Programmer's Guide for MERLIN MAGIXÒ Integrated System and MERLIN LEGENDÒ Communications System PBX Driver

Issue 2.2 June 2002

Notice

Every effort was made to ensure that the information in this book was complete and accurate at the time of printing. However, information is subject to change.

Avaya Web Page

The world wide web home page for Avaya is http://www.avaya.com.

Heritage Statement

Intellectual property related to this product (including trademarks) and registered to Lucent Technologies Inc. has been transferred or licensed to Avaya. Any reference within the text to Lucent Technologies Inc. or Lucent should be interpreted as reference to Avaya. The exception is cross references to books published prior to April 1, 2001, which may retain their original Lucent titles. Avaya, formed as a result of Lucent's planned restructuring, designs, builds, and delivers voice, converged voice and data, customer-relationship management, messaging, multiservice networking, and structured cabling products and services. Avaya Labs is the research and development arm for the company.

Preventing Toll Fraud

Toll Fraud is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or working on your company's behalf). Be aware that there is a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.
Avaya Fraud Intervention
If you suspect that you are being victimized by toll fraud and you need technical assistance or support, call the Avaya Customer Care Center at
1 800 628-2888.

Providing Telecommunications Security

Telecommunications security of voice, data, and/or video communications is the prevention of any type of intrusion to, that is, either unauthorized or malicious access to or use of, your company's telecommunications equipment by some party.

Your company's “telecommunications equipment” includes both this Avaya product and any other voice/data/video equipment that could be accessed via this Avaya product (that is, “networked equipment”).

An “outside party” is anyone who is not a corporate employee, agent, subcontractor, or working on your company's behalf. Whereas, a “malicious party” is anyone, including someone who may be otherwise authorized, who accesses your telecommunications equipment with either malicious or mischievous intent.Such intrusions may be either to/through synchronous (time multiplexed and/or circuit-based) or asynchronous (character-, message-, or packet-based) equipment or interfaces for reasons of:
Utilization (of capabilities special to the accessed equipment)
Theft (such as, of intellectual property, financial assets, or toll-facility access)
Eavesdropping (privacy invasions to humans)
Mischief (troubling, but apparently innocuous, tampering)
Harm (such as harmful tampering, data loss or alteration, regardless of motive or intent).

Be aware that there may be a risk of unauthorized or malicious intrusions associated with your system and/or its networked equipment. Also realize that, if such an intrusion should occur, it could result in a

variety of losses to your company, including, but not limited to, human/data privacy, intellectual property, material assets, financial resources, labor costs, and/or legal costs.
Your Responsibility for Your Company's Telecommunications Security
The final responsibility for securing both this system and its networked equipment rests with you - an Avaya customer's system administrator, your telecommunications peers, and your managers. Base the fulfillment of your responsibility on acquired knowledge and resources from a variety of sources, including, but not limited to:
Installation documents
System administration documents
Security documents
Hardware-/software-based security tools
Shared information between you and your peers
Telecommunications security experts

To prevent intrusions to your telecommunications equipment, you and your peers should carefully program and configure your:
Avaya provided telecommunications system and their interfaces
Avaya provided software applications, as well as their underlying hardware/software platforms and interfaces
Any other equipment networked to your Avaya products

Federal Communications Commission Statement

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at their own expense. For further FCC information, see Appendix A, “Customer Support Information,” in Feature Reference.
Canadian Department of Communications (DOC) Interference Information
This digital apparatus does not exceed the Class A limits for radio noise emissions set out in the radio interference regulations of the Canadian Department of Communications.
Le Présent Appareil Numérique n’émet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de la classe A préscrites dans le réglement sur le brouillage radioélectrique édicté par leministère des Communications du Canada.

Trademarks

MERLIN, MERLIN LEGEND, and MERLIN MAGIX are registered trademarks of Avaya Inc.
Microsoft, Windows, Windows NT, and MS-DOS are registered trademarks of Microsoft Corporation.
UNIX is a registered trademark of UNIX System Laboratories, Inc.
Intel and Pentium are registered trademarks of Intel Corporation.
Apple and Macintosh are registered trademarks of Apple Computer, Inc.
IBM is a registered trademark of International Business Machines, Inc.
Novell and NetWare are registered trademarks of Novell Corporation.
All products and company names are trademarks or registered trademarks of their respective holders.

Support Telephone Number

In the continental U.S., Avaya provides a toll free customer helpline 24 hours a day. Call the Avaya Customer Care Center at 1 800 628-2888 or your Avaya authorized dealer if you need assistance when installing, programming, or using your system. Outside the continental U.S., contact your local Avaya authorized representative.

Contents

Contents

3 Control Services and Events

Contents

Contents

Contents

Contents

Figures

Figures

8 Call Events

Tables

2 MERLIN LEGEND/MERLIN MAGIX TSAPI Overview

3 Control Services and Events

Tables

4 Call Control Services

Tables

5 Supplementary Services

Tables

6 Monitoring

7 Snapshot Services

Tables

8 Call Events

Tables

Tables

Tables

Tables

12 Event Flows

Tables

About This Document

Contents

Contents

About This Document

Purpose and Scope

NOTE:
Computer Telephony Integration (CTI) is supported by Releases 5.0, 6.0, 6.1 and 7.0 of the MERLIN LEGEND® switch, and Releases 1.0, 1.5, 2.0, 2.1, and 2.2 of the MERLIN MAGIX® switch.

The MERLIN LEGEND PBX Driver operates with Releases 5.0, 6.0, 6.1 and 7.0 of the MERLIN LEGEND® switch, and Releases 1.0 and 1.5 of the MERLIN MAGIX switch. It will not operate with any other release of the MERLIN LEGEND or MERLIN MAGIX switches.

The MERLIN MAGIX PBX Driver operates with all MERLIN LEGEND and MERLIN MAGIX switch releases that support CTI.

This document provides application developers with detailed information about Computer-Supported Telecommunications Applications (CSTA) and the Telephony Services Application Programming Interface (TSAPI) for the MERLIN LEGEND Advanced Communications System and MERLIN MAGIX Integrated System PBX Driver. This programming interface is for use in a CentreVu® Telephony Services environment.

This document:

- lists the CSTA and TSAPI services and events that MERLIN LEGEND and MERLIN MAGIX CTI provide and explains the programming interface to each
- lists the service and event parameters that MERLIN LEGEND and MERLIN MAGIX CTI provide and details their semantics
- describes the service and event interactions with MERLIN LEGEND and MERLIN MAGIX switch features

- provides TSAPI syntax for programming
- explains the programming interface to MERLIN LEGEND and MERLIN MAGIX private data

Intended Audience

This document is for Telephony Services application developers who are programming applications for use with the MERLIN LEGEND Advanced Communications System or the MERLIN MAGIX Integrated System. Refer to *Telephony Services* in the "Terminology" section later in this chapter. This document assumes a familiarity with

- the CSTA model and services presented in Standard ECMA-217 Services for Computer-Supported Telecommunications Applications (CSTA)
- the programming interface in *Telephony Services Application Programming Interface (TSAPI)*
- MERLIN LEGEND switch features and operations described in *MERLIN LEGEND Communications System Feature Reference*
- MERLIN MAGIX switch features and operations described in *MERLIN MAGIX Integrated System Feature Reference*

Terminology

The definitions below describe some important terms. More detailed definitions appear in context as key concepts, functions, and services are fully described. In addition, there is a Glossary and List of Acronyms at the end of the document.

API Control Services (ACS)

An application uses ACS services (a subset of TSAPI) to open, close, and control a communication channel (known as a stream) to a Telephony Server. Once an application opens a stream, the application uses other TSAPI function calls on the stream to request CSTA services from the Telephony Server.

CentreVu Computer-Telephony

The name of the combined product that includes:

- CentreVu Telephony Services
- CentreVu CallVisor® PC

Only the CentreVu Telephony Services component of the product is relevant to the MERLIN LEGEND PBX Driver and MERLIN MAGIX PBX Driver.

CentreVu Telephony Services

An implementation of Telephony Services for Windows.

CentreVu Telephony Services supercedes PassageWay® Telephony Services for Windows NT. The MERLIN LEGEND Advanced Communications System PBX Driver and MERLIN MAGIX Integrated System PBX Driver are only certified to operate with CentreVu Telephony Services; they are not certified to operate with PassageWay Telephony Services for Windows NT.

Computer-Supported Telecommunications Applications (CSTA)

CSTA is a European Computer Manufacturers' Association (ECMA) standard that defines a standard set of Telephony Services, responses, and events. An example of a service is a request for a call to be made from one phone to another. An example of an event is a message that an incoming call is ringing a phone. The CSTA definitions form the foundation for CentreVu Telephony Services. Although CSTA provides standard service and event definitions, it does not provide an Application Programming Interface (API) definition. TSAPI provides the API for CentreVu Telephony Services.

Computer-Supported Telecommunications Applications (CSTA)

The connection between the Telephony server and the MERLIN LEGEND/MERLIN MAGIX system that allows Computer-Telephony Integration.

MERLIN LEGEND Computer Telephony Integration (CTI)

The ability to monitor and control call activity at MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) stations through telephony-enabled software applications. CTI capabilities for MERLIN LEGEND and MERLIN MAGIX are implemented through Telephony Services.
MERLIN LEGEND CTI does not support MERLIN MAGIX Release 2.0 or later releases of the MERLIN MAGIX switch. MERLIN MAGIX CTI has superceded MERLIN LEGEND CTI.

MERLIN MAGIX Computer Telephony Integration (CTI)

The ability to monitor and control call activity at MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX stations through telephony-enabled software applications. CTI capabilities for MERLIN LEGEND and MERLIN MAGIX are implemented through Telephony Services.

A number of MERLIN MAGIX CTI capabilities are release dependent. Access to these capabilities requires that the system is equipped with the appropriate releases of the MERLIN MAGIX switch software and the MERLIN MAGIX PBX Driver. For example, access to MERLIN MAGIX Release 2.0 CTI capabilities requires that the system is equipped with Release 2.0 (or later) of the MERLIN MAGIX switch software and Release 2.0 (or later) of the MERLIN MAGIX PBX Driver. Access to Release 2.1 CTI capabilities requires that the system is equipped with Release 2.1 (or later) of the MERLIN MAGIX switch software and Release 2.1 (or later) of the MERLIN MAGIX PBX Driver.

MERLIN LEGEND PBX Driver (MLPD)

The MERLIN LEGEND PBX driver is a set of software modules on a Telephony Server that interfaces switch-independent Telephony Server software to the MERLIN LEGEND Advanced Communications System or MERLIN MAGIX (Releases 1.0 and 1.5) Integrated System. This software terminates and manages the MERLIN LEGEND or MERLIN MAGIX CTI link. The MERLIN LEGEND PBX Driver does not support Release 2.0 or later of the MERLIN MAGIX switch. The MERLIN MAGIX PBX Driver has superceded the MERLIN LEGEND PBX Driver.

MERLIN MAGIX PBX Driver (MMPD)

The MERLIN MAGIX PBX driver is a set of software modules on a Telephony Server that interfaces switch-independent Telephony Server software to the MERLIN LEGEND Advanced Communications System or MERLIN MAGIX Integrated System. This software terminates and manages the MERLIN LEGEND or MERLIN MAGIX CTI link.

PassageWay Telephony Services

Implementations of Telephony Services on NetWare® (PassageWay Telephony Services for NetWare) and Windows NT (PassageWay Telephony Services for Windows NT).

Implementations of the MERLIN LEGEND PBX Driver are available for both the NetWare and Windows NT environments; an implementation of the MERLIN MAGIX PBX Driver is available for Windows NT only. The MERLIN LEGEND Advanced Communications System Windows NT Driver and the MERLIN MAGIX Integrated System PBX Driver are only certified to operate with CentreVu Telephony Services; they are not certified to operate with PassageWay Telephony Services for Windows NT.

Private Data

Private Data is a TSAPI mechanism that allows a PBX vendor to enhance TSAPI services and events and even provide new services within the TSAPI framework. The MERLIN LEGEND PBX Driver and MERLIN MAGIX PBX Driver use private data to provide value-added features:

- Both the MERLIN LEGEND and MERLIN MAGIX switches pass any call prompting digits that have been collected for a call in certain events for the call.
- When an application uses the ***cstaConsultationCall( )*** to extend a call to another desktop, the MERLIN LEGEND and MERLIN MAGIX PBX Drivers pass information about the original caller (and prompted digits) in private data so that an application monitoring the receiving extension can pop a screen using information about the original caller as soon as the consultation call begins to alert at the receiving desktop.
- Beginning with MERLIN MAGIX Release 2.0, the switch includes information for account codes as well as trunk identifiers for incoming calls.

- Beginning with MERLIN MAGIX Release 2.0 the switch offers several escape services for applications to obtain information about administered DGC groups, administered labels and trunk status information. Additional escape services have been added in MERLIN MAGIX Release 2.1.

When MERLIN LEGEND or MERLIN MAGIX private data is provided within a TSAPI event, the private data appears in the ***privateData*** parameter. This document defines a C structure that overlays the ***privateData*** parameter and gives programmers access to MERLIN LEGEND or MERLIN MAGIX private data.

Telephony Services Application Programming Interface (TSAPI)

TSAPI is the C programming language interface to CentreVu Telephony Services. Application programmers use TSAPI to access CSTA services, responses, and events. TSAPI is switch independent and supports many CentreVu Telephony Services-compliant drivers, including the MERLIN LEGEND Advanced Communications System Windows NT Driver and MERLIN MAGIX Integrated System PBX Driver.

Telephony Server

A Telephony Server is a server on a local area network that provides CentreVu Telephony Services to client applications. The Telephony Server has a CTI link to a MERLIN LEGEND Advanced Communications System or MERLIN MAGIX Integrated System. A client application makes TSAPI requests of the Telephony Server. The Telephony Server passes these requests to the MERLIN LEGEND Advanced Communications System Windows NT Driver or MERLIN MAGIX Integrated System PBX Driver, which, in turn, passes them over the CTI link to the MERLIN LEGEND or MERLIN MAGIX switch. The MERLIN LEGEND or MERLIN MAGIX switch processes these requests and returns responses and call events through the Telephony Server to the requesting application.

NOTE:
The term *Telephony Server* is also commonly used to refer to the Telephony Services software running on the Telephony Server machine.

Telephony Services

A technology providing server-based telephony control for client (desktop) or server applications on an enterprise network.

Telephony Services implementations include:

- PassageWay Telephony Services for NetWare
- PassageWay Telephony Services for Windows NT
- CentreVu Telephony Services

No support for MERLIN LEGEND or MERLIN MAGIX CTI is available with PassageWay Telephony Services for NetWare or PassageWay Telephony Services for Windows NT.

Related Documents

Following is a list of documents related to the MERLIN LEGEND Advanced Communications System, MERLIN MAGIX Integrated System, CSTA, TSAPI, and Telephony Services. A description follows each document name describing the role of the document.

Telephony Services Application Programming Interface (TSAPI), Version 2

This document:

- Defines the TSAPI programming interface, an Application Programming Interface for CSTA Services and Events;
- Provides a tutorial on the CSTA client/server operational model.

TSAPI provides a programming environment that may be used with any switch for which there is a Telephony Services Driver (such as the MERLIN LEGEND or MERLIN MAGIX PBX Driver). The TSAPI specification is required reading for a Telephony Services application developer.

Standard ECMA-217 Services for Computer-Supported Telecommunications Applications (CSTA), European Computer Manufacturers' Association, December 1994

The above standard reflects agreements of ECMA member companies on a set of Telephony Services and Events. This document contains the CSTA model and service and event definitions. The CSTA standard is optional reading for an application developer.

MERLIN LEGEND Advanced Communications System Release 7.0 Feature Reference, 555-770-110

The above document provides a comprehensive description of MERLIN LEGEND Advanced Communications System features. It is an important reference for the planning, operation, and administration of MERLIN LEGEND CTI application development. It is recommended reading for a MERLIN LEGEND CTI application developer.

This document is provided with the MERLIN LEGEND switch hardware. Additional copies are available at the Fulfillment Center.

MERLIN MAGIX® Integrated System Feature Reference – Release 2.2 and Earlier, 555-722-110

This document provides detailed information about how the MERLIN MAGIX Release 2.2 or earlier switch and telephone features operate. It is an important reference for the planning, operation, and administration of MERLIN MAGIX CTI application development. It is recommended reading for a MERLIN MAGIX CTI application developer.

This document is provided with the MERLIN MAGIX switch hardware. Additional copies are available at the Fulfillment Center.

Network Manager's Guide for MERLIN LEGEND® Advanced Communications System

The above document describes the hardware, software, and configuration requirements for the MERLIN LEGEND Advanced Communications System PBX Driver. It also provides information about the installation, administration, maintenance, and troubleshooting of the MERLIN LEGEND PBX Driver. It is required reading for a MERLIN LEGEND CTI system administrator.

This document is provided on the CD-ROM for the MERLIN LEGEND CTI product.

Network Manager's Guide for MERLIN MAGIX® Integrated System PBX Driver

The above document describes the hardware, software, and configuration requirements for the MERLIN MAGIX Integrated System PBX Driver. It also provides information about the installation, administration, maintenance, and troubleshooting of the MERLIN MAGIX PBX Driver. It is required reading for a MERLIN MAGIX CTI system administrator.

This document is provided on the CD-ROM for the MERLIN MAGIX CTI product.

CentreVu® Computer-Telephony - Telephony Services and CallVisor® PC Installation

The above document provides detailed information for LAN administrators and service technicians on how to install, load, and run CentreVu Telephony Services for Windows and CallVisor PC products. It is required reading for LAN administrators and service technicians.

This document is provided on the CD-ROM for the CentreVu Computer-Telephony product.

CentreVu® Computer-Telephony - Telephony Services Administration and Maintenance

The above document describes the computer hardware, software, and configuration requirements for CentreVu Telephony Services for Windows, as well as information about the administration of the Security Database. The Security Database validates application requests against user privileges. The Security Database permits an application executing on behalf of a user to monitor and control only specified devices. It is required reading for a CentreVu Telephony Services administrator.

This document is provided on the CD-ROM for the CentreVu Computer-Telephony product.

TSAPI Model

1

Contents

Contents

TSAPI Model

1

This chapter contains an introduction to the TSAPI programming model. TSAPI is based on the European Computer Manufacturers' Association (ECMA) standard for Computer Supported Telecommunications Applications (CSTA). This chapter gives a summary explanation of the model. For a complete discussion, refer to the *Telephony Services API* Manual. Readers that are well versed in the TSAPI programming model may skip this chapter.

Definitions

Active Call

The call (at an extension) that is connected (in a talking state) at that extension. The Connection (see *Connection*) for the Active Call is in the Connected State (see the definition of *Connection State* in "TSAPI Connection Object" later in this chapter).

Alerting Call

A call that is either visually or audibly alerting at a Device. The Connection (see *Connection*) for an Alerting Call is in the Alerting State. When the Device is a telephone, the Alerting Call is ringing the telephone instrument.

Call (TSAPI programming object)

A Call is a communications relationship between two or more Devices. Note, however, during call set-up and release, and at other times during a call, there may be only one Device on the Call.

