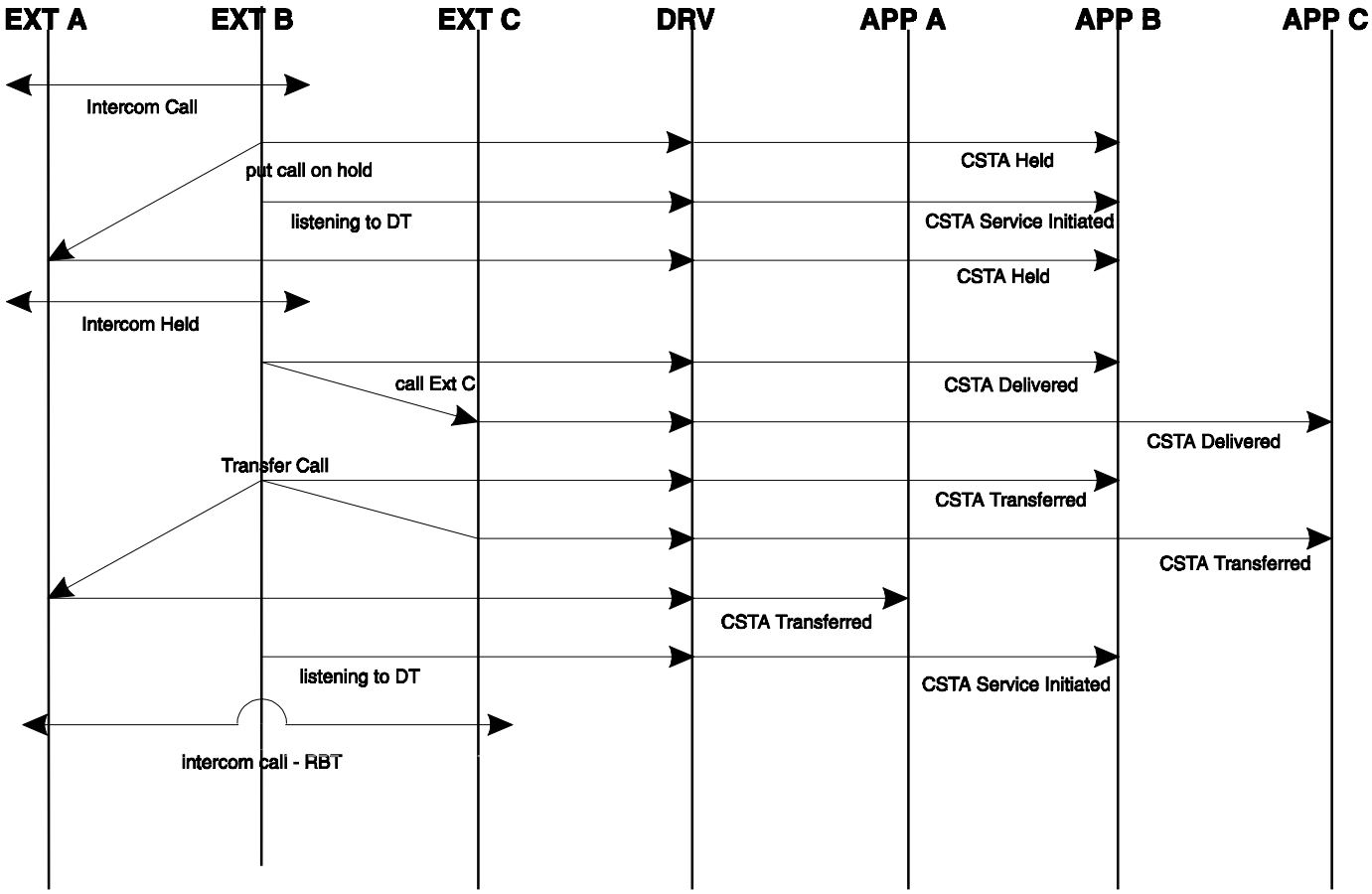
Paging/Meet Me Answer



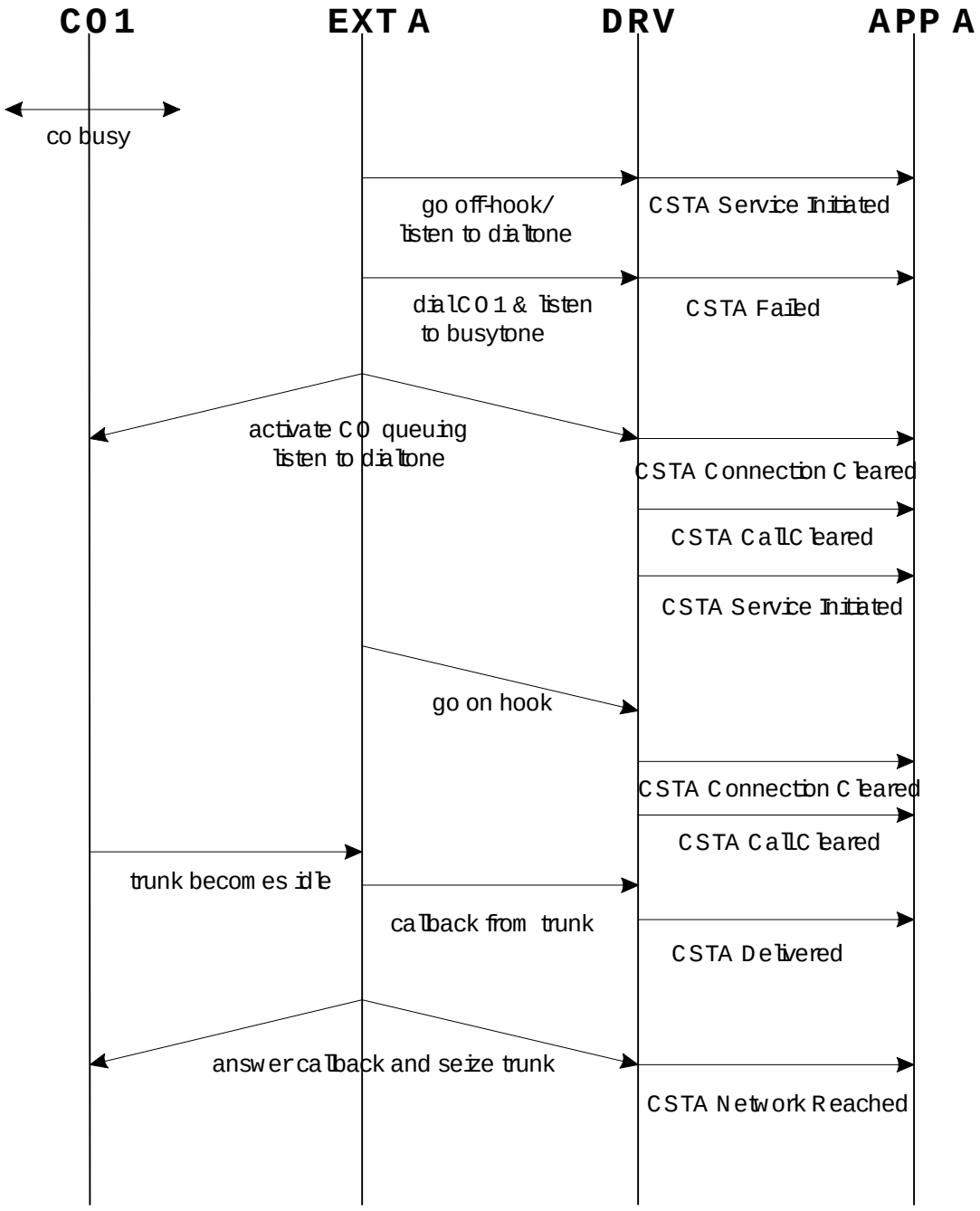
Transfer - Supervised



Transfer - Unsupervised



Trunk Queuing



Additional DBS Feature Handling

Absence Message

An extension calling another extension with an absence message set will receive a CSTAFailed event.

Account Codes

Non-verified account codes are supported via CSTAMakeCall request, using "A" for the auto key. Verified account codes are supported via CSTAMakeCall request.

Auto-Redial

The events will be the same as the original call.

Barge-in for Direct Line

Not supported.

Call Coverage

Not supported.

Caller ID

The Caller ID number, if available, is passed in the CSTADelivered Event and the CSTATransferred Event.

Caller ID Call Log

Sends no events to the TSAPI client.

Call Waiting/OHVA Text Reply

Call Waiting/Off-hook Voice Announce Text Reply cannot be activated from a TSAPI client.

CO Line Key Trunk Access

When the FF-Key is pressed the TSAPI client will receive a CSTAServiceInitiated Event followed by a CSTANetworkReached event.

Delayed Ringing

When the extension begins alerting, the same events are sent as if the extension had started alerting immediately.

Dial "0" for Attendant

The same events are sent as in the intercom call.

Dialtone Disabled

Doesn't affect the CSTA events for that extension. Any time when the dialtone would have been present, a CSTAServiceInitiated event will be sent.