Call Identifier

A TSAPI programming handle that identifies a Call.

Connection (TSAPI programming object)

A relationship between a Call and a Device. A Connection is in one of a number of states (alerting, held, connected, etc.). Note that when a Call connects (for example) three Devices, there are three Connections for the Call. Each Connection reflects the state of the Call at one of the Devices.

Connection Identifier (TSAPI programming handle)

A TSAPI programming handle that identifies a Connection. A Call Identifier and a Device Identifier comprise a TSAPI Connection Identifier.

Device (TSAPI programming object)

A device is an Object, which abstracts the interface between a user and communications in the Switch. TSAPI allows a device to be a single endpoint (such as a telephone), or multiple endpoints that form a group. Chapter 2 details the subset of the TSAPI Devices that the MERLIN LEGEND and MERLIN MAGIX switches support.

Device Identifier (TSAPI programming handle)

A TSAPI programming handle that identifies a Device.

Event

A message from a Switch to a Computer indicating that an occurrence of interest to an Application (that typically has caused a change in the state of a Connection) has occurred.

Held Call

A call (at an extension) that is held (in a hold state) at that extension. The Connection (see *Connection*) for a Held Call is in the Hold State (see the definition of *Connection State* in “TSAPI Connection Object” later in this chapter).

Object

TSAPI programming objects include Connections, Calls, and Devices. Each has a corresponding programming handle, or identifier.

State

An object's current condition. Specifically, TSAPI Connections have an associated state.

Client/Server Model

TSAPI uses a client/server architecture to make telephony services available to applications software. The application runs on a computer and typically plays the role of a client, making requests of a server, which, in turn, relays these requests on to a switch. Such requests often include monitoring extensions and controlling connections.

TSAPI Services are independent of the specific CTI link connecting the switch with the application. Since TSAPI is independent of the particular telephone terminal types, the Switch must determine how to support a given TSAPI request for its specific telephone types. For example, TSAPI does not specify how to provide the Make Call Service for analog or ISDN telephones. A Switch will use its existing service definitions to provide TSAPI Services on telephones where that service already exists.

The Switch provides event reports, which allow a service requester to assess the progress of its service requests.

TSAPI Programming Objects

The TSAPI model defines several Switching Sub-Domain Model Objects for use in Application programming: Device, Call, and Connection.

TSAPI Device Object

The TSAPI model permits an application to monitor and control Devices of various types (including telephones). In CSTA, a Device can refer to either a physical device (such as buttons, lines, trunks, and stations) or a logical device (such as groups of devices, and ACDs). Chapter 2 details the subset of these Devices that the MERLIN LEGEND and MERLIN MAGIX switches support.

Devices have associated attributes, which allow applications to monitor and control them.

TSAPI device attributes are:

- **Device Type** - the MERLIN LEGEND and MERLIN MAGIX switches support the following TSAPI Device Types (refer to the *Telephony Services Application Programming Interface (TSAPI) Version 2* for a complete list):
  - Station - is the traditional telephone device.
  - Trunk - a device that spans switches (or interfaces a switch to the public network).

- Calling Group Queue This is a holding place for calls that are being directed to one member of a group of stations when all the stations are unavailable to receive the calls. TSAPI support for Calling Group Queues was introduced in MERLIN MAGIX Release 1.5.

- **Device Class** - An application may monitor or control TSAPI Devices in the various Device Classes in different ways. TSAPI defines several classes. MERLIN LEGEND and MERLIN MAGIX CTI support class **Voice**, a device that is used to make audio calls.
- **Device Identifier** - a TSAPI programming handle for a Device that allows an application to uniquely identify each device. TSAPI identifies Devices using static and/or dynamic identifiers:
  - Static Device Identifier - A Static Device Identifier is stable over time. It remains constant and unique as calls arrive at (and leave) the device. A Static Device Identifier is typically the extension number for the Device.
  - Dynamic Device Identifier - the MERLIN LEGEND and MERLIN MAGIX CTI do not use dynamic device identifiers.
- **Device State** - is a list of the Connection States for all the connections at the Device. For information about Connection states, see “TSAPI Connection Object” later in this chapter.

TSAPI Call Object

TSAPI applications can monitor and control calls (including call establishment and release). In certain operations, such as conference and transfer, one Device in a Call is replaced with another Device or two Calls are merged into a single Call. In these situations, the TSAPI Call object is maintained as long as the communications relationship remains across each operation (i.e. the call survives transfer, conference, and forwarding operations). TSAPI Call object attributes are:

- **Call Identifier** - a Call Identifier is a TSAPI programming handle that the Switch assigns to each Call. The Call ID may or may not be unique among all calls within a Switch, but coupled with a Device ID, the pair will form a unique Connection ID within a Switching Sub-Domain. To allow reference to a nascent call, the switch will assign a Call ID before a call is fully established. For example, a switch will assign a Call ID to an incoming call when the called Device is Alerting (the assignment is done before the call is answered).

  Certain Services merge multiple calls into a single call. Examples of such TSAPI Services are Transfer and Conference. During operations of Services that merge multiple calls, the call identifier may change, but the call continues as a TSAPI object. The management of the call identifier is described in “Identifier Management” later in this chapter
- **Call State** - is a list of the Connection states for all the Connections that are a part of the Call.

TSAPI Connection Object

A Connection is a relationship between a Call and a Device. A TSAPI application can control a Connection. For example, the TSAPI Services Hold Call, RetrieveCall, and Clear Call all control Connections. Connections are TSAPI programming objects with the following attributes:

ion Identifier - is a TSAPI handle that is made up of a Call Identifier and Device Identifier. For a call, there are as many Connection identifiers as there are associated devices. Similarly, for a device, there are as many Connection identifiers as there are associated calls. The Connection Identifier is unique within a Switch and within a single TSAPI server. A TSAPI application cannot use a Connection Identifier until it has received the identifier from the Switch.

- **Connection State** - is the state of a call at a Device. The Connection state always refers to a single Call/Device relationship. Monitors report Events, which are changes in Connection States for the monitored device.

TSAPI Connection State Model

Figure 1-1 shows a sample Connection state model. Note that since TSAPI is switch-independent, and since switch features vary from switch to switch (and therefore interact differently on different switches), there is no definitive TSAPI Connection State model to which all switches comply.

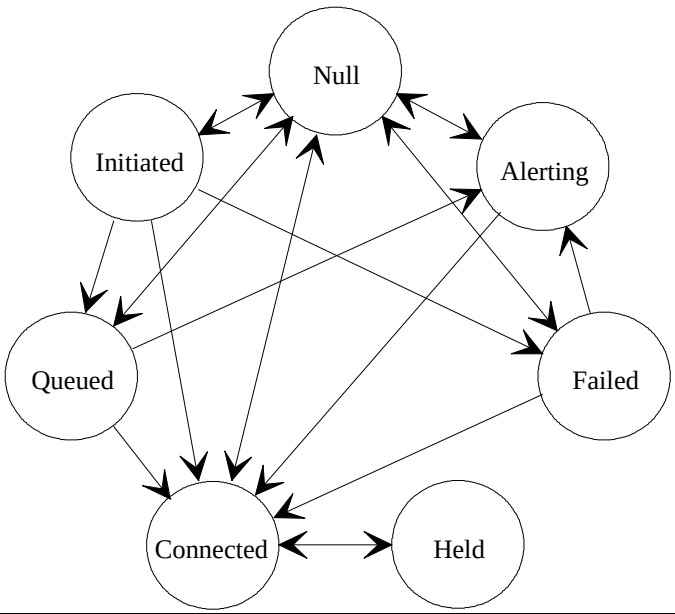


Figure 1-1. Sample Connection State Model

The transitions between states, shown by arrows, form the basis for providing Event Reports.

The TSAPI Connection states are defined as follows.

- **Null** - the state where there is no relationship between the call and device.
- **Initiated** - the state where the device is requesting service. Usually this results in the creation of a call. Often this is thought of as the "dialing" state.
- **Alerting** - the state where a device is visually or audibly alerting.
- **Connected** - the state where a device is connected to the communications channel for a call (but is not a held call).
- **Held** - the call is "on hold" at the Device.
- **Queued** - the state where normal state progression has been stalled. For example, a call being processed by an ACD that is waiting for an ACD agent to become available is "queued."
- **Failed** - the state where normal state progression has been aborted. A "Failed" state can result because of failure to connect to the calling (originator) device, failure to connect the called (destination) device, failure to create the call, and other reasons.

NOTE:
The MERLIN LEGEND and MERLIN MAGIX switches have additional states, such as the associative states and held-for-transfer and held-for-conference that the TSAPI model does not reflect. Be sure to read and understand the treatment that TSAPI applications will see for these MERLIN LEGEND and MERLIN MAGIX connection states.

A call can be modeled as a collection of Connection state machines. Signaling causes changes in the connection state machines.

Certain operations involve changes to many Connections. TSAPI reports these events (such as Transfer and Conference) in a single Event Report. Each TSAPI Event Report defines which Connections have changed state.

Identifier Management

The Switching Function provides Connection Identifiers when either a new Call or Device Identifier is needed. When a call is made the switch provides a Connection Identifier. The switch then provides the Connection ID in any following Event Reports that pertain to that call. Similarly, the switch provides Connection IDs containing a Device ID for a device involved in a call.

The switch updates identifiers when needed. If a Conference or Transfer (merging two calls) changes a Call ID, then the switch provides Event Reports containing Connection IDs that link the old call identifier to the new identifier. Both Service Acknowledgments and Event Reports may contain information necessary to manage identifiers.

Identifiers cease to be valid when their context vanishes. If a call ends, its call identifier is no longer valid. Many Event Reports and Services specify when a Connection Identifier has lost or will lose its context.

Identifiers can be reused. Once an identifier has lost its context it may be re-used to identify another object. Most implementations will not reuse identifiers immediately.

In the TSAPI model, Call and Device Identifiers can be, but are not guaranteed to be, globally unique. The TSAPI model ensures that connections (the combination of Call and Device Identifier) are globally unique within a Switch. In the MERLIN LEGEND and MERLIN MAGIX CTI implementations, both Call and Device Identifiers are unique within the switch, but an application that makes use of this fact will not be programmed in a switch-independent manner.

MERLIN LEGEND/MERLIN MAGIX TSAPI Overview

2

Contents

Contents

MERLIN LEGEND/MERLIN MAGIX TSAPI Overview

2

Introduction

Telephony Services integrates telephony monitoring and control with software applications on a local area network. The Telephony Server integrates the existing telephones on users' desktops with telephony-based or telephony-enabled applications. Applications developers use the Telephony Services Applications Programming Interface (TSAPI) to program these applications. These applications can either reside on the server (where they are referred to as server-based applications) or on desktop PCs (where they are referred to as client-based applications).

Telephony Services is a distributed client/server application environment that logically integrates the telephone on a user's desk with an application running on his or her computer (see Figure 2-1.) The system accomplishes this integration without the need for special telephones, PC circuit boards, or wiring at the user's desktop.

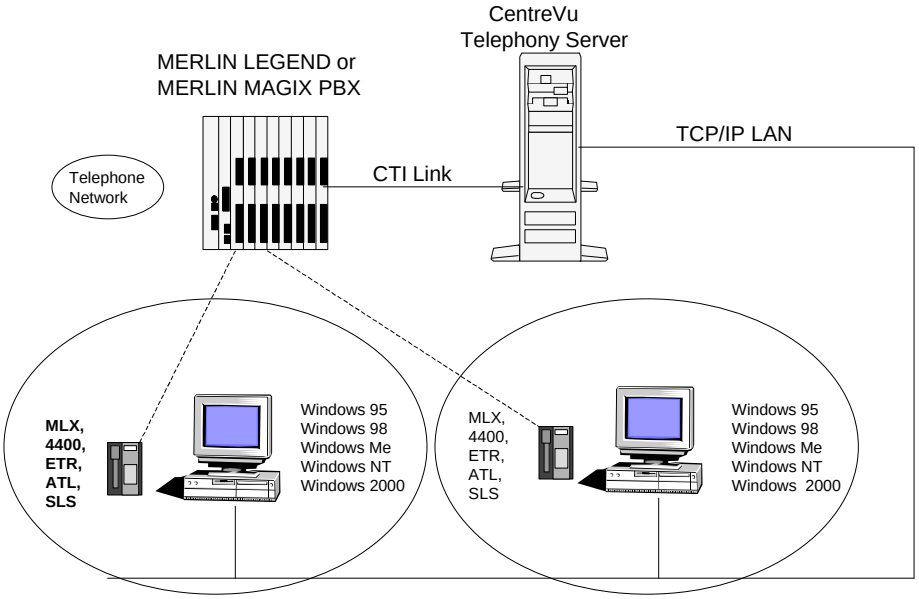


Figure 2-1. MERLIN LEGEND/MERLIN MAGIX CentreVu Telephony Services Configuration

A CTI link connects the MERLIN LEGEND or MERLIN MAGIX switch to the telephony server. Applications use the TSAPI library to pass telephony requests to the server, which, in turn, passes these requests across the CTI link to the switch.

Applications

The following types of applications might take advantage of TSAPI services to operate in a MERLIN LEGEND or MERLIN MAGIX CTI environment:

- Call Center or Customer Service Center
- Call Logging
- Call Management
- Call Screening
- Conference Management
- Custom Call Distribution

- Telemarketing Agent Management
- Preview Outbound Dialing Campaigns
- Screen pop using caller-ID, called number (DNIS), or information collected by voice prompter (collected digits[1])
- Boss/Secretary
- Call Tracking, Reporting
- Directory
- Telephony Enabling Non-CTI applications (using middleware)

While many applications might be implemented as either client or server applications, certain factors may result in one approach being more satisfactory, economical, or more manageable than the other. Consider the following factors:

- **Round-the-clock application availability**: Implementing an application on the server could eliminate the problem of a key application being inaccessible when a desktop PC is turned off.
- **Backup**: System administrators (or some automatic procedure) typically back up servers on a regular basis while individual users determine when to back up desktop PCs.
- **Scarce or expensive resources**: Such resources can be centralized, especially when light usage makes their inclusion in the client PC impractical or costly.
- **Single instance running on behalf of multiple users**: Certain applications (such as tracking and billing) typically run on behalf of many users. A single application may be more manageable than many copies of a single application, particularly when the data must be combined for reports.
- **Centralized, shared data**: Data files on a server are available to many users and more manageable than many copies of a database.

MERLIN LEGEND and MERLIN MAGIX CTI support TSAPI applications in conjunction with certain types of extension devices. The sections “Extension Types” and “Button Types” later in this chapter provide further information about supported devices. Appendix A provides a table showing CTI support for the various MERLIN LEGEND and MERLIN MAGIX station types.

1 Some industry publications refer to collected digits as “prompted digits.”

MERLIN LEGEND and MERLIN MAGIX Release 1.0

In the MERLIN LEGEND or MERLIN MAGIX Release 1.0 environment, applications may find the following CTI features to be especially useful:

- "power dial" from a System Access (SA) button - An application originates calls from a specified extension to an external or intercom number. A user might identify a name or number in an application and use a mouse to "point and click." The application then requests the switch to originate the call.
- "screen pop" for an incoming voice call to an SA button - An application uses calling number (Automatic Number Identification (ANI) and Individual Call Line Identification (ICLID) for external calls[2]), called number (Dialed Number Identification Service (DNIS) for external calls[3]), or collected digits to pop a screen for an alerting call. MERLIN LEGEND CTI provides events that support application screen pop either when a call alerts, or when it is answered. Events support application screen pop for calls that arrive through:
  - DGC distribution
  - ISDN PRI "routing by dial plan"
  - Direct Inward Dialing (DID)
  - transfer after answer at a voice response unit or VMI port; or
  - transfer after answer at an unmonitored DLC or QCC[4].
- Call control on SA buttons - An application can answer, hold, or retrieve a call; clear a connection; make a call (including consultation); transfer or conference a consultation call; transfer a call on hold for transfer; or conference a call on hold for conference (or transfer[5]). When an application makes a consultation call, an application monitoring the extension receiving the call receives the original caller's ANI/ICLID/extension information, DNIS, and original collected digits. Table 2-1 details the availability of control and monitoring for the different types of SA buttons. Applications cannot control calls on Shared SA buttons (SSA buttons.)

---

2 External calls must arrive on PRI/BRI facilities provisioned to provide ANI or loop start trunks that provide ICLID.

3 Called number is the local called extension in the case of a local caller or DNIS in the case of an external caller. DNIS in the MERLIN LEGEND or MERLIN MAGIX switch is the group extension number.

4 Non-CTI operation is used to answer and transfer the calls at the DLC/QCC, announcement units, and voice response units.

5 As the manual pages describing the services will show in more detail, an application can use the conference service to conference a call that is on hold for either transfer or conference.

A MERLIN LEGEND private data library provides collected digits in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent***. Refer to "MERLIN LEGEND and MERLIN MAGIX Private Data Libraries" later in this chapter for more information on the private data libraries.

MERLIN MAGIX Release 1.5

In a MERLIN MAGIX Release 1.5 environment, applications have access to all of the functionality provided by MERLIN LEGEND and MERLIN MAGIX Release 1.0, plus the following:

- Queue information –An application can monitor a Calling Group (split) to receive information about calls entering and leaving the queue.
- Agent information – An application monitoring a Calling Group member (agent) will be notified when the agent logs in, logs out, or enters After Call Work, and may also set the state of the agent.

MERLIN MAGIX Release 2.0

In a MERLIN MAGIX Release 2.0 environment, applications have access to all of the functionality provided by MERLIN MAGIX Release 1.5, plus the following:

- Queue information – An application can obtain a list of the administered Calling Groups, a list of administered Group Members (agents) within a Calling Group, and a list of lines and trunks assigned to a Calling Group. In addition, the application can query the status of the Calling Group queue to determine the number of queued calls.
- Agent control – An application can query or set an agent's status (i.e., Logged In, Logged Out, or Work Not Ready).
- Call control on Coverage, Line and Pool buttons – An application can answer, hold, or retrieve a call; clear a connection; make a consultation call; transfer or conference a consultation call; transfer a call on hold for transfer; or conference a call on hold for conference (or transfer[5]). An application can not originate a call on these buttons. When an application makes a consultation call, an application monitoring the extension receiving the call receives the original caller's ANI/ICLID/extension information, DNIS, and original collected digits. Table 2-2 details the availability of control and monitoring for the different types of buttons. Applications cannot control calls on Shared SA buttons (SSA buttons.)
- Tip/Ring (Single Line Set) control – An application can monitor a Single Line set and can hold or retrieve a call, clear a call, transfer or conference a consultation call; transfer a call on hold for transfer; or conference a call on hold for conference (or transfer). A Single Line set will not have access to the ***cstaAnswerCall( )*** or ***cstaMakeCall( )*** services.
- Deflect call capability – An application can redirect an unanswered calling group call to a specific agent, or to another queue.

- Account code information – An application can receive Account code information when an external call is cleared.
- Feature events – An application can receive notification when a user has activated or deactivated the Do Not Disturb feature.

MERLIN MAGIX Release 2.1

In a MERLIN MAGIX Release 2.1 environment, applications have access to all of the functionality provided by MERLIN MAGIX Release 2.0, plus the following:

- Supplementary Services – An application can set or query the status of the Do Not Disturb feature or a station's Message Waiting Indicator.
- Snapshot Device Service – An application can take a "snapshot" of calls appearing at an extension.
- Enhanced Agent Control – An application can set or query the status of a calling group agent with respect to a specific calling group.
- Enhanced Call Deflection – An application can redirect an unanswered calling group call to any extension that is available to receive the call.
- Additional Escape Services – An application can use new escape services to obtain additional switch configuration data.

A MERLIN MAGIX private data library provides collected digits in the ***CSTADeliveredEvent, CSTAEstablishedEvent*** and ***CSTAQueuedEvent***, and account codes in the ***CSTAConnectionClearedEvent***. Refer to the section "MERLIN LEGEND and MERLIN MAGIX Private Data Libraries" later in this chapter for more information.

Switch Environment

MERLIN LEGEND/MERLIN MAGIX CTI is available on MERLIN LEGEND switches (Release 5.0 and later) and MERLIN MAGIX switches operating in Hybrid/PBX Mode. (MERLIN LEGEND/MERLIN MAGIX CTI is not available on MERLIN LEGEND and MERLIN MAGIX switches in Key Mode, nor on MERLIN LEGEND and MERLIN MAGIX switches in "behind the switch" mode). MERLIN LEGEND/MERLIN MAGIX CTI is available only on the domestic version.

The physical CTI link to the MERLIN LEGEND or MERLIN MAGIX switches is MLX extension wiring (4 pair category 3). This requires an ISDN BRI interface card in the server and an MLX extension port on the MERLIN LEGEND or MERLIN MAGIX switch. No device other than the CTI link can be connected to the MLX CTI port on the switch.

Extension Types on MERLIN LEGEND

The MERLIN LEGEND switch will monitor and control SA buttons on a variety of MLX, ETR and ATL extension sets equipped with built-in speakerphone (BIS.) The sets must be directly connected to the MERLIN LEGEND switch. The MERLIN LEGEND switch does not monitor and control BRI (7500) or SLS sets. An application cannot monitor/control calls at other types of voice facilities (such as QCCs, trunks, MLX adjuncts, etc.)

Although the MERLIN LEGEND switch restricts the extension types that an application can monitor or control, call events provide information about any type of telephone that connects to a monitored device.

Appendix A provides a list of the supported extension types for MERLIN LEGEND.

Extension Types on MERLIN MAGIX

The MERLIN MAGIX switch will monitor and control buttons on a variety of MLX, 4400-series, and ETR extension sets equipped with built-in speakerphone (BIS.) Beginning with MERLIN MAGIX Release 2.0, single line sets (SLSs) may also be monitored and controlled. The sets must be directly connected to the MERLIN MAGIX switch. The MERLIN MAGIX switch does not monitor and control BRI (7500) sets. An application cannot monitor/control calls at other types of voice facilities (such as QCCs, trunks, MLX adjuncts, etc.)

Appendix A provides a list of the supported extension types for MERLIN MAGIX.

Normal, Responding Mode

A telephone must be in “Normal, Responding Mode” (see Glossary) for the MERLIN MAGIX switch to successfully complete a call control request at that telephone. A telephone need not be in “Normal, Responding Mode” for an application to successfully monitor that telephone.

Button Types on MERLIN LEGEND and MERLIN MAGIX

The MERLIN LEGEND (Releases 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches provide monitoring and call control for calls appearing on non-shared SA buttons. These switch releases do not provide call control on any other type of button.

MERLIN MAGIX Release 2.0 and later provide monitoring and call control for calls appearing on non-shared SA buttons, and also provides monitoring and call control for calls appearing on Coverage, Line, and Pool buttons. Tables 2-1 and 2-2 show CTI monitoring and control capabilities.

Table 2-1. MERLIN LEGEND (Releases 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) CTI Control and Monitoring for Button Types

Button Type	CTI Application can control a call on this type button?	CTI application receives events for call activity at this type button?
SA-Ring	yes	yes
SA-Voice	yes	yes
SA-Originate-Only-Ring	yes	yes
SA-Originate-Only-Voice	yes	yes
Shared SA	no	partial[6]
DFT	no	partial[6]
Pool	no	partial[6]
Cover	no	partial[6]
Loop	no	no

Table 2-2. MERLIN MAGIX Releases 2.0 and later CTI Control and Monitoring for Button Types

Button Type	CTI Application can control a call on this type button?	CTI application receives events for call activity at this type button?
SA-Ring	yes	yes
SA-Voice	yes	yes
SA-Originate-Only-Ring	yes	yes
SA-Originate-Only-Voice	yes	yes
Shared SA	no	partial[6]
DFT	yes[7]	yes
Pool	yes[7]	yes
Cover	yes[7]	yes
Loop	no	no

6 Does not supply ***CSTADeliveredEvent***.

7 These button types can not use the cstaMakeCallService( )

NOTE:
An application will receive events only for calls present at the button types shown above.

There are a variety of ways that an incoming CO call might appear at an SA button on an extension and provide events to a monitoring application:

- the extension is a DGC group member
- the call arrived on a PRI trunk with Routing By Dial Plan administered
- the CO call was transferred from another extension
- the call is an arriving remote access call
- DID Routing

LAN & Computing Environment

MERLIN LEGEND/MERLIN MAGIX CTI will operate with CentreVu Telephony Services for Windows, Release 3.1 and later. For more information about Telephony Services implementations, refer to *Telephony Services* in the "Terminology" section of About This Document.

Prior to MERLIN MAGIX Release 2.1, the MERLIN LEGEND and MERLIN MAGIX PBX Drivers operated on servers or work stations running Windows NT 4.0. Beginning with MERLIN MAGIX Release 2.1 the MERLIN MAGIX PBX Driver is supported on Windows 2000.

All clients that operate with CentreVu Telephony Services may be used with MERLIN LEGEND/MERLIN MAGIX CTI. At the time of writing, those clients included Microsoft® Windows® 3.1, Windows for Workgroups 3.11, Windows NT, Windows® 95, Windows® 98, Windows® 2000, UnixWare®, and HP-UX. Since backward compatible clients may be released on an incremental basis, there may be additional clients that also operate with CentreVu Telephony Services.

MERLIN MAGIX CTI private data versions 1-3 are provided on the Windows 95, Windows 98, Windows Me, Windows 2000, and Windows NT clients.

TCP/IP is used as the LAN protocol between the clients and the Telephony Server

Architecture

MERLIN LEGEND and MERLIN MAGIX CTI use the Telephony Services architecture and platform infrastructure.

The Telephony Services software consists of two important modules. The Telephony Server is a switch-independent module that manages communication between client workstations and the second major module, the PBX Driver. The PBX Driver is a switch-specific module that interfaces TSAPI with the switch-specific CTI link.

TSAPI is based on the European Computer Manufacturing Association (ECMA) standard for Computer-Supported Telecommunications Applications (CSTA). An application developer may use the TSAPI library to develop client-based or server-based applications.

The MERLIN LEGEND and MERLIN MAGIX switches each support a single CTI link.

System managers use the existing Telephony Services administration software to manage and administer the switch independent portions of the system.

A Windows OA&M client application provides maintenance and administration access to the MERLIN LEGEND PBX Driver or MERLIN MAGIX PBX Driver.

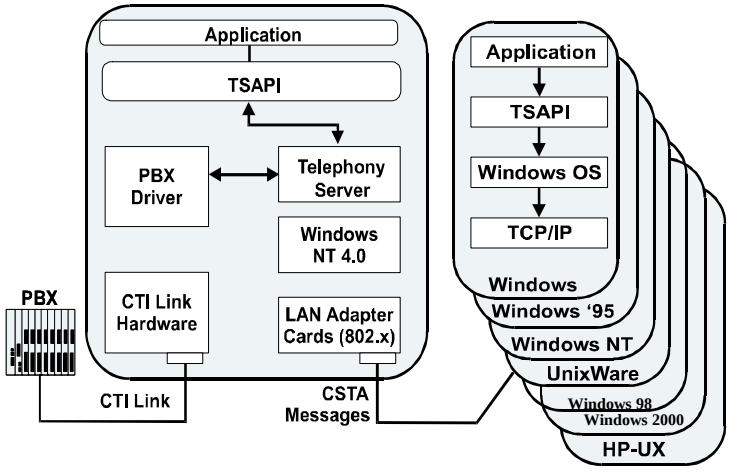


Figure 2-2. CentreVu Computer-Telephony Software Architecture

CSTA Architectural Considerations

TSAPI is based on the ECMA CSTA standard. MERLIN LEGEND and MERLIN MAGIX CTI support the following TSAPI objects and related concepts:

Device Object

TSAPI uses Device Identifiers (DeviceID) to refer to device objects. CTI applications can control and monitor extension devices. In MERLIN MAGIX Release 1.5, support was added to monitor Calling Group Queues.

Event reports may contain identifiers for trunks, but applications cannot directly control or monitor trunk connections.

All MERLIN LEGEND and MERLIN MAGIX TSAPI device identifiers are TSAPI static device IDs. MERLIN LEGEND and MERLIN MAGIX CTI only supports a Device Class of “Voice.”

Call Object

TSAPI uses Call Identifiers (CallID) to refer to call objects. TSAPI call identifiers map to MERLIN LEGEND or MERLIN MAGIX switch Call Identifiers. These TSAPI call identifiers uniquely identify a call within the MERLIN LEGEND or MERLIN MAGIX switch.

Connection Object

TSAPI uses Connection Identifiers (ConnectionID) to refer to the connection of a device to a call. In the programming sense, a ConnectionID is a structure containing a DeviceID and CallID component. A connection identifier identifies the appearance (or appearances) of a call at a device.

MERLIN LEGEND and MERLIN MAGIX CTI Capacity and Limits

The MERLIN LEGEND and MERLIN MAGIX CTI configurations each support a single CTI link.

Table 2-3. MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) CTI Capacity Limits

Parameter	Limit
Maximum number of monitored extensions	136
Maximum number of CTI links	1

The MERLIN LEGEND PBX Driver permits applications to monitor up to 136 extensions. Multiple applications may monitor the same extension.

Table 2-4. MERLIN MAGIX (Release 2.0 and later) CTI Capacity Limits

Parameter	Limit
Maximum number of monitored extensions	200
Maximum number of monitored DGC queues	32
Maximum number of CTI links	1

The MERLIN MAGIX Release 2.0 (and later) switch permits applications to monitor up to 200 extensions and 32 DGC queues. Multiple applications may monitor the same extension or queue.

MERLIN LEGEND and MERLIN MAGIX Support for TSAPI

MERLIN LEGEND/MERLIN MAGIX CTI supports various TSAPI functions and events (but not all of them). The supported TSAPI services and events are shown in Table 2-5.

There are parameters in each service request and event. Refer to the service description or event description page to determine the parameters that MERLIN LEGEND/MERLIN MAGIX CTI supports.

Table 2-5. Support for TSAPI Services and Events

	TSAPI Control Functions and Events
Ö	acsOpenStream( ) & ACSOpenStreamConfEvent
Ö	acsCloseStream( )& ACSCloseStreamConfEvent
Ö	acsAbortStream( )
Ö	acsGetEventBlock( )
Ö	acsGetEventPoll( )
Ö	acsGetFile( ) [where provided in client library]
Ö	acsSetESR( ) [where provided in client library]
Ö	acsEventNotify( ) [where provided in client library]
Ö	acsFlushEventQueue( )
Ö	acsEnumServerNames( )
Ö	acsQueryAuthInfo( )
Ö	ACSUniversalFailureConfEvent
Ö	ACSUniversalFailureEvent
Ö	cstaGetAPICaps( ) & CSTAGetAPICapsConfEvent
Ö	cstaGetDeviceList( ) & CSTAGetDeviceListConfEvent
Ö	cstaQueryCallMonitor( ) & CSTAQueryCallMonitorConfEvent

	TSAPI Call Control Services and Events - MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5)
	cstaAlternateCall( ) & CSTAAlternateCallConfEvent
Ö	cstaAnswerCall( ) & CSTAAnswerCallConfEvent
	cstaCallCompletion( ) & CSTACallCompletionConfEvent
	cstaClearCall( ) & CSTAClearCallConfEvent
Ö	cstaClearConnection( ) & CSTAClearConnectionConfEvent
Ö	cstaConferenceCall( ) & CSTAConferenceCallConfEvent
Ö	cstaConsultationCall( ) & CSTAConsultationCallConfEvent
	cstaDeflectCall( ) & CSTADeflectCallConfEvent
	cstaGroupPickupCall( ) & CSTAGroupPickupCallConfEvent
Ö	cstaHoldCall( ) & CSTAHoldCallConfEvent
Ö	cstaMakeCall( ) & CSTAMakeCallConfEvent
	cstaMakePredictiveCall( ) & CSTAMakePredictiveCallConfEvent
	cstaPickupCall( ) & CSTAPickupCallConfEvent
	cstaReconnectCall( ) & CSTAReconnectCallConfEvent
Ö	cstaRetrieveCall( ) & CSTARetrieveCallConfEvent
Ö	cstaTransferCall( ) & CSTATransferCallConfEvent

TSAPI Call Control Services and Events - MERLIN MAGIX Release 2.0 and later

cstaAlternateCall( ) & CSTAAlternateCallConfEvent
- cstaAnswerCall( ) & CSTAAnswerCallConfEvent

cstaCallCompletion( ) & CSTACallCompletionConfEvent
cstaClearCall( ) & CSTAClearCallConfEvent
- cstaClearConnection( ) & CSTAClearConnectionConfEvent
- cstaConferenceCall( ) & CSTAConferenceCallConfEvent
- cstaConsultationCall( ) & CSTAConsultationCallConfEvent
- cstaDeflectCall( ) & CSTADeflectCallConfEvent

cstaGroupPickupCall( ) & CSTAGroupPickupCallConfEvent
- cstaHoldCall( ) & CSTAHoldCallConfEvent
- cstaMakeCall( ) & CSTAMakeCallConfEvent

cstaMakePredictiveCall( ) & CSTAMakePredictiveCallConfEvent
cstaPickupCall( ) & CSTAPickupCallConfEvent
cstaReconnectCall( ) & CSTAReconnectCallConfEvent
- cstaRetrieveCall( ) & CSTARetrieveCallConfEvent
- cstaTransferCall( ) & CSTATransferCallConfEvent

TSAPI Supplementary Services and Events - MERLIN MAGIX Release 1.5

cstaSetMsgWaitingInd( ) & CSTASetMwiConfEvent
cstaSetDoNotDisturb( ) & CSTASetDndConfEvent
cstaSetForwarding( ) & CSTASetFwdConfEvent
- cstaSetAgentState( ) & CSTASetAgentStateConfEvent

cstaQueryMsgWaitingInd( ) & CSTAQueryMwiConfEvent
cstaQueryDoNotDisturb( ) & CSTAQueryDndConfEvent
cstaQueryFwd( ) & CSTAQueryFwdConfEvent
cstaQueryAgentState( ) & CSTAQueryAgentStateConfEvent
cstaQueryLastNumber( ) & CSTAQueryLastNumberConfEvent
cstaQueryDeviceInfo( ) & CSTAQueryDeviceInfoConfEvent

TSAPI Supplementary Services and Events - MERLIN MAGIX Release 2.0

cstaSetMsgWaitingInd( ) & CSTASetMwiConfEvent
cstaSetDoNotDisturb( ) & CSTASetDndConfEvent
cstaSetForwarding( ) & CSTASetFwdConfEvent
- cstaSetAgentState( ) & CSTASetAgentStateConfEvent

cstaQueryMsgWaitingInd( ) & CSTAQueryMwiConfEvent
cstaQueryDoNotDisturb( ) & CSTAQueryDndConfEvent
cstaQueryFwd( ) & CSTAQueryFwdConfEvent
- cstaQueryAgentState( ) & CSTAQueryAgentStateConfEvent

cstaQueryLastNumber( ) & CSTAQueryLastNumberConfEvent
cstaQueryDeviceInfo( ) & CSTAQueryDeviceInfoConfEvent

TSAPI Supplementary Services and Events - MERLIN MAGIX Release 2.1 and later

- cstaSetMsgWaitingInd( ) & CSTASetMwiConfEvent
- cstaSetDoNotDisturb( ) & CSTASetDndConfEvent
  cstaSetForwarding( ) & CSTASetFwdConfEvent
- cstaSetAgentState( ) & CSTASetAgentStateConfEvent
- cstaQueryMsgWaitingInd( ) & CSTAQueryMwiConfEvent
- cstaQueryDoNotDisturb( ) & CSTAQueryDndConfEvent
  cstaQueryFwd( ) & CSTAQueryFwdConfEvent
- cstaQueryAgentState( ) & CSTAQueryAgentStateConfEvent
  cstaQueryLastNumber( ) & CSTAQueryLastNumberConfEvent
  cstaQueryDeviceInfo( ) & CSTAQueryDeviceInfoConfEvent

TSAPI Monitoring Services and Events

- cstaMonitorDevice( )
  cstaMonitorCall( )
  cstaMonitorCallsViaDevice( )
- CSTAMonitorConfEvent
- cstaMonitorStop( ) & CSTAMonitorStopConfEvent
  cstaChangeMonitorFilter( ) & CSTAChangeMonitorFilterConfEvent
- CSTAMonitorEndedEvent

TSAPI Call Events -
MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX Release 1.0

CSTACallClearedEvent

- CSTAConferencedEvent
- CSTAConnectionClearedEvent
- CSTADeliveredEvent
  CSTADivertedEvent
- CSTAEstablishedEvent
  CSTAFailedEvent
- CSTAHeldEvent
- CSTANetworkReachedEvent
  CSTAOriginatedEvent
  CSTAQueuedEvent
- CSTARetrievedEvent
- CSTAServiceInitiatedEvent
- CSTATransferredEvent

TSAPI Call Events - MERLIN MAGIX Release 1.5 and later

- CSTACallClearedEvent
- Ö CSTAConferencedEvent
- Ö CSTAConnectionClearedEvent
- Ö CSTADeliveredEvent
- Ö CSTADivertedEvent
- Ö CSTAEstablishedEvent
- CSTAFailedEvent
- Ö CSTAHeldEvent
- Ö CSTANetworkReachedEvent
- CSTAOriginatedEvent
- Ö CSTAQueuedEvent
- Ö CSTARetrievedEvent
- Ö CSTAServiceInitiatedEvent
- Ö CSTATransferredEvent

TSAPI Agent Status Events - MERLIN MAGIX Release 1.5

- Ö CSTALoggedOnEvent
- Ö CSTALoggedOffEvent
- CSTANotReadyEvent
- CSTAReadyEvent
- Ö CSTAWorkNotReadyEvent
- CSTAWorkReadyEvent

TSAPI Agent Status Events - MERLIN MAGIX Release 2.0

- Ö CSTALoggedOnEvent
- Ö CSTALoggedOffEvent
- Ö CSTANotReadyEvent
- Ö CSTAReadyEvent
- Ö CSTAWorkNotReadyEvent
- CSTAWorkReadyEvent

TSAPI Agent Status Events - MERLIN MAGIX Release 2.1 and later

- Ö CSTALoggedOnEvent
- Ö CSTALoggedOffEvent
- Ö CSTANotReadyEvent
- Ö CSTAReadyEvent
- Ö CSTAWorkNotReadyEvent
- Ö CSTAWorkReadyEvent

TSAPI Feature Event Reports - MERLIN MAGIX Release 2.0

- CSTACallInfoEvent
- Ö CSTADoNotDisturbEvent
- CSTAForwardingEvent
- CSTAMessageWaitingEvent

TSAPI Feature Event Reports - MERLIN MAGIX Release 2.1 and later

- Ö CSTACallInfoEvent
- Ö CSTADoNotDisturbEvent
- CSTAForwardingEvent
- CSTAMessageWaitingEvent

TSAPI Escape Services - MERLIN MAGIX Release 2.0

- Ö cstaEscapeService( ) & CSTAEscapeServiceConfEvent
- Ö CSTAPrivateEvent
- Ö CSTAPrivateStatusEvent
- CSTAEscapeServiceReq
- cstaEscapeServiceConf( )

TSAPI Snapshot Services - MERLIN MAGIX Release 2.1 and later

- cstaSnapshotCallReq( ) & CSTASnapshotCallConfEvent
- Ö cstaSnapshotDeviceReq( ) & CSTASnapshotDeviceConfEvent

MERLIN MAGIX Escape Services - MERLIN MAGIX Release 2.0

- Ö mlGetDGCGroupList( )
- Ö mlGetDGCGroupMemberList( )
- Ö mlGetDGCGroupTrunkList( )
- Ö mlQueryDeviceName( )
- Ö mlQueryDGCQueueStatus( )
- Ö mlQueryTrunkStatus( )

MERLIN MAGIX Escape Services - MERLIN MAGIX Release 2.1 and later

- Ö mlGetDGCGroupList( )
- Ö mlGetDGCGroupMemberList( )
- Ö mlGetDGCGroupTrunkList( )
- Ö mlQueryDeviceName( )
- Ö mlQueryDGCGroupDAUInfo( )
- Ö mlQueryDGCGroupParameters( )
- Ö mlQueryDGCQueueStatus( )
- Ö mlQueryTrunkStatus( )

The detailed descriptions of the TSAPI functions and events are split into several chapters:

- Chapter 3: This chapter contains detailed descriptions for control services and events that an application uses to start, stop, and manage a telephony services communication stream. Certain functions and events in this chapter are API Control Services (and have names prefixed with ACS), while others derive from CSTA (and are prefixed with CSTA).