Direct Trunk Access

A trunk can be directly accessed via CSTAMakeCall request, with the dialed digits containing "88XX".

DID

Incoming DID calls send CSTADelivered Events with the calling party the trunk number.

DISA

Events with DISA calls are not supported.

Do Not Disturb

The TSAPI client can set or clear DND via a CSTASetDoNotDisturb request. An extension that makes a call to an extension with DND set will receive a CSTAFailed event.

EM/24 Console

Any programmed keys supported on the K-Tel are also supported on the EM/24 console.

FF-Keys

All FF keys are supported.

Handsfree Answerback

Voice calls send the same events as a tone call. The caller and called party will receive CSTADelivered events. The call will stay in that state until either the caller hangs up or the called party answers the call, at which point the CSTAEstablished events are sent.

Handsfree Operation

Uses the speaker to answer/drop calls. The speaker is used by CSTAMakeCall, CSTAAnswerCall, and CSTARetrieveCall request. Therefore, a speakerphone is necessary for a TSAPI client.

Hot Dial Pad

Calls initiated via a hot dial pad will send the same events as an intercom/trunk call.

Internal Hold Tone

Internal hold tone, if administered, will be applied to a trunk put on hold via a CSTAHoldCall request.

Key Bank Hold

If administered, will operate the same as if the FF-keys were pushed and send the appropriate event: CSTAHeld or CSTAConnectionCleared followed by CSTACallCleared.

Last Number Redial

Sends the same events as the original call.

LCR

Least Cost Routing (LCR) will apply to a CSTAMakeCall request with the first dialed digit "9".

Line Appearances

The user can manually pick up any alerting call on a BLF appearance and the TSAPI client will be notified of the pickup. The client will not be sent CSTADelivered events when the call first starts alerting a coverage member's extension or a BLF key whether or not BLF ringing has been administered. All incoming calls on an ML or MCO appearance will send a CSTADelivered event to the client. Any non-appearance trunk or intercom calls will also send a CSTADelivered event to the client.

Music on Hold

Music on hold, if administered and available, will be applied to a trunk put on hold via a CSTAHoldCall request.

One Touch Keys

Any key that turns on the speaker will cause a CSTAServiceIninitiated event. Any other messages are dependent on the call progress.

One Touch VM Access

Upon pressing the Voice Mail Access key, a CSTAService Initiated Event will be sent, followed by a CSTADelivered Event.

Paging

A page can be made via CSTAMakeCall request with "#<pagegroup>" as the dialed digits.

Pooled Trunk Access

Pooled trunk access can be used via CSTAMakeCall request with the dialed digits "9" or "81-86".

Prime Line Preference

When the station user goes off-hook a CSTAServiceInitiated event will be sent followed by a CSTANetworkReached event. No trunk access codes are needed in the dialed digits for the CSTAMakeCall request.

Private Line

The same events are sent as for "regular" trunks, both incoming and outgoing calls.

Reminder Call

This has no affect on TSAPI clients.

Ringing Line Preference

The same events are sent as when the call is answered, via either CSTAAnswerCall request or handset off/speaker on.

Saved Number Redial

The same events are sent as the original call but the user must first sieze a trunk to use this feature.

SMDR

Any calls made via CSTAMakeCall request and any call answered via CSTAAnswerCall request will produce SMDR records when appropriate.

Speed Dials

Sends the same events as if the digits were dialed from the key pad or sent via a CSTAMakeCall request.

Station Class of Service

Class Of Service (COS) will be checked where appropriate.

Station Hunting

Any hunt group call received by a TSAPI client extension will receive CSTA events for the hunt group call.

Station Lockout

Station Lockout can be activated/deactivated via a CSTAMakeCall request.

T1 Trunks

Any incoming calls on a T1 trunk will send a CTSADelivered event with the calling party as the trunk number and any outgoing T1 trunk call will send a CSTANetworkReached event.

Trunk-to-Trunk Transfer

The same restrictions apply as for the extension transferring the call via the "prog" button on the telephone.

Voice Mail Transfer Key

Upon pressing the Voice Mail Transfer key, a CSTAHeld Event and a CSTAService Initiated Event will be sent. After dialing an extension or pressing a DSS key, a CSTATransferred Event will be sent.

UNA

If assigned to ring at a TSAPI client extension, the call can be answered via CSTAAnswerCall request. Also, at an extension where the UNA is not set to alert, dialing "78" via the telephone or as the dialed digits in a CSTAMakeCall request will pick up the call and the appropriate CSTA events will be sent to display the call.

Index

D

E

F

H

I

K

L

M

N