  In addition, there are a number of TSAPI services that are implemented entirely in the client libraries and do not require any interaction with the MERLIN LEGEND or MERLIN MAGIX switch. MERLIN LEGEND/MERLIN MAGIX CTI supports these services and while an application may use these services, Chapter 3 does not contain detailed manual pages for them (refer to Telephony Services Application Programming Interface (TSAPI) Version 2). Chapter 3 does list the services and events that are in this category.
- Chapter 4: This chapter contains detailed descriptions for the services that an application uses to control calls.
- Chapter 5: This chapter contains detailed descriptions for the supplementary services that an application uses to request or change agent or switch feature status.
- Chapter 6: This chapter contains detailed descriptions for the services that an application uses to monitor devices.
- Chapter 7: This chapter contains detailed descriptions for the services that an application uses to take a snapshot of calls and call states at a device.
- Chapter 8: This chapter contains detailed descriptions of the call events that the MERLIN LEGEND and MERLIN MAGIX switches provide for a monitored device and calls.
- Chapter 9: This chapter contains detailed descriptions of the Feature Events that the MERLIN MAGIX switch provides for a monitored device.
- Chapter 10: This chapter contains detailed descriptions of the Agent Status Events that the MERLIN MAGIX switch provides for a monitored device.
- Chapter 11: This chapter contains detailed descriptions of the escape services that are supported in MERLIN MAGIX Release 2.0 and later.
- Chapter 12: This chapter provides TSAPI event flows for a variety of scenarios.

The MERLIN LEGEND and MERLIN MAGIX switches do not support any TSAPI service or event that is not specifically cited in this book.

MERLIN LEGEND and MERLIN MAGIX CTI provide TSAPI version 2.

NOTE:
TSAPI versions should not be confused with Telephony Services or MERLIN LEGEND or MERLIN MAGIX product release numbers. A Telephony Services Product Release (such as Release 2 Telephony Services) refers to a specific release of the product software. TSAPI, one component of the product, undergoes modification over time. The TSAPI modifications are called “versions.” Thus, the TSAPI version numbers are independent of the Telephony Services product release numbers. A PBX Driver in the Telephony Services architecture supports a set of TSAPI versions. Thus, while CentreVu Telephony Services supports TSAPI versions 1 and 2, the MERLIN LEGEND PBX Driver and MERLIN MAGIX PBX Driver only support TSAPI version 2.

Programming Guidelines for MERLIN LEGEND and MERLIN MAGIX CTI Applications

Recommendations for TSAPI Application Use

The following sections note certain configuration recommendations that apply when TSAPI applications are in use.

Direct Line Console (DLC) Configuration

DLCs are used for a variety of purposes. It is important that the role of the DLC in the presence of TSAPI applications be clearly defined. In some uses, such as a DGC supervisor, the DLC may be a monitored station. This lets the DGC supervisor run the same CTI application as the group members. In other cases, such as a receptionist who transfers all incoming calls to customer service representatives, it may not be desirable to monitor the DLC.

Microphone Mute Recommendation

TSAPI applications should never be used at an extension with the microphone mute (MIC-MUTE) feature enabled.

NOTE:
Application programmers should provide this recommendation in their product documentation.

TSAPI Application Use with Existing Voice Mail Systems

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, some existing voice mail systems may be configured in such a way as to conserve SA buttons. As a result, incoming calls arrive on LINE buttons, not SA buttons, making it impossible for an application to provide screen pops. Applications providers must be aware that a change in customer configuration is necessary in this case to take full advantage of TSAPI features.

Beginning with MERLIN MAGIX Release 2.0, LINE buttons receive the supported events and thus the above restriction no longer applies.

Monitoring

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application cannot use TSAPI to monitor or control a VMI (voice prompting) port.

Beginning with MERLIN MAGIX Release 2.0 , an application can use TSAPI to monitor and control Single Line Sets (including VMI ports).

Confirmation Events & Unsolicited Event Ordering

Each service's manual page contains a section describing the service's confirmation event and the semantics for that service's confirmation event. In general, when an application requests a service, MERLIN LEGEND/MERLIN MAGIX CTI provides the confirmation event for that service before any events flow as a result of the service invocation.[7] There is one exception worth noting:

- ***cstaConsultationCall( )*** - sends the confirmation event for the consultation call service after the active call has been placed on hold and before the consultation call is originated. Thus, the confirmation event comes after the ***CSTAHeldEvent*** and before the ***CSTAServiceInitiatedEvent***.

NOTE:
Applications, especially applications that are to be switch-independent, should never depend on a relative ordering of a service confirmation and the resulting events. Feature interactions and differing switch architectures can cause this to vary. Applications should use the events, rather than service confirmations to reflect call status in a switch-independent way. This also facilitates connection tracking when manual operations and service requests from other applications change connection states.

The MERLIN LEGEND and MERLIN MAGIX switches treat TSAPI call control service requests in two ways:

1. A service request may be an atomic operation that, once switch processing begins, is processed to completion. In this case, the service confirmation means that the service has successfully completed.
2. A service may be broken into a number of discrete call processing operations. A confirmation for the ***cstaMakeCall( )*** or ***cstaConsultationCall( )*** service does not mean that the service has successfully completed. Events that follow the confirmation track the progress of the service request. Refer to the service manual pages for details.

[7] This is a characteristic of MERLIN LEGEND and MERLIN MAGIX switch behavior and not a part of the TSAPI specification.

Transferring or Conferencing a Call with Original Call Information (Consultation Call)

An application may use the ***cstaConsultationCall( )*** service to extend a call to another user in such a way that an application running on behalf of the receiving user can use the Original Call Information (private data) to pop a screen. An application monitoring the extension receiving the consultation call can use the Original Call Information to pop a screen:

- as soon as the consultation call alerts,
- when the consultation call is answered, or
- to retain the Original Call Information for a later screen pop (such as when the consultation call is transferred).

When a user transfers (or conferences) a call, the operation may be:

- supervised – the consulting party waits for the consulted party to answer.
- unsupervised – the consulting party immediately completes the operation without waiting for the consulted party to answer.

Application designers should be aware that the availability of information about the original call varies in manual operations. In a supervised scenario (i.e., the consultation call is answered before completing the transfer or conference), information about the original call is not available when the consultation call rings (or is answered). Some information about the original call becomes available in the ***CSTATransferredEvent*** (or ***CSTAConferencedEvent***). In an unsupervised scenario (i.e., the transfer or conference operation is completed before the consultation call is answered), some information about the original call is also available in the event, but the sequencing of that event with respect to the delivered and established events varies. Chapter 12 contains sections showing event flows for a variety of consultation calls and manual scenarios (supervised and unsupervised).

NOTE:
An application must use the ***cstaConsultationCall( )*** service to conference or transfer a call. An application cannot use a sequence of the ***cstaHoldCall( )***, ***cstaMakeCall( )*** and ***cstaTransferCall( )*** (or ***cstaConferenceCall( )***) services. In a MERLIN LEGEND or MERLIN MAGIX switch environment, the ***cstaHoldCall( )*** service does not put a call on hold-for-conference or on hold-for-transfer.

NOTE:
When a user uses the telephone set to manually transfer or conference a call to another user, an application running on behalf of the receiving user *does not* receive the private data containing Original Call Information. Some information about the original call may be available in the ***CSTATransferredEvent*** or ***CSTAConferencedEvent***. The information available depends on factors such as the type of trunk the original call arrived on. The timing of the information with respect to the delivered and established events varies according to whether the manual operation was supervised or unsupervised. Chapter 12 contains detailed event flows for a variety of scenarios.

Private Data in MERLIN MAGIX Release 2.0

In MERLIN MAGIX Release 2.0, Original Call Information (OCI) is provided for the following cases:

- When a call goes to any type of cover button, the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** for the cover button will contain the Coverage Sender in the Original ***calledDevice***. The Original ***callingDevice*** will contain the Internal Extension number or the ANI number when available. The ***lastRedirectionDevice*** will contain the Coverage sender.
- When a call goes to the forwarded-to station, the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** will contain the Forward-from extension in the Original ***calledDevice***. The Original ***callingDevice*** will contain the Internal Extension number or the ANI number when available. The ***lastRedirectionDevice*** will contain the Forward-from extension.
- For calls that get picked up, the ***CSTAEstablishedEvent*** for the station that performs the pickup will receive Original Call Information if it is monitored and the station where the call is being picked up from is monitored. The ***CSTAEstablishedEvent*** will contain the picked-up extension in the Original ***calledDevice.***

When a trunk call arrives or is answered, the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** will contain the trunk id in the ***trunkUsed*** parameter in Private Data.

When an external call is disconnected, the ***CSTAConnectionClearedEvent*** for the station where an account code was entered will contain that information.

Private Data in MERLIN MAGIX Release 2.1 and Later

Beginning with MERLIN MAGIX Release 2.1, Original Call Information (OCI) is no longer provided for the following cases:

- When a call goes to any type of cover button, the ***calledDevice*** parameter in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** accurately identifies the extension number of the Coverage Sender, so it is not necessary to provide this data in OCI.
- When a call is forwarded to another extension, the ***calledDevice*** parameter in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** accurately identifies the extension number of the forwarding extension, so it is not necessary to provide this data in OCI.
- For calls that get picked up, the ***calledDevice*** parameter in the ***CSTAEstablishedEvent*** for the station that performs the pickup accurately identifies the extension number of the picked up station, so it is not necessary to provide this data in OCI.

Programming for Busy Conditions

The MERLIN LEGEND and MERLIN MAGIX switches contain a number of features that ensure that processing of a call that meets a busy condition continues in an appropriate manner. Thus, encountering a busy condition does not imply that call processing for the call stops, or that the call has "failed." Some examples:

- A user transfers a call to a busy extension. The call may be queued for the destination extension. If the call is not answered at the destination extension, it returns to the transferring extension.
- A user transfers a call, parks a call, or camps a call onto an extension that does not connect to the call. The MERLIN LEGEND and MERLIN MAGIX switches will let the call remain in that state, waiting for the destination to become available for some length of time, and then will return the call to the extension that performed that operation on the call.

NOTE:
The MERLIN LEGEND and MERLIN MAGIX switches do not provide the TSAPI ***CSTAFailedEvent***.

Since a call that is delivered to a destination generates a ***CSTADeliveredEvent***, an application should use the presence of a ***CSTADeliveredEvent*** to indicate that the call is alerting at the destination. The absence of a ***CSTADeliveredEvent*** indicates that the call has not yet alerted at a destination; however, the call may be delivered to an extension at a future point. Users who hear a busy signal, and do not wish to wait to see if other call processing features will deliver the call to an alternate destination, may manually hang up the call or can use an application to request ***cstaClearConnection( )*** for the appearance of that call at their extension.

Trunk Events for External Connections

When a call leaves the switch on a non-PRI trunk, MERLIN LEGEND/MERLIN MAGIX CTI also provides the ***CSTANetworkReachedEvent***. Once the switch provides that event, it does not provide any further events pertaining to the trunk endpoint.

When a call leaves the switch on a PRI trunk, MERLIN MAGIX CTI also provides the ***CSTANetworkReachedEvent***. In addition, beginning with MERLIN MAGIX Release 2.0, MERLIN MAGIX CTI provides the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** when the call alerts and is answered—provided that the call has been routed on digital facilities.

Feature Interactions

Applications designers must be aware that use of certain features will terminate the MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) switch event reporting for calls. Of special note are:

- Pickup - an application monitoring an extension where a user has used the call pickup feature will not receive events for the call (specifically a ***CSTAEstablishedEvent)***. (Beginning with MERLIN MAGIX Release 2.0, an application will receive events for the call).
- Forward/Follow Me - an application monitoring an extension receiving a forwarded call will not receive events for the call. (Beginning with MERLIN MAGIX Release 2.0, an application will receive events for a forwarded call).

The following feature will end MERLIN LEGEND and MERLIN MAGIX switch event reporting.

- Shared System Access - See “Shared System Access Interactions” later in this chapter.

Applications designers may determine that specific interactions are not relevant to the application, may design specific event handling for such interactions into the application, or may document specific recommendations about the use (or prohibiting the use) of specific features with an application.

Shared System Access Interactions

An understanding of MERLIN LEGEND and MERLIN MAGIX Shared System Access (SSA) terminology and its relationship to the TSAPI model will help in understanding the TSAPI event flows that occur when connections interact with Shared System Access buttons.

An SSA button on an extension provides an appearance of an SA button at another extension. Using SSA buttons causes connections at the SA button to transition into the MERLIN LEGEND/MERLIN MAGIX *associative active* and *associative held* states. MERLIN LEGEND/MERLIN MAGIX CTI makes a distinction between the TSAPI connected and held states and the associative states (which TSAPI does not model).

In MERLIN LEGEND/MERLIN MAGIX CTI terminology, when a call is alerting at an SA button and a user at another station presses an SSA button and connects to that call, that user has *answered* the call. The state of the call at the SA button changes to associative active. The state of the call at the SSA is connected (a TSAPI state). Thus, an application monitoring an extension where an SSA answers a call will receive further events about the call.

When a call is active at a SA button and a user at another station presses an SSA button and connects to that call, the user *bridged* onto the call. The state of the call at the SA button remains active. The state of the call at the SSA is bridged (not a TSAPI state). Thus, an application monitoring an extension where an SSA bridges onto a call will not receive further events about the call.

Depending on whether an SSA user answers a call or bridges onto a call, event flows will differ for an application monitoring the extension with the SSA button.

The following rules govern event flows when SSA buttons interact with calls:

- MERLIN LEGEND/MERLIN MAGIX CTI considers connections that transition into the associative or bridged states as having left the defined TSAPI model. As a result, they are considered to have been cleared from the device where this transition occurred, and any applications monitoring the device with the SA button where this occurs will receive a ***CSTAConnectionClearedEvent*** the first time a connection transitions into an associative state.
- Once MERLIN LEGEND/MERLIN MAGIX CTI supplies a ***CSTAConnection-ClearedEvent*** for a connection in an associative state at a device, there will be no further events generated for that connection at the device. The device may reconnect to the call and MERLIN LEGEND/MERLIN MAGIX CTI will not supply any further events. (Note that the call is still in an associative state.)
- An application monitoring an extension where an SSA answers a call will receive events for that call (so long as the call does not enter an associative state due to some later feature interaction).
- An application monitoring an extension where an SSA bridges onto a call will not receive events for that call.
- Applications monitoring an extension having an SSA button do not receive any events about an incoming call on the corresponding SA button unless a user at the extension with the SSA button uses the SSA button to answer the call. Of special interest is the fact that such an application will not receive a ***CSTADeliveredEvent***. Thus, the application cannot be aware of the call on the corresponding SA button and the user must manually answer the call on the SSA button.

Coverage Button Interactions

Beginning with MERLIN MAGIX Release 2.0, an application receives ***CSTADeliveredEvents***, ***CSTAEstablishedEvents,*** and ***CSTAConnectionClearedEvents*** for calls on Cover buttons at monitored stations. Until a call is answered it can appear (alert) at several monitored stations with Cover buttons.

- A device monitor for the coverage sender will receive events that describe call activity for a coverage call at the coverage sender and at all coverage receivers.
- A device monitor for a coverage receiver will only receive events that describe call activity at the coverage sender and at the monitored coverage receiver; it will not receive events describing call activity at other coverage receivers.

Direct Facility Termination (DFT) and Direct Pool Termination (DPT) Interactions

An extension may have a call appear at multiple buttons. Of special importance is the case when a call appears at an SA button and a Direct Facility Termination (DFT) button.

"Shared System Access Interactions" earlier in this chapter explains, in detail, how using another button (there an SSA button) to answer or connect to a call on an SA button will cause the call to transition to an associate state at the SA button. The rules detailed in "Shared System Access Interactions" pertaining to event reporting in associative and bridged states also apply to DFT interactions. For example, the same interaction occurs when a DFT button answers or bridges onto a call: the connection at the SA button transitions into an associative state. Like an SSA button, a DFT button can answer or bridge onto a call (and the rules detailed in "Shared System Access Interactions" apply).

NOTE:
When an incoming call appears at an SA and a DFT button on a monitored extension, a monitoring application will receive a Delivered event because the call is ringing on an SA button. If the user presses the DFT to answer the call, the SA button transitions to associative active, and the DFT transitions to connected, so, using the rules detailed in "Shared System Access Interactions," the monitoring application will receive an Established event.

NOTE:
Beginning with MERLIN MAGIX Release 2.0, a monitoring application will receive the ***CSTADeliveredEvent*** for a call that is alerting on a DFT button. If the call is answered at another appearance of the DFT, the application will receive a ***CSTAConnectionClearedEvent*** for the call.

Beginning with MERLIN MAGIX Release 2.0, an application receives ***CSTADeliveredEvents***, ***CSTAEstablishedEvents,*** and ***CSTAConnectionClearedEvents*** for calls on DFT and DPT buttons on monitored stations. Until a call is answered it can appear (alert) at several monitored stations on DFT or DPT buttons. In general, a device monitor will only receive events describing call activity at the DFT or DPT button on the monitored station; it will not receive events describing call activity at other DFT or DPT buttons where the call appears.

Networking

Beginning with MERLIN LEGEND Release 6.0, the system supports the networking of multiple switches together. The switch with the server connected will receive all events for available devices on that switch (i.e. extensions and queues). Once the call has left the switch, events will no longer be provided. It is suggested that configurations that use server based CTI reporting applications not use non-local calling groups because the events are not generated for calls that leave the system and the application may not reflect the true state of the calls.

Beginning with MERLIN MAGIX Release 2.0, the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** are provided for outgoing network calls provided the network consists of digital PRI trunks.

MERLIN LEGEND and MERLIN MAGIX Private Data Libraries

Both MERLIN LEGEND CTI and MERLIN MAGIX CTI include a TSAPI Private Data Library. The next sections below indicate what information is available in the private data libraries.

MERLIN LEGEND Private Data Library & Collected Digits

MERLIN LEGEND CTI includes a TSAPI Private Data Library. The private data library supports private data version 1. The Private Data Library provides Collected Digits in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent***.

MERLIN LEGEND private data is provided on the Windows 3.1, Windows 95, Windows 98, Windows Me, Windows NT, Windows 2000, and UnixWare clients. It is not provided on other CentreVu Computer-Telephony clients.

The MERLIN LEGEND switch provides collected digits when incoming external call routes to a voice response unit (VMS/AA) connected to a VMI port. The voice response unit may request a caller to input data. If the caller provides caller input data, then that data will be present in the private data parameter for any ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** for the call.

When an application uses the ***cstaConsultationCall*** service to transfer a call with associated collected digits to another extension, the collected digits, like the original calling number and DNIS, is present in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** for the consultation call. The ***cstaConsultationCall*** service description (Chapter 4) and ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** event description sections (Chapter 8) provide further details.

MERLIN MAGIX Private Data Library & Collected Digits

MERLIN MAGIX CTI includes a TSAPI Private Data Library. The Private Data library supports Private Data Versions 1-3. The Private Data Library provides Collected Digits in the ***CSTADeliveredEvent, CSTAQueuedEvent*** and ***CSTAEstablishedEvent***.

MERLIN MAGIX private data is provided on the Windows 95, Windows 98, Windows Me, Windows NT and Windows 2000 clients. It is not provided on other CentreVu Computer-Telephony clients.

The MERLIN MAGIX switch provides collected digits when incoming external call routes to a voice response unit (VMS/AA) connected to a VMI port. The voice response unit may request a caller to input data. If the caller provides caller input data, then that data will be present in the private data parameter for any ***CSTADeliveredEvent, CSTAQueuedEvent*** and ***CSTAEstablishedEvent*** for the call.

When an application uses the ***cstaConsultationCall*** service to transfer a call with associated collected digits to another extension or to a Calling Group, the collected digits, like the original calling number and DNIS, is present in the ***CSTADeliveredEvent***, ***CSTAEstablishedEvent*** and ***CSTAQueuedEvent*** for the consultation call. The ***cstaConsultationCall*** service description (Chapter 4) and ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** event description sections (Chapter 8) provide further details.

Collected Digit System Operation

In order for a TSAPI application to make use of collected digits, the system must be configured for digit collection. The digit collection operates as follows:

1. The MERLIN LEGEND or MERLIN MAGIX switch directs an incoming external call to an idle port of the VMI-DGC group. The Interactive Voice Response (IVR) system then answers and provides the Automated Attendant feature.

2. The customer enters a selector code at either the Main Menu or a Sub-menu that has been administered for “Collected Digits Transfer.” The Collected Digits Transfer is a special Automated Attendant Selector Code that is associated with:

- an announcement
- a maximum collected digit length of up to 32 digits
- an extension number to which the call is transferred

3. The IVR system plays the associated announcement.
4. The caller enters 0 to 32 digits until one of the following occurs:
   - four seconds elapses with no DTMF digit entered
   - the associated maximum collected digit length is reached
   - a # or * is detected
5. The IVR system then transfers the call to #58<CDIG>#<EXT> where:
   - #58 indicates collected digit information transfer
   - <CDIG> is the string of collected digits
   - # is a delimiter
   - <EXT> is the associated extension to which the call is transferred

MERLIN LEGEND and MERLIN MAGIX Private Data Libraries and Original Call Information

Applications such as incoming customer service applications may use information about the call (such as ANI, ICLID, DNIS, or collected digits) to pop a screen for a customer representative when the incoming call alerts at the representative's desk. If that representative then transfers the call or conferences to another representative, then it is often desirable to use the information about the original calling party to pop a screen for the new representative. The second representative may run an application different from the first representative, but also use information about the original call to pop an application screen.

When an application uses the ***cstaConsultationCall( )*** service to extend a call from one user to another, the ***CSTADeliveredEvent, CSTAEstablishedEvent*** and/or (beginning with MERLIN MAGIX Release 2.0) ***CSTAQueuedEvent*** that result from the consultation call contain private data giving information about the original call. An application monitoring the device receiving the consultation call can then use information about the original call to pop a screen, or, beginning with MERLIN MAGIX Release 2.0, to redirect the call using the ***cstaDeflectCall( )*** service. Figure 2-3 illustrates a situation where a caller's information popped a screen at a claims agent's desk about a claim for a very expensive automobile. The claims agent, wanting to retain the funds within the company while the car is replaced, has transferred the caller to an investment specialist. The investment specialist's application will pop a screen (different than the claims screen) using information about the original call.

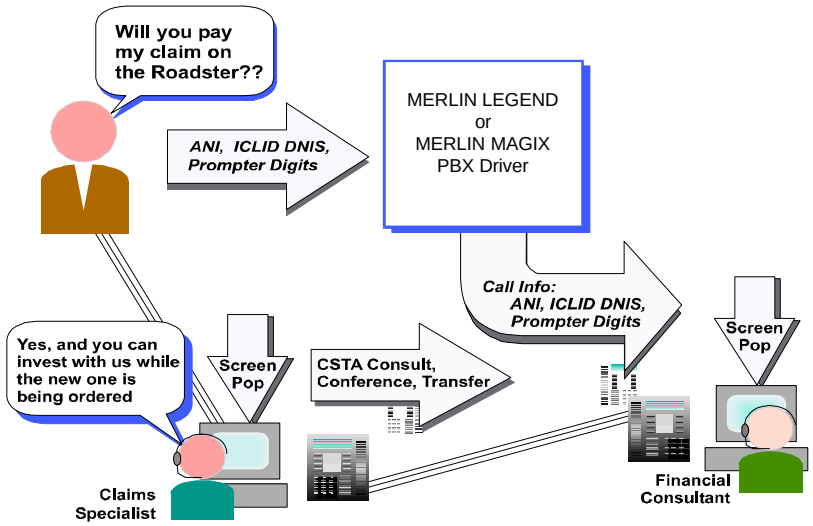


Figure 2-3. Original Call Information Illustration

Two conditions must occur for Original Call Information to pass in events to an application running on behalf of a user receiving a call.

1. An application must use ***cstaConsultationCall( )*** to have Original Call Information passed with a call. Manual user operations will not pass Original Call Information in the events for a call.
2. An application must be monitoring the extension[7] from which the consultation call is being made. This is the Claims Specialist in the figure above.

MERLIN MAGIX Private Data and Original Call Information for Forwarded and Covered Calls

In MERLIN MAGIX Release 2.0, when a call is forwarded to another extension, an application monitoring the forwarding destination receives a ***CSTADeliveredEvent***; when the call is answered, the application receives a ***CSTAEstablishedEvent***. However, MERLIN MAGIX Release 2.0 does not accurately populate the ***calledDevice*** parameter in the ***CSTADeliveredEvent*** and the ***CSTAEstablishedEvent***. If the call has been forwarded from an SA button, an application may determine the actual called device (the forwarding

7 "Monitoring" means that the application has used ***cstaMonitorDevice( )*** to request events for that extension.

extension) by examining the Original Call Information (OCI) ***calledDevice*** parameter in Private Data. Beginning with MERLIN MAGIX Release 2.1, the ***calledDevice*** parameter in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** accurately identifies the extension number of the forwarding extension, so no OCI is provided in Private Data.

In MERLIN MAGIX Release 2.0, when a call alerts on a Coverage button at an extension, an application monitoring that extension receives a ***CSTADeliveredEvent***; when the call is answered, the application receives a ***CSTAEstablishedEvent***. However, MERLIN MAGIX Release 2.0 does not accurately populate the ***calledDevice*** parameter in the ***CSTA-DeliveredEvent*** and the ***CSTAEstablishedEvent***. The application may determine the actual called device (the Coverage Sender) by examining the Original Call Information (OCI) ***calledDevice*** parameter in Private Data. Beginning with MERLIN MAGIX Release 2.1, the ***calledDevice*** parameter in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** accurately identifies the extension number of the Coverage Sender, so no OCI is provided in Private Data.

MERLIN MAGIX Private Data Library and Trunk ID

Beginning with MERLIN MAGIX Release 2.0, the MERLIN MAGIX private data library provides the Trunk ID for external (incoming or outgoing) calls as private data in the ***CSTADeliveredEvent, CSTAQueuedEvent*** and ***CSTAEstablishedEvent***. The format of the trunk ID is “Txxxx”, where xxxx is the administered number for the line (by default these are 801-880).

MERLIN MAGIX Private Data Library and Account Codes

Beginning with MERLIN MAGIX Release 2.0, the MERLIN MAGIX private data library provides the Account Code for external (incoming or outgoing) calls as private data in the ***CSTAConnectionClearedEvent***. Beginning with MERLIN MAGIX Release 2.1, the ***CSTACallInfoEvent*** is added to also provide the Account Code at the time the information is entered by a user.

Although Account Code information is still provided in Private Data in the ***CSTAConnectionClearedEvent*** , an application should use the ***CSTACallInfoEvent*** to collect Account Code information.

Extracting Private Data from Events

Certain events carry MERLIN LEGEND or MERLIN MAGIX private data. The following code fragment shows how an application can extract the private data.

```
/*
 * Code fragment to retrieve MERLIN LEGEND or MERLIN MAGIX
 * private data from a CSTA event.  A pointer to the buffer
 * privateDataBuffer has been passed to acsGetEventPoll( )
 * or acsGetEventBlock( ).
 */
#include <mlpriv.h>

MLPrivateData_t   privateDataBuffer;
MLEvent_t         mlEventBuffer;
RetCode_t         rc;

/*
 * Did the application receive MERLIN LEGEND or MERLIN MAGIX private
 * data?
 */
if ( privateDataBuffer.length != 0 &&
   strcmp(privateDataBuffer.vendor, ML_VENDOR_STRING) == 0 ) {
      /*
       * Received MERLIN LEGEND or MERLIN MAGIX private data.
       * Transfer the data to a MLEvent_t structure.
       */
      mlPrivateData( &privateDataBuffer, &mlEventBuffer );

      switch ( mlEventBuffer.eventType ) {

      case ML_CONNECTION_CLEARED:
         /*
          * Add code here to extract the private data sent
          * in a CSTAConnectionClearedEvent. Refer to the
          * Private Data Syntax section of the manual page for
          * the CSTAConnectionClearedEvent, Chapter 8.
          */
      break;

      case ML_DELIVERED:
      case MLV1_DELIVERED:
         /*
          * Add code here to extract the private data sent
          * in a CSTADeliveredEvent. Refer to the Private Data
          * Syntax section of the manual page for the
          * CSTADeliveredEvent, Chapter 8.
          */
      break;
```

```
        case ML_ESTABLISHED:
        case MLV1_ESTABLISHED:
           /*
            * Add code here to extract the private data sent
            * in a CSTAEstablishedEvent. Refer to the Private Data
            * Syntax section of the manual page for the
            * CSTAEstablishedEvent, Chapter 8.
            */
        break;

        case ML_QUEUED:
           /*
            * Add code here to extract the private data sent
            * in a CSTAQueuedEvent. Refer to the Private Data
            * Syntax section of the manual page for the
            * CSTAQueuedEvent, Chapter 8.
            */
        break;

        case ML_GETAPI_CAPS_CONF:
           /*
            * Add code here to extract the private data sent
            * in a CSTAGetAPICapsConfEvent. Refer to the Private Data
            * Syntax section of the manual page for
            * cstaGetAPICaps( ), Chapter 3.
            */
        break;

        }
    }
```

NOTE:
An application must ask for private data when it opens a stream. Refer to the ***acsOpenStream( )*** manual page in Chapter 3 for details.

Control Services and Events

3

Contents

Contents

Control Services and Events

3

Control services[1] consist of TSAPI API Control Services (ACS) and certain basic CSTA control services.

Applications use TSAPI Control Services to:

- Open a Telephony Services stream. Once an application successfully opens a stream, the application can monitor devices, make call control requests, and receive events on the stream.
- Select the TSAPI version for use on the stream (when opening the stream).
- Select a private data version for use on the stream (when opening the stream).
- Close a Telephony Services stream.
- Abort a Telephony Services stream.
- Block or poll for events on a Telephony Services stream.
- Initialize an operating system event notification facility for events arriving on a Telephony Services stream.
- Query for a list of all available advertised services (switch driver services).
- Query for the CSTA services available on the stream.
- Query for a list of devices that the application may monitor or control.
- Query to determine if user permissions allow Call/Call monitoring on the stream. (The MERLIN LEGEND and MERLIN MAGIX switches do not provide Call/Call monitoring.)

Table 3-1 shows the TSAPI control services and events. The MERLIN LEGEND and MERLIN MAGIX switches do not provide all of the optional parameters for the control services and events.

1 These control services are described in Chapter 4 of the TSAPI specification.

Table 3-1. MERLIN LEGEND/MERLIN MAGIX CTI Support for TSAPI Control Services and Events

TSAPI Control Functions and Events
Ö acsOpenStream( ) & ACSOpenStreamConfEvent
Ö acsCloseStream( )& ACSCloseStreamConfEvent
Ö acsAbortStream( )
Ö acsGetEventBlock( )
Ö acsGetEventPoll( )
Ö acsGetFile( ) [where provided in client library]
Ö acsSetESR( ) [where provided in client library]
Ö acsEventNotify( ) [where provided in client library]
Ö acsFlushEventQueue( )
Ö acsEnumServerNames( )
Ö acsQueryAuthInfo( )
Ö ACSUniversalFailureConfEvent
Ö ACSUniversalFailureEvent
Ö cstaGetAPICaps( ) & CSTAGetAPICapsConfEvent
Ö cstaGetDeviceList( ) & CSTAGetDeviceListConfEvent
Ö cstaQueryCallMonitor( ) & CSTAQueryCallMonitorConfEvent

NOTE:
The ***cstaQueryCallMonitor( )*** and ***cstaGetDeviceList( )*** services indicate whether the Telephony Services Security Database gives permissions for the application to make certain requests on a given stream. Even though the Telephony Services Security Database permissions may be enabled for various services, the MERLIN LEGEND and MERLIN MAGIX switches do not support certain services.

NOTE:
The ACS confirmation events are a part of two unions, `ACSEvent_t` and `CSTAEvent_t`. Typically a program will use the `CSTAEvent_t` union since it spans both the CSTA and ACS events. Thus, the *Syntax* sections in this chapter show `CSTAEvent_t`.

Opening, Closing, and Aborting a Stream

An application must open a stream over which it may then request monitors and control services. Opening a stream creates a logical link from the application, through the Telephony Server and PBX driver, to the MERLIN LEGEND or MERLIN MAGIX switch. The Telephony Server software and PBX driver cooperate to provide stream resources and do permissions checking for application requests.

NOTE:
Application design, in some circumstances, may require a working knowledge of the Telephony Services Security Database. An application that needs to monitor several phones on a stream must open the stream giving user information for a user who has permissions in the Security Database to monitor those devices. Refer to *CentreVuÒ Computer-Telephony Telephony Services Administration and Maintenance* for additional information.

A PBX driver (such as the MERLIN LEGEND or MERLIN MAGIX driver) registers one or more physical CTI links as an advertised service(s). When an application opens a stream, it must specify the advertised service. An application may open streams to several different advertised services.

When an application opens a stream, it receives an ***acsHandle*** that identifies that stream for its lifetime.

An application is responsible for closing or aborting any stream that it opens. If an application needs to quickly shut down a stream and release stream resources in a single step, then the application should use ***acsAbortStream( )*** to abort the stream. Aborting a stream terminates any call control in progress and flushes the event buffers for the stream. If an application needs to close a stream in a more orderly fashion (one that provides the application with all the outstanding events and confirmations), then the application should use ***acsCloseStream( )***.

CAUTION:
A stream remains open until the application receives the ***ACSCloseStreamConfEvent*** *on that stream. When an application uses* ***acsCloseStream( )*** *to close a stream, it must continue to receive events for that stream until it receives the* ***ACSCloseStreamConfEvent****. If an application fails to do this, the system may not immediately release all of the stream resources.*

Closing a stream does not affect the switch processing of any calls that have been controlled or monitored on that stream.

The *Telephony Services Application Programming Interface (TSAPI)* specification has step-by-step procedures in Chapter 4 for opening, closing, and aborting a stream.

Sending TSAPI Requests and Receiving Confirmations

After an application opens a stream, it may request services on that stream. In each service request, the application passes the ***acsHandle*** for the stream.

An application supplies an ***invokeID*** with each service request. Applications may have several service requests outstanding, so the ***invokeID*** lets the application correlate service confirmation events with service requests. When an application opens a stream, it specifies whether:

- the application will explicitly provide values for each ***invokeID***. In this case, the application provides a 32-bit value for ***invokeID***. If a service request returns a negative value, the function call for the request was not successful. If the function returns zero, then the service request was successful and the service confirmation event will contain the application-provided ***invokeID***.
- the TSAPI client library will generate unique values for each ***invokeID***. In this case, when the function returns, a negative value indicates an error and a positive value is the ***invokeID*** value for this request. The service confirmation event will contain the library-provided ***invokeID***.

NOTE:
In general, having the TSAPI library generate ***invokeID***s simplifies application design. However, when service requests correspond to entries in a data structure, it may simplify application design to use the indexes into the data structure as the ***invokeID***s. Application-generated ***invokeID***s might also point to Windows handles. Application-generated ***invokeID***s may take on any 32-bit value.

The *Telephony Services Application Programming Interface (TSAPI)* specification has step-by-step procedures in Chapter 4 for sending requests and receiving confirmations.

Receiving Events

When an application successfully opens a stream, TSAPI queues the ***ACSOpenStreamConfEvent*** for the application. Any additional confirmation or call events will arrive on the same queue. To receive an event, the application must use one of two event handling modes:

- blocking — The application uses ***acsGetEventBlock( )*** to block (does not execute) until an event becomes available. Blocking is appropriate in threaded or preemptive operating system environments.
- non-blocking — The application uses ***acsGetEventPoll( )*** to receive an event (if one is queued) and then returns control to the application regardless of whether an event is available.

CAUTION:
Blocking may be appropriate for applications that monitor a device and require processing only when an event occurs. However, there may be operating system specific implications. For example, if a Windows 3.1 application blocks waiting for call events, then it cannot process events from its Windows queue.

When an application receives an event, it may specify that the event is to be taken from the queue belonging to a specific stream, or from a queue for any open stream. TSAPI provides events in chronological order for the specified streams. Thus, if the application always receives all events from all streams, TSAPI will pass the application the events in the order of their arrival.

In some operating system environments, an application may set an Event Service Routine (ESR) so that the operating system passes the application an asynchronous notification when an event arrives. This mechanism does not remove events from the event queue. The application must use ***acsGetEventBlock( )***, ***acsGetEventPoll( )***, or ***acsEventNotify( )*** to receive the event. See the TSAPI manual page for ***acsSetESR( )*** for more information.

An application may use ***acsFlushEventQueue( )*** to flush events from a specified queue or queues.

The *Telephony Services Application Programming Interface (TSAPI)* specification has step-by-step procedures for receiving events in Chapter 4.

TSAPI Version Control

As TSAPI evolves over time, it will include more services and events. To ensure that applications written with earlier versions of TSAPI can continue to work with later versions, TSAPI provides version control.

When an application opens a stream, it provides a list of the TSAPI versions that it will accept. CentreVu Telephony Services will open the stream using the latest version that all components support. MERLIN LEGEND and MERLIN MAGIX CTI provide TSAPI version 2.

The *Telephony Services Application Programming Interface (TSAPI)* specification has step-by-step procedures in Chapter 4 for requesting TSAPI versions and determining the version that TSAPI will supply (when a request indicates support for multiple versions.)

Private Data Version Control

Just as TSAPI evolves over time, so do switch vendors' private data libraries. When an application opens a stream, it also specifies the vendor and versions of the private data libraries that it supports.

When an application needs to obtain private data on a stream, it requests private data from a specific vendor (and acceptable versions of that vendor's private data) in the ***acsOpenStream( )*** request. The *Request Syntax* section of the ***acsOpenStream( )*** description contains a code fragment that requests MERLIN LEGEND or MERLIN MAGIX private data. When an application receives the ***ACSOpenStreamConfEvent***, the private data arriving with that event gives the vendor and version of the private data that will arrive on the stream.

The *Telephony Services Application Programming Interface (TSAPI)* specification has step-by-step procedures in Chapter 4 for requesting private data vendors and/or versions as well as determining from the response the vendor and version that TSAPI will supply.

MERLIN LEGEND CTI provides private data version 1. MERLIN MAGIX CTI provides private data versions 1-3.

Migration from MERLIN LEGEND Private Data Version 1 to MERLIN MAGIX Private Data Version 2 or 3

An existing MERLIN LEGEND CTI application that uses private data version 1 will work in a MERLIN MAGIX environment without any changes. without any changes, will work in a MERLIN MAGIX CTI environment. However, the application cannot open a private data version 2 or 3 interfaces and access any of the private data version 2 or 3 features. To migrate an existing private data version 1 application (i.e., MERLIN LEGEND CTI) into the private data version 2 or 3 environment (i.e., MERLIN MAGIX CTI) the changes shown in Table 3-2 are required.

- The list of Protocol Data Units (PDUs) or structure members in column one represents the original private data version 1 code that is affected by the private data version 2 or 3 interface.
- If you need to recompile an application written to the private data version 1 interface using the header files from the private data version 2 or 3 interface, you must change the PDUs or structure members listed in column one in your code to the associated name listed in column two (i.e., The "ML" portion of the name is changed to "MLV1" for the PDUs while “v1” is prepended in the case of structure members).
- The PDU code names or structure members listed in column three are identical to the original private data version 1 code names; however, their definitions are changed in the header files for the private data version 2 or 3 interface.

NOTE:
The private data library has a convention whereby PDU names for the most recent private data version are always "unqualified," that is, the names do not contain any indication of a particular private data version. When a new version of an existing PDU is introduced, the new PDU assumes the name of the old PDU, and the name of the old PDU is changed to reflect the last private data version for which it was valid. The same naming convention is used when introducing a new version of an existing data type or structure member.

Table 3-2. Migration of Structure Member and PDU Names from Private Data Version 1 to Private Data Version 2 or 3

Original Private Data Version 1 PDU or Structure Member Name	Required Changes to Private Data Version 1 Names for Private Data Version 2 or 3 Interface	New Private Data Version 2 or 3 PDU or Structure Member Name
ML_DELIVERED MLDeliveredEvent_t deliveredEvent	MLV1_DELIVERED MLV1DeliveredEvent_t v1deliveredEvent	ML_DELIVERED MLDeliveredEvent_t deliveredEvent
ML_ESTABLISHED MLEstablishedEvent_t establishedEvent	MLV1_ESTABLISHED MLV1EstablishedEvent_t v1establishedEvent	ML_ESTABLISHED MLEstablishedEvent_t establishedEvent

Querying for Available Services

An application may use the ***acsEnumServerNames( )*** service to obtain a list of advertised service names. A PBX driver registers one or more physical CTI links as an advertised service. An application may open a stream to one or more of these advertised services.

The presence of a service name in the response indicates only that the service is registered, not that it is operational.

An application does not have to have an open stream to call ***acsEnumServerNames( )***.

Querying Login and Password Requirements

An application that needs to operate with multiple server operating systems may use the ***acsQueryAuthInfo( )*** service to determine the structure of the login and password information that it must supply to open a stream to a given advertised service.

Querying for Supported TSAPI Services and Events

An application may use the ***cstaGetAPICaps( )*** service to determine the CSTA services and events that a given stream provides. The ***CSTAGetAPICapsConf-Event*** service contains an entry for each CSTA service and event.

Querying for Devices

An application may use the ***cstaGetDeviceList( )*** service to obtain a list of devices that the Telephony Services Security Database permits it to control, monitor, query, or route on a given stream.

NOTE:
No devices will be returned if the Telephony Services is installed without the optional Telephony Services Database. If an application relies upon the ***cstaGetDeviceList( )*** service to obtain a list of devices, then the application vendor should indicate in their documentation that installation of the Telephony Services Database is required.

NOTE:
Although an application may be given permissions for various operations on various devices in the Telephony Services Security Database, MERLIN LEGEND and MERLIN MAGIX switches do not support any TSAPI routing, call/call monitoring or call/device monitoring services.

Querying for Call/Call Monitor Support

An application may use the ***cstaQueryCallMonitor( )*** service to determine if the Telephony Services Security Database permits it to do call/call monitoring on a given stream.

NOTE:
Although an application may be given permissions for call/call monitoring in the Telephony Services Security Database, the MERLIN LEGEND and MERLIN MAGIX switches do not support TSAPI call/call monitoring.

Client Library TSAPI Functions

The TSAPI client libraries provide the TSAPI functions and events shown in Table 3-3 to applications in all switch environments, including the MERLIN LEGEND or MERLIN MAGIX switch. Refer to the *Telephony Services Application Programming Interface (TSAPI)* for further details. Since these functions are documented in the *Telephony Services Application Programming Interface (TSAPI)*, they are not described in this guide.

Table 3-3. Client Library TSAPI Functions and Confirmation Events

Client Library TSAPI Functions and Confirmation Events
Ö acsGetEventBlock( )
Ö acsGetEventPoll( )
Ö acsGetFile( )
Ö acsSetESR( )
Ö acsEventNotify( )
Ö acsFlushEventQueue
Ö acsEnumServerNames
Ö acsQueryAuthInfo( )
Ö cstaGetDeviceList( ) & CSTAGetDeviceListConfEvent
Ö cstaQueryCallMonitor( ) & CSTAQueryCallMonitorConfEvent

acsAbortStream( )

The abort stream service terminates any CTI services in progress on a specified stream, shuts down the stream, and frees all stream resources in a single operation.

The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver terminates all CTI operations in progress on ***acsHandle***. When the ***acsAbortStream*** function call returns, the stream is aborted. The application does not receive a confirmation event. Once a stream is aborted, the application will not receive:

- confirmation events for outstanding requests on the stream
- call events for monitors that were in progress on the stream

Aborting a stream has no effect on call processing (or on call processing requests that have already been made). Thus, the MERLIN LEGEND or MERLIN MAGIX switch will not take any special action on any call control requests that may be outstanding on the aborted stream. The MERLIN LEGEND or MERLIN MAGIX switch will process any pending requests from an aborted stream in the normal way.

If aborting the stream terminates any device monitors, the application receives a ***CSTAMonitorEndedEvent*** for those device monitors.

Service Request Parameters

Table 3-4. acsAbortStream( ) Request Parameters

acsHandle	handle of the ACS stream to be aborted
privateData	NULL, not used in this service request

Return Values

Table 3-5. acsAbortStream( ) Return Values

zero	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier

Confirmation Event

There is no confirmation event for the abort service. When the function call returns, Telephony Services has aborted the stream and released all the stream resources.

Syntax

```
acsAbortStream (ACSHandle_t      acsHandle,     /* INPUT */
                PrivateData_t    *privateData); /* INPUT */
```

acsCloseStream( )

acsCloseStream( ) closes a Telephony Services stream.

The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver will close the stream and send an ***ACSCloseStreamConfEvent*** to the application. Once an application receives an ***ACSCloseStreamConfEvent*** for a stream, it will not receive:

- a confirmation event for any outstanding service requests on that stream
- any further call events for monitors that were in progress on that stream

Thus, the last event that the closing application receives on the stream is the confirmation of the stream close request, ***ACSCloseStreamConfEvent***. Since a stream close has no effect on call processing (or on call processing requests that have already been made), the MERLIN LEGEND or MERLIN MAGIX switch will not take any special action relating to any call control requests that may be outstanding on the closed stream. The MERLIN LEGEND or MERLIN MAGIX switch will process and respond to any such outstanding requests in the normal way.

CAUTION:
A stream remains open until the application receives the ***ACSCloseStreamConfEvent*** *on that stream. When an application uses* ***acsCloseStream( )*** *to close a stream, it must continue to receive events for that stream until it receives the* ***ACSCloseStreamConfEvent****. If an application fails to do this, the system may not immediately release all of the stream resources.*

If closing the stream terminates any device monitors, the application receives a ***CSTAMonitorEndedEvent*** for those device monitors.

Service Request Parameters

Table 3-6. acsCloseStream( ) Request Parameters

acsHandle	handle of the ACS stream to be closed
invokeID	identifies this request within the stream
privateData	`NULL`, not used in this service request

Return Values

Table 3-7. acsCloseStream( ) Return Values

zero or positive value	Success
`ACSERR_BADHDL`	***acsHandle*** not a valid stream identifier

Confirmation Event - *ACSCloseStreamConfEvent*

Table 3-8. ACSCloseStreamConfEvent Parameters

acsHandle	handle for ACS stream that was closed
eventClass	`ACSCONFIRMATION`
eventType	`ACS_CLOSE_STREAM_CONF`
invokeID	from ***acsCloseStream( )*** service request

Request Syntax

```
acsCloseStream (ACSHandle_t      acsHandle,     /* INPUT */
                InvokeID_t       invokeID,      /* INPUT */
                PrivateData_t    *privateData); /* INPUT */
```

Confirmation Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      ACSConfirmationEvent acsConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union {
      ACSCloseStreamConfEvent_t     acsclose;
   } u;
} ACSConfirmationEvent;

typedef struct ACSCloseStreamConfEvent_t {
   Nulltype       null;
} ACSCloseStreamConfEvent_t;
```

acsOpenStream( )

acsOpenStream( ) opens a Telephony Services stream to the advertised service ***serverID***.

acsOpenStream( ) initializes all stream data structures necessary for the client to use MERLIN LEGEND or MERLIN MAGIX CTI services. All existing TSAPI stream guidelines apply.

MERLIN LEGEND and MERLIN MAGIX CTI support TSAPI version 2 only.

The ***acsOpenStream( )*** function call returns the stream handle in ***acsHandle***.

Service Request Parameters

Table 3-9. acsOpenStream( ) Request Parameters

acsHandle	return parameter - When a stream is successfully opened, this parameter contains the handle of the ACS stream.
invokeIDType	specifies whether the application supplies invokeID values or the TSAPI library generates invokeID values.
invokeID	identifies this request within the stream.
streamType	ST_CSTA
serverID	advertised CTI link service
loginID	user's login ID
passwd	user's password
applicationName	application name (for reporting and tracking). May be null.
acsLevelReq	not used
apiVer	"TS2" (string indicating TSAPI version 2)
sendQSize	library queues this number of application-to-switch service requests. 0 indicates default library size.
sendExtraBufs	Number of additional buffers TSAPI allocates for the send queue.
recvQSize	library queues this number of switch-to-application events and confirmations. 0 indicates default library size.
recvExtraBufs	Number of additional buffers TSAPI allocates for the receive queue.

privateData If an application does not desire private data on the stream, set to `NULL`. To receive MERLIN LEGEND or MERLIN MAGIX private data, set the `vendor` field in the `privateData` structure to the null terminated string `"VERSION"`. Set the data field to contain the one-byte manifest constant `PRIVATE_DATA_ENCODING`. Use the private library function ***mlMakeVersionString( )*** as shown in Private Data Request Syntax that follows to set the private data version.

Return Values

Table 3-10. acsOpenStream( ) Return Values

zero or positive value	Stream opened. The parameter ***acsHandle*** contains the handle for ACS stream.
`ACSERR_APIVERDENIED`	***apiVer*** not supported
`ACSERR_BADPARAMETER`	One or more of the parameters is invalid
`ACSERR_NODRIVER`	No TSAPI client library found or installed
`ACSERR_NOSERVER`	***serverID*** not available
`ACSERR_NORESOURCE`	Insufficient resources to open ACS stream

Confirmation Event - *ACSOpenStreamConfEvent*

The private data that arrives with the ***ACSOpenStreamConfEvent*** indicates the vendor and version of the private data for the opened stream. If private data will arrive on a stream, the `data` field in the `PrivateData_t` structure will contain the one-byte discriminator `PRIVATE_DATA_ENCODING` followed by an ASCII string giving the vendor and version of the private data. When the stream supplies MERLIN LEGEND or MERLIN MAGIX private data, this string matches the `ML_VENDOR_STRING` constant in the header file <mlpriv.h>.

Table 3-11. ACSOpenStreamConfEvent Parameters

acsHandle	handle for ACS stream that was aborted
eventClass	ACSCONFIRMATION
eventType	ACS_OPEN_STREAM_CONF
invokeID	from ***acsOpenStream( )*** service request
apiVer	TSAPI version in use. The MERLIN LEGEND PBX driver and MERLIN MAGIX PBX driver provide only TSAPI version 2. This parameter contains "ST2".
libVer	client library version in use
tsrvVer	Tserver version in use
drvrVer	MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver version in use

Request Syntax

```
acsOpenStream (   ACSHandle_t       *acsHandle,       /* RETURN */
                  InvokeIDType_t    invokeIDType,     /* INPUT */
                  InvokeID_t        invokeID,         /* INPUT */
                  StreamType_t      streamType,       /* INPUT */
                  ServerID_t        *serverID,        /* INPUT */
                  LoginID_t         *loginID,         /* INPUT */
                  Passwd_t          *passwd,          /* INPUT */
                  AppName_t         *applicationName, /* INPUT */
                  Level_t           acsLevelReq,      /* INPUT */
                  Version_t         *apiVer,          /* INPUT */
                  unsigned short    sendQSize,        /* INPUT */
                  unsigned short    sendExtraBufs,    /* INPUT */
                  unsigned short    recvQSize,        /* INPUT */
                  unsigned short    recvExtraBufs,    /* INPUT */
                  PrivateData_t     *privateData);    /* INPUT */
```

Private Data Request Syntax

```
mlMakeVersionString( char *requestedVersion,    /* INPUT */
                     char *supportedVersion);   /* RETURN */

/*
 * EXAMPLE - Code fragment to request MERLIN LEGEND private data
 * version 1 or MERLIN MAGIX private data version 2 or 3.
 */
#include <mlpriv.h>

MLPrivateData_t   privateData;
RetCode_t         rc;

/* Prepare private data buffer for version request */
(void)strcpy(privateData.vendor, "VERSION");
privateData.data[0] = PRIVATE_DATA_ENCODING;

/* Use private library function to prepare a version string
 * for either MERLIN LEGEND private data version 1 or MERLIN
 * MAGIX private data version 2 or 3
 */
if (( rc = mlMakeVersionString("1-3", &(privateData[1]))) > 0 )
{
      /*
       * Al least one of the requested private data versions is
       * supported by the client private data library.
       */
      privateData.length = rc + 2;
}
else
{
      /*
       * None of the requested private data version are supported
       * by the client private data library.
       */
      privateData.length = 0;
 }
```

Confirmation Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      ACSConfirmationEvent acsConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union {
      ACSOpenStreamConfEvent_t      acsopen;
   } u;
} ACSConfirmationEvent;

typedef struct ACSOpenStreamConfEvent_t {
   Version_t   apiVer;
   Version_t   libVer;
   Version_t   tsrvVer;
   Version_t   drvrVer;
} ACSOpenStreamConfEvent_t;
```

ACSUniversalFailureConfEvent

The ***ACSUniversalFailureConfEvent*** can occur in place of a confirmation event for both the ACS and CSTA services. It indicates that an ACS problem occurred while processing the service request. It does not necessarily indicate a failure or loss of the ACS Stream. If the ACS Stream has failed, then the application will receive the ***ACSUniversalFailureEvent***.

Event Parameters

Table 3-12. ACSUniversalFailureConfEvent Parameters

acsHandle	ACS stream on which failure occurred
eventClass	ACSCONFIRMATION
eventType	ACS_UNIVERSAL_FAILURE_CONF
error	TSAPI error value

Error Values

Refer to the TSAPI specification for possible error values.

Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union    {
      ACSConfirmationEvent    acsConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union    {
      ACSUniversalFailureConfEvent_t   failureEvent;
   } u;
} ACSConfirmationEvent;

typedef struct ACSUniversalFailureConfEvent_t {
   ACSUniversalFailure_t   error;
} ACSUniversalFailureEvent_t;
```

ACSUniversalFailureEvent

Telephony Services sends this event when an asynchronous non-CSTA error condition occurs. An application must be able to handle this event on any stream at any time.

If the error condition requires the driver to tear down the ACS stream (certain of these errors do; others do not), then the MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver will tear down the stream as described in the ***acsAbortStream( )*** section. If the failure is one that causes the driver to close the stream, then the application will receive the error `DRIVER_ACSHANDLE_TERMINATION` and an application should take action to clean up its data structures and release any identifiers for this stream.

This event may indicate a loss of the CTI link to the MERLIN LEGEND or MERLIN MAGIX switch.

Event Parameters

Table 3-13. ACSUniversalFailureEvent Parameters

acsHandle	ACS stream on which failure occurred
eventClass	`ACSUNSOLICITED`
eventType	`ACS_UNIVERSAL_FAILURE`
error	ACS error value

Error Values

Refer to the *Telephony Services Application Programming Interface (TSAPI)* specification for possible error values.

Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union    {
      ACSUnsolicitedEvent     acsUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct  {
   union    {
      ACSUniversalFailureEvent_t failureEvent;
   } u;
} ACSUnsolicitedEvent;

typedef struct ACSUniversalFailureEvent_t {
   ACSUniversalFailure_t   error;
} ACSUniversalFailureEvent_t;
```

CSTAUniversalFailureConfEvent

If an application has made a service request that has failed, the application receives the ***CSTAUniversalFailureConfEvent*** in place of a confirmation event for the service request. The ***CSTAUniversalFailureConfEvent*** contains an ***error*** value that gives the reason for the failure. Since the ***CSTAUniversalFailureConfEvent*** may be sent in many contexts, the meaning of the ***error*** value may vary. Each service's manual page lists the ***error*** values that it might return and the meaning for the value in the context of that service.

This event does not indicate a loss of the CTI link to the MERLIN LEGEND or MERLIN MAGIX switch.

Event Parameters

Table 3-14. CSTAUniversalFailureConfEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_UNIVERSAL_FAILURE_CONF`
invokeID	contains the value of the ***invokeID*** parameter that the application supplied in the service request that has failed. This associates the failure with a service request on the stream.
error	contains a value shown on the manual page for the service that failed.

Error Values

The ***error*** parameter contains a value indicating why the corresponding service request has failed. The pages describing service requests list the possible values of the ***error*** parameter and their meanings in the context of that service.

Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union {
      CSTAUniversalFailureConfEvent_t  universalFailure;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAUniversalFailureConfEvent_t {
   CSTAUniversalFailure_t error;
} CSTAUniversalFailureConfEvent_t;
```

cstaGetAPICaps( )

Applications use the ***cstaGetAPICaps( )*** query to obtain environment information. The resulting ***CSTAGetAPICapsConfEvent*** lists the supported TSAPI services and events. Private data in the confirmation indicates the MERLIN LEGEND or MERLIN MAGIX switch release, as indicated in Table 3-15.

Table 3-15. CSTAGetAPICapsConfEvent Private Data

MERLIN LEGEND Release 5.0	`5.0`
MERLIN LEGEND Release 5.1	`5.1`
MERLIN LEGEND Release 6.0	`6.0`
MERLIN LEGEND Release 6.1	`6.1`
MERLIN LEGEND Release 7.0	`7.0`
MERLIN MAGIX Release 1.0	`MAGIX 1.0`
MERLIN MAGIX Release 1.5	`MAGIX 1.5`
MERLIN MAGIX Release 2.0	`MAGIX 2.0`
MERLIN MAGIX Release 2.1	`MAGIX 2.1`
MERLIN MAGIX Release 2.2	`MAGIX 2.2`
Unrecognized Switch Release	`Version Unknown`

NOTE:
If a stream is opened to an MLPD on NetWare, the private data string in the confirmation event contains the string "`ML50`" regardless of the switch version.

NOTE:
The ***CSTAGetAPICapsConfEvent*** does not distinguish between providing events for local monitored stations and trunk connections. The ***CSTAGetAPICapsConfEvent*** indicates that the MERLIN LEGEND or MERLIN MAGIX switch provides certain events. Programmers must understand the limitation in the ***CSTAGetAPICapsConfEvent*** and not program applications to expect events for far-end parties on trunk calls.

Service Request Parameters

Table 3-16. cstaGetAPICaps( ) Request Parameters

acsHandle	handle of ACS stream for this service request
invokeID	identifies this service request within the stream

Return Values

Table 3-17. cstaGetAPICaps( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** not a valid stream identifier

Confirmation Event - *CSTAGetAPICapsConfEvent*

For MELRIN LEGEND (Release 5.0 and later) and MERLIN MAGIX Release 1.0, the ***CSTAGetAPICapsConfEvent*** returns positive values for the following `getAPIcaps` structure elements:

- `answerCall`
- `clearConnection`
- `conferenceCall`
- `conferencedEvent`
- `connectionClearedEvent`
- `consultationCall`
- `deliveredEvent`
- `establishedEvent`
- `heldEvent`
- `holdCall`
- `makeCall`
- `monitorDevice`
- `monitorEnded`
- `monitorStop`
- `networkReachedEvent`
- `retrieveCall`
- `retrievedEvent`
- `serviceInitiatedEvent`
- `transferCall`
- `transferredEvent`.

The driver returns zero values (not supported) for all other elements.

For MERLIN MAGIX Release 1.5, the ***CSTAGetAPICapsConfEvent*** returns positive values for the same `getAPIcaps` structure elements as MERLIN MAGIX Release 1.0, plus the following:

- `divertedEvent`
- `loggedOffEvent`
- `loggedOnEvent`
- `queuedEvent`
- `setAgentState`
- `workNotReadyEvent`.

The driver returns zero values (not supported) for all other elements.

For MERLIN MAGIX Release 2.0, the ***CSTAGetAPICapsConfEvent*** returns positive values for the same `getAPIcaps` structure elements as MERLIN MAGIX Release 1.5, plus the following:

- `deflectCall`
- `doNotDisturbEvent`
- `escapeService`
- `notReadyEvent`
- `privateEvent`
- `queryAgentState`
- `readyEvent`.

The driver returns zero values (not supported) for all other elements.

For MERLIN MAGIX Release 2.1 and later, the ***CSTAGetAPICapsConfEvent*** returns positive values for the same `getAPIcaps` structure elements as MERLIN MAGIX Release 2.0, plus the following:

- `callInformationEvent`
- `queryDnd`
- `queryMwi`
- `readyEvent`
- `setDnd`
- `setMwi`
- `snapshotDeviceReq`
- `workReadyEvent`.

The driver returns zero values (not supported) for all other elements.

Table 3-18. CSTAGetAPICapsConfEvent Parameters

acsHandle	handle for ACS stream from service request
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_GETAPI_CAPS_CONF`
getAPICaps	structure with element for each service and event

Request Syntax

```
cstaGetAPICaps (  ACSHandle_t acsHandle,  /* INPUT */
                  InvokeID_t invokeID);   /* INPUT */
```

Confirmation Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union    {
      CSTAGetAPICapsConfEvent_t        getAPICaps;
   } u;
} CSTAConfirmationEvent;
```

Private Data Confirmation Event Syntax

```
typedef struct
{
   MLEventType eventType;        // ML_GETAPI_CAPS_CONF
   union    {
      MLGetAPICapsConfEvent_t       getAPICaps;
   } u;
} MLEvent_t;

typedef struct  MLGetAPICapsConfEvent_t {
   char  switchVersion[16];      // specifies switch version
} MLGetAPICapsConfEvent_t;
```

Call Control Services

4

Contents

Contents

Contents

Contents

Contents

Contents

Call Control Services

4

Applications use Call Control Services to control calls at extensions. MERLIN LEGEND and MERLIN MAGIX CTI Call Control services allow an application to:

- Make a call.
- Answer an alerting connection at an extension.
- Place a connection on hold.
- Retrieve a held, held-for-transfer, or held-for-conference connection.
- Clear a connection at an extension (e.g., drop the call from that extension).
- Place a connection on hold-for-transfer and make a consultation call to another party. Information about the original caller (the call on hold) passes in private data to any application monitoring the consultation party's extension. Once an application makes the consultation call, it may conference or transfer the original call with the consultation call.
- Transfer a connection on-hold-for-transfer with an active connection at an extension.
- Conference a connection on-hold-for-conference or on-hold-for-transfer with an active connection at an extension.

Beginning with MERLIN MAGIX Release 2.0, Call Control services also allow an application to deflect an unanswered Calling Group Call to a Calling Group Queue or Agent.

Beginning with MERLIN MAGIX Release 2.1, Call Control services allow an application to deflect an unanswered Call Group Call to any available extension.

Tables 4-1 and 4-2 show the TSAPI Call Control Services and confirmation events that the MERLIN LEGEND and MERLIN MAGIX switches provide. Note that the MERLIN LEGEND and MERLIN MAGIX switches do not provide all of the TSAPI Call Control Services.

Table 4-1. MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) CTI Support for TSAPI Call Control Services

	TSAPI Call Control Service
	cstaAlternateCall( ) & CSTAAlternateCallConfEvent
Ö	cstaAnswerCall( ) & CSTAAnswerCallConfEvent
	cstaCallCompletion( ) & CSTACallCompletionConfEvent
	cstaClearCall( ) & CSTAClearCallConfEvent
Ö	cstaClearConnection( ) & CSTAClearConnectionConfEvent
Ö	cstaConferenceCall( ) & CSTAConferenceCallConfEvent
Ö	cstaConsultationCall( ) & CSTAConsultationCallConfEvent
	cstaDeflectCall( ) & CSTADeflectCallConfEvent
	cstaGroupPickupCall( ) & CSTAGroupPickupCallConfEvent
Ö	cstaHoldCall( ) & CSTAHoldCallConfEvent
Ö	cstaMakeCall( ) & CSTAMakeCallConfEvent
	cstaMakePredictiveCall( ) & CSTAMakePredictiveCallConfEvent
	cstaPickupCall( ) & CSTAPickupCallConfEvent
	cstaReconnectCall( ) & CSTAReconnectCallConfEvent
Ö	cstaRetrieveCall( ) & CSTARetrieveCallConfEvent
Ö	cstaTransferCall( ) & CSTATransferCallConfEvent

Table 4-2. MERLIN MAGIX Release 2.0 and later CTI Support for TSAPI Call Control Services

	TSAPI Call Control Service
	cstaAlternateCall( ) & CSTAAlternateCallConfEvent
Ö	cstaAnswerCall( ) & CSTAAnswerCallConfEvent
	cstaCallCompletion( ) & CSTACallCompletionConfEvent
	cstaClearCall( ) & CSTAClearCallConfEvent
Ö	cstaClearConnection( ) & CSTAClearConnectionConfEvent
Ö	cstaConferenceCall( ) & CSTAConferenceCallConfEvent
Ö	cstaConsultationCall( ) & CSTAConsultationCallConfEvent
Ö	cstaDeflectCall( ) & CSTADeflectCallConfEvent
	cstaGroupPickupCall( ) & CSTAGroupPickupCallConfEvent
Ö	cstaHoldCall( ) & CSTAHoldCallConfEvent
Ö	cstaMakeCall( ) & CSTAMakeCallConfEvent
	cstaMakePredictiveCall( ) & CSTAMakePredictiveCallConfEvent
	cstaPickupCall( ) & CSTAPickupCallConfEvent
	cstaReconnectCall( ) & CSTAReconnectCallConfEvent
Ö	cstaRetrieveCall( ) & CSTARetrieveCallConfEvent
Ö	cstaTransferCall( ) & CSTATransferCallConfEvent

CAUTION:
When designing an application, be aware that the MERLIN LEGEND and MERLIN MAGIX switches do not support all of the optional TSAPI call control parameters. The pages describing each call control service show all of the TSAPI parameters and indicate those that the MERLIN LEGEND and MERLIN MAGIX switches support.

Sending Call Control Requests and Receiving Confirmations

Each Call Control request has an associated confirmation event. This book presents information about each service's confirmation event under the heading for the service.

An application must receive the confirmation event on the stream where it sent the Call Control request. "Receiving Events" in Chapter 3 describes how applications receive confirmation events.

Confirmations have different meanings for various services. Refer to the manual page for each service when coding applications so as to use the service confirmations properly. In some cases, an application must wait for the corresponding Call Event to ensure that the request was carried out. In general, it is recommended that an application monitor the device it is controlling so that it receives Call Events reflecting the call activity at those devices. Chapter 6 describes the Monitoring Services.

Call Control Request Failures

If the service request fails for some reason, the application will receive a ***CSTAUniversalFailureConfEvent*** in place of the service confirmation. Each service description includes a list of the ***error*** values that the ***CSTAUniversalFailureConfEvent*** may carry for that service as well as the meanings of those values in the context of that service. The description of the ***CSTAUniversalFailureConfEvent*** is found in Chapter 3 as well as in each service description.

Call Control Service Page Format

The pages describing each TSAPI call control service contain the following sections, as appropriate:

Service Name and Description

The service name appears first. A description of that service immediately follows the name.

Some Call control service descriptions state that the MERLIN LEGEND or MERLIN MAGIX switch will leave connections in a certain state under certain conditions. This occurs in the absence of feature interactions (that is, the users do not invoke any features that make completion of the service impossible, such as hanging up). This feature interaction clause is *not* explicitly restated on each page.

Service Request Parameters

A table lists the service request parameters and summarizes their use.

Scenario Diagram

A figure shows the devices, connections, and calls before and after successful service invocation. In the diagrams, squares are devices and are labeled D1, D2, etc. Circles are calls and are labeled C1, C2, etc. Lines are connections and their label identifies the device and the call (for example D1C2 would be the connection of device D1 to call C2.) Table 4-3 shows the symbols used to label connections with their connection state.

Table 4-3. Symbols Used in Call Control Service Scenario Figures

Symbol	Connection State
i	Initiated (the extension is hearing dial tone, is in the process of dialing, or has completed dialing but the call has not yet originated)
a	Alerting (often audible ringing, but not necessarily)
c	Connected
h	Held
ht, hc	Held for Transfer, Held for Conference - these are used when necessary to distinguish these states from Held
q	Queued
*	Any non-null state (the call appears at the device, and may be connected, held, held-for-conference, held-for-transfer)

Return Values

A table lists the return values for the service request.

In all function returns, success values follow the TSAPI rules. If the requesting application generated the ***invokeID*** value, then a successful function call returns zero. If the TSAPI library generates the ***invokeID*** value, then a successful function call returns the value of the ***invokeID***. This is not explicitly re-stated for each service. “Sending TSAPI Requests and Receiving Confirmations” in Chapter 3 describes ***invokeID*** usage in more detail.

Confirmation Event

This section names the TSAPI confirmation event for the service and contains a table describing the confirmation event parameters.

CSTA Universal Failure Confirmation Event Error Values

This section lists error values that the ***CSTAUniversalFailureConfEvent*** may return to an application when a service request fails. Items in all capitals are #defines from the TSAPI header files.

Request Syntax

This section contains C coding information for the service request.

Confirmation Event Syntax

This section contains C coding information for the service’s confirmation event.

Important Feature Interactions

This section describes important interactions between the call control service and MERLIN LEGEND/MERLIN MAGIX switch features.

cstaAnswerCall( )

The ***cstaAnswerCall( )*** service answers an alerting connection (***alertingCall***) at an extension.

The MERLIN LEGEND and MERLIN MAGIX switches do not provide the Answer Call service for any extension that it cannot take off-hook or that is not already off-hook idle. If the extension does not meet these requirements, then an application making a ***cstaAnswerCall( )*** request for the extension will receive a ***CSTAUniversalFailureConfEvent***.

If successful, the effect is the same as if the user had selected and answered the alerting appearance.

- If the extension is on-hook, the ***alertingCall*** alerting appearance is preselected and the extension is forced off-hook on speakerphone.
- If there are multiple calls alerting at the extension, then the ***alertingCall*** appearance is preselected and the extension is forced off-hook on speakerphone.
- If the extension is off-hook idle, then the ***alertingCall*** appearance is postselected and the connection is made to the speaker, headset, or handset (whichever is off hook).

Service Request Parameters

Table 4-4. cstaAnswerCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
alertingCall	alerting connection containing both deviceID and callID
privateData	`NULL`, not used for this service request

Scenario Diagram

Figure 4-1 illustrates a successful ***cstaAnswerCall( )*** request where ***alertingCall*** is the connection D1C1.

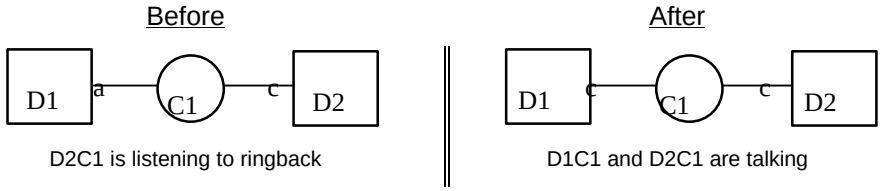


Figure 4-1. cstaAnswerCall( ) Scenario

Return Values

Table 4-5. cstaAnswerCall( ) Return Values

zero or positive value	Success
`ACSERR_BADHDL`	***acsHandle*** is not a valid stream identifier
`ACSERR_STREAM_FAILED`	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTAAnswerCallConfEvent*

The ***CSTAAnswerCallConfEvent*** indicates that the switch has accepted the request, validated the parameters, and signaled the extension to answer the call. Application(s) monitoring the extension will receive a ***CSTAEstablishedEvent*** when the extension connects to the call.

Table 4-6. CSTAAnswerCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_ANSWER_CALL_CONF`
invokeID	identifies service request within stream
privateData	`NULL`, no private data present

CSTA Universal Failure Confirmation Event Errors

If the ***alertingCall*** connection cannot be answered, the MERLIN LEGEND and MERLIN MAGIX switches return one of the errors below. For all ***error*** values except `GENERIC_UNSPECIFIED`, the MERLIN LEGEND or MERLIN MAGIX switch leave the ***alertingCall*** connection in the state that it was in before the switch processed the ***cstaAnswerCall( )*** request. `GENERIC_UNSPECIFIED` will, in most instances, also leave the ***alertingCall*** connection in its initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaAnswerCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when:

- callID in ***alertingCall*** is not present on a supported button type at the deviceID in ***alertingCall***:
  - — For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), only SA buttons are supported; SSA, Coverage, Line and Pool buttons are not supported.
  - — Beginning with MERLIN MAGIX Release 2.0, SA, Coverage, Line and Pool buttons are supported; only SSA buttons are not supported.
- The ***alertingCall*** connection could not be answered for some reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***alertingCall*** is not valid. Some possible reasons are:

- No callID in alertingCall.
- The callID in alertingCall does not exist in the MERLIN LEGEND or MERLIN MAGIX switch.
- the callID in alertingCall is not present at the deviceID in alertingCall.
- Invalid deviceID in alertingCall. One of the following may have occurred:
  - deviceID is unknown or has a null value.
  - deviceID is configured as a QCC.
  - deviceID is configured as a Single Line Set.
- The deviceID in ***alertingCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The application supplied a dynamic device identifier (the MERLIN LEGEND and MERLIN MAGIX switches do not use dynamic device identifiers).

`INVALID_OBJECT_STATE` – The ***alertingCall*** connection is a valid connection identifier (the call is present at the extension) and one of the following conditions occurred:

- The callID in ***alertingCall*** is not alerting (it may have been answered).
- The deviceID in ***alertingCall*** is active on another call.
- The deviceID in ***alertingCall*** is Responding, but is not in Normal Mode.
- Answering ***alertingCall*** would disrupt some activity already in progress at the extension.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***cstaAnswerCall( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`RESOURCE_BUSY` – A needed resource is not in service. Possible causes include:

- The switch is processing another TSAPI request for the extension in ***alertingCall***. Services such as ***cstaMakeCall( )*** and ***cstaConsultationCall( )*** may be in progress when a ***cstaAnswerCall( )*** request arrives.

`REQUEST_TIMEOUT_REJECTION` – The MERLIN LEGEND PBX or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server, MERLIN LEGEND PBX driver, or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaAnswerCall (ACSHandle_t   acsHandle,     /* INPUT */
            InvokeID_t        invokeID,      /* INPUT */
            ConnectionID_t    *alertingCall, /* INPUT */
            PrivateData_t     *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAAnswerCallConfEvent_t        answerCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAAnswerCallConfEvent_t {
   Nulltype    null;
} CSTAAnswerCallConfEvent_t;
```

Important Feature Interactions

An application must receive a ***CSTADeliveredEvent*** for a connection prior to using ***cstaAnswerCall( )*** to answer the connection because the ***CSTADeliveredEvent*** provides the application with the connection identifier that it must use to answer the call. Certain feature interactions may cause calls at buttons to transition into states that are outside of the TSAPI model (particularly associative and bridged states) that will prevent the application from receiving a ***CSTADeliveredEvent.*** In such a case, the application will not be able to use ***cstaAnswerCall( )*** to answer an alerting call. Refer to the scenarios in Chapter 12 for detailed information.

Barge-In

A Barge-In call arriving at an idle extension with Do Not Disturb enabled will alert; an application may use ***cstaAnswerCall( )*** to answer such a call.

A Barge-In at an extension active on a call does not alert (the Barge-In call merges with the active call.) An application may not use ***cstaAnswerCall( )*** to answer such a Barge-In call.

Bridged Appearances (SSA and DFT buttons)

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may *not* use ***cstaAnswerCall( )*** to answer an alerting call on an SSA or DFT button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaAnswerCall( )*** to answer an alerting call on a DFT button, but may *not* use ***cstaAnswerCall( )*** to answer an alerting call on an SSA button.

Call Pickup

An application cannot use ***cstaAnswerCall( )*** to pick up a call.

Callback Queuing (CBQ)

When the originator of a call invokes the Callback Queuing (CBQ) feature and hangs up or places the call on associative hold, the MERLIN LEGEND and MERLIN MAGIX switches will priority alert the originator when the destination is available.

An application can not use the ***cstaAnswerCall( )*** service to answer the CBQ alert at the originator.

When the MERLIN LEGEND and MERLIN MAGIX switches place the callback call to the destination, an application may use ***cstaAnswerCall( )*** to answer the alerting call at the destination (subject to extension and button type restrictions.)

Camp On Return

An application may use ***cstaAnswerCall( )*** to answer an alerting Camp On return call.

Coverage

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaAnswerCall( )*** to answer an alerting call on a Primary, Secondary or Group Coverage button.

Prior to MERLIN MAGIX Release 2.0, an application can not use ***cstaAnswerCall( )*** to answer an alerting call on a Primary, Secondary or Group Coverage button.

Direct Inward Dial (DID)

An application may use ***cstaAnswerCall( )*** to answer either assigned or unassigned DID calls alerting on an SA button.

Do Not Disturb (DND)

With a small number of exceptions, calls do not alert at an extension with DND active, so an application cannot use ***cstaAnswerCall( )*** to answer calls at such an extension.

Barge-In calls will alert at an extension with DND active (see ***Barge-In*** above). An application may use ***cstaAnswerCall( )*** to answer such a call.

When an extension has DND active and a call arrives on the DFT, the call alerts visually (but not audibly). Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaAnswerCall( )*** to answer such a call.

If a coverage receiver calls the sender with DND active, the call will alert on an SA button, so an application may use ***cstaAnswerCall( )*** to answer such a call.

DGC

An application may use ***cstaAnswerCall( )*** to answer an alerting DGC call on an SA button.

Forward On Busy

An application monitoring a station that receives a Forward On Busy call may use ***cstaAnswerCall( )*** to answer the forwarded call.

Forward/Follow Me

When an extension forwards a call, the call may alert at the forwarding extension before the call forwards. Whether the call alerts there, and the duration of the alerting, depends on which variation of the forwarding feature is active.

If a call alerts at a forwarding extension, an application may use ***cstaAnswerCall( )*** to answer the call at the forwarding extension while the call is alerting at the extension.

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application monitoring an extension where a forwarded call alerts will not receive a ***CSTADeliveredEvent*** for the forwarded call. Since an application monitoring an extension where a forwarded call alerts does not receive a ***CSTADeliveredEvent*** event with a connection identifier for the call, the application cannot use ***cstaAnswerCall( )*** to answer the forwarded call.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring an extension where a forwarded call alerts will receive a ***CSTADeliveredEvent*** for the forwarded call, and may use ***cstaAnswerCall( )*** to answer the call at the forwarding destination.

MLX and 4400-series Headset

An application may use ***cstaAnswerCall( )*** to answer an alerting call at an MLX or 4400-series extension where a headset is in use and the headset auto answer feature is off.

Multiple Call Appearances

When a call alerts on an SA button of a monitored station, and also alerts on a DFT or DPT button at the same station two ***CSTADeliveredEvents*** are generated. In a case where the call on the SA button is a DGC call, the call will be cleared from the SA button if it is not answered within a specific time interval. If the alerting call is cleared from the SA button a ***CSTAConnectionClearedEvent*** is generated. Although the call may continue to alert on the DFT/DPT button the application should assume the call is cleared from all buttons at the station. Only one ***CSTAConnectionClearedEvent*** is generated for a call at a monitored station. Attempts to use ***cstaAnswerCall( )*** to answer the DFT/DPT will fail.

Night Service

An application may use ***cstaAnswerCall( )*** to answer a night service call on an SA button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaAnswerCall( )*** to answer a night service call on a DFT button.

Park Return

An application may use ***cstaAnswerCall( )*** to answer a Park return call alerting on an SA button.

Pools (DPT buttons)

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may *not* use ***cstaAnswerCall( )*** to answer an alerting call on a Pool button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaAnswerCall( )*** to answer an alerting call on a DPT button.

Reminder

An application monitoring an extension where a Reminder Service call alerts will not receive ***CSTADeliveredEvent*** for the reminder call. Since an application monitoring an extension where a reminder call alerts does not receive an event with a connection identifier for the call, the application cannot use ***cstaAnswerCall( )*** to answer the reminder call.

Remote Access

An application may use ***cstaAnswerCall( )*** to answer a remote access call on an SA button.

Ringing Line Preference (RLP)

An application uses the ***alertingCall*** parameter to specify the alerting call that is to be answered. This selection overrides any Ringing Line Preference administered for the answering extension.

Service Observing

An application may use ***cstaAnswerCall( )*** to answer a call at a station that is being observed.

An application may *not* use ***cstaAnswerCall( )*** to answer an observed call at the Service Observer

Single Line Sets

Beginning with MERLIN MAGIX Release 2.0, Single Line Sets may be monitored and controlled. However, an application may *not* use ***cstaAnswerCall( )*** to answer a call at a Single Line Set.

System Access Originate Only

An application may use ***cstaAnswerCall( )*** to answer an alerting call on an SA button of this type. (A call may alert on an SA Originate-Only button for certain types of return calls.)

System Access Ring/Voice

An application may use ***cstaAnswerCall( )*** to answer an alerting call on an SA button of this type.

System Access Voice Announce

An application may not use ***cstaAnswerCall( )*** to answer a voice announce call.

Transfer Return

An application may use ***cstaAnswerCall( )*** to answer an alerting transfer return call on an SA button.

cstaClearConnection( )

The ***cstaClearConnection( )*** service clears the connection ***call***. Specifically, the callID in connection ***call*** is disconnected from the deviceID in connection ***call***. In some cases, this results in the switch tearing down all connections to a call (as it does when one party hangs up on a two-party call).

Both the MERLIN LEGEND and MERLIN MAGIX switches support this service for any connection that is in a state such that the deviceID extension user could go on-hook and drop the connection. This includes an active call at the extension, and calls where the user is hearing ringback, busy, reorder, etc. It does not include held or alerting connections at the extension.

When an application successfully clears a connection:

- If the connection was active on the speakerphone, then the MERLIN LEGEND or MERLIN MAGIX switch hangs up the speakerphone and the extension is on-hook.
- If the connection was active on the handset, then the MERLIN LEGEND or MERLIN MAGIX switch leaves the extension off-hook idle.
- If the connection was active on the headset, then the MERLIN LEGEND or MERLIN MAGIX switch presses the HANGUP button on behalf of the user and leaves the extension off-hook idle and the DSS console LED goes off.

When an application successfully clears a connection, this frees the connectionID associated with the connection ***call***.

Service Request Parameters

Table 4-7. cstaClearConnection( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
call	connection to clear. Must contain deviceID and callID
privateData	NULL, not used for this service request

Scenario Diagram

Figure 4-2 illustrates various ***cstaClearConnection( )*** scenarios where ***call*** is the connection D1C1.

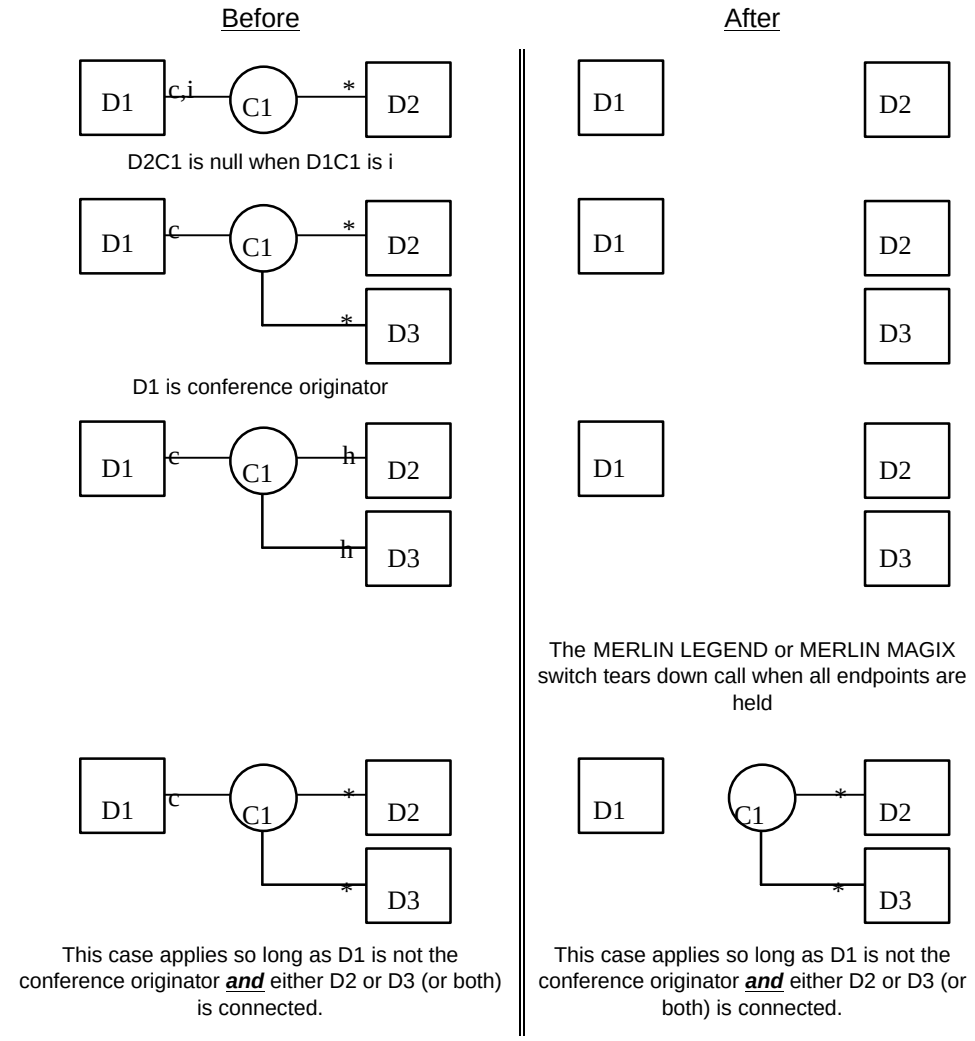


Figure 4-2. cstaClearConnection( ) Scenarios

Return Values

Table 4-8. cstaClearConnection( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event *CSTAClearConnectionConfEvent*

The ***CSTAClearConnectionConfEvent*** indicates that the switch has accepted the request, validated the parameters, and signaled the extension to clear the connection. Application(s) monitoring the extension will receive a ***CSTAConnectionClearedEvent*** when the connection clears.

Table 4-9. CSTAClearConnectionConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_CLEAR_CONNECTION_CONF
invokeID	identifies service request within stream
privateData	NULL, no private data present

CSTA Universal Failure Confirmation Event Error Values

If the ***call*** connection cannot be cleared, the MERLIN LEGEND or MERLIN MAGIX switch returns one of the errors below. For all ***error*** values except GENERIC_UNSPECIFIED, the MERLIN LEGEND or MERLIN MAGIX switch leaves the ***call*** connection in the state that it was in before the switch processed the ***cstaClearConnection( )*** request. GENERIC_UNSPECIFIED will, in most instances, also leave the connections in their initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaClearConnection( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when:

- callID in ***call*** is not present on a supported button type at the deviceID in ***call***.
  - — For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), only SA buttons are supported; SSA, Coverage, Line and Pool buttons are not supported.
  - — Beginning with MERLIN MAGIX Release 2.0, SA, Coverage, Line and Pool buttons are supported; only SSA buttons are not supported.
- The connection ***call*** could not be cleared for some reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***call*** is not valid. Some possible reasons are:

- No callID in ***call***.
- The callID in ***call*** does not exist in the MERLIN LEGEND or MERLIN MAGIX switch.
- the callID in ***call*** is not present at the deviceID in ***call***.
- Invalid deviceID in ***call***. One of the following may have occurred:
  - — deviceID is unknown or has a null value.
  - — deviceID is configured as a QCC.
- The deviceID in ***call*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The application supplied a dynamic device identifier (the MERLIN LEGEND and MERLIN MAGIX switches do not use dynamic device identifiers).

`INVALID_OBJECT_STATE` – The deviceID in ***call*** is not in the correct state. Possible causes include:

- The ***call*** connection is held or alerting (cannot clear a held or alerting connection).
- The deviceID in ***call*** is Responding, but is not in Normal Mode.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***cstaClear-Connection( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`RESOURCE_BUSY` - A needed resource is not in service. Possible causes include:

- The switch is processing another TSAPI request for the extension in ***call***. Services such as ***cstaMakeCall( )*** and ***cstaConsultationCall( )*** may be in progress when a ***cstaClearConnection( )*** request arrives.

REQUEST_TIMEOUT_REJECTION - The MERLIN LEGEND or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server, MERLIN LEGEND PBX Driver or MERLIN MAGIX PBX Driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaClearConnection ( ACSHandle_t      acsHandle,     /* INPUT */
                      InvokeID_t       invokeID,      /* INPUT */
                      ConnectionID_t   *call,         /* INPUT */
                      PrivateData_t    *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAClearConnectionConfEvent_t   clearConnection;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAClearConnectionConfEvent_t {
   Nulltype    null;
} CSTAClearConnectionConfEvent_t;
```

Important Feature Interactions

Bridging (SSA and DFT buttons)

The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches will deny any request to clear a connection appearing on an SSA or DFT button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaClearConnection( )*** to clear a call on a DFT button. An application may not use ***cstaClearConnection( )*** to clear a connection appearing on an SSA button.

Call Screening

An application may use the ***cstaClearConnection( )*** service to clear the connection of a station participating in a screened call.

An application may use the ***cstaClearConnection( )*** service to clear the connection of a station that is screening a call.

Coverage

The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches will deny any request to clear a connection appearing on a Primary, Secondary or Group Coverage button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaClearConnection( )*** to clear a connection appearing on a Primary, Secondary or Group Coverage button.

Direct Facility/Pool Termination (DFT/DPT)

The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches will deny any request to clear a connection appearing on a DFT or DPT button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaClearConnection( )*** to clear a connection appearing on a DFT or DPT button.

Paging

The MERLIN LEGEND and MERLIN MAGIX switches will deny any request to clear a connection appearing at a group page member.

The MERLIN LEGEND and MERLIN MAGIX switches will clear a connection appearing at an SA button at the paging extension.

Service Observing

An application may use ***cstaClearConnection( )*** to clear a connection at a station that is being observed.

An application may use ***cstaClearConnection( )*** to clear an observer's connection to a call that is being observed.

Single Line Sets

The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches will deny a request to clear a connection appearing at a Single Line Set.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaClearConnection( )*** to clear a connection at a Single Line Set that is plugged in.

cstaConferenceCall( )

The ***cstaConferenceCall( )*** service conferences a held-for-conference or held-for-transfer connection and an active connection at a common extension. The deviceID in the ***heldCall*** and ***activeCall*** must specify the common extension.

The ***heldCall*** must be either on hold-for-conference or on hold-for-transfer. Note that the ***cstaConsultationCall( )*** service puts a call on hold-for-transfer, thus ***cstaConferenceCall( )*** can successfully follow ***cstaConsultationCall( )***.

The MERLIN LEGEND and MERLIN MAGIX switches will deny an application request for conference after successful execution of ***cstaHoldCall( )*** and ***cstaMakeCall( )*** services since the ***cstaHoldCall( )*** service does not put the call on hold-for-conference (or on hold-for-transfer).

The MERLIN LEGEND and MERLIN MAGIX switches will permit the interleaving of manual and CTI operations to affect a conference as follows:

- Prerequisite: The user has an active connection and the application has a connectionID for that connection. This may occur when
  - — the user manually answers an incoming call (application has connectionID from Delivered and Established events),
  - — the application uses the ***cstaAnswerCall( )*** service to answer an incoming call,
  - — the application uses the ***cstaMakeCall( )*** service to make a call, or
  - — the user manually places a call to another extension.
- The conference originator manually presses CONFERENCE (or TRANSFER) button. The previously active connection is now held-for-conference (or held-for-transfer.)
- The conference originator becomes connected on a second call either through using the ***cstaMakeCall( )*** service to make a call or answering an incoming call (manually or using the ***cstaAnswerCall( )*** service. The application now has the connectionIDs for the active call and the held call.
- The application makes a ***cstaConferenceCall( )*** request giving the connectionIDs for the held and active calls.

Conferencing must adhere to the MERLIN LEGEND and MERLIN MAGIX switch limits for conferencing: the number of internal parties cannot exceed three; the number of external parties cannot exceed two; and the total number of parties cannot exceed five.

Service Request Parameters

Table 4-10. cstaConferenceCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
heldCall	held connection. Must contain deviceID and callID
activeCall	active connection. Must contain deviceID and callID
privateData	`NULL`, not used for this service request

Scenario Diagram

Figure 4-3 illustrates various ***cstaConferenceCall( )*** scenarios where ***heldCall*** is the connection D1C1 and ***activeCall*** is the connection D1C2.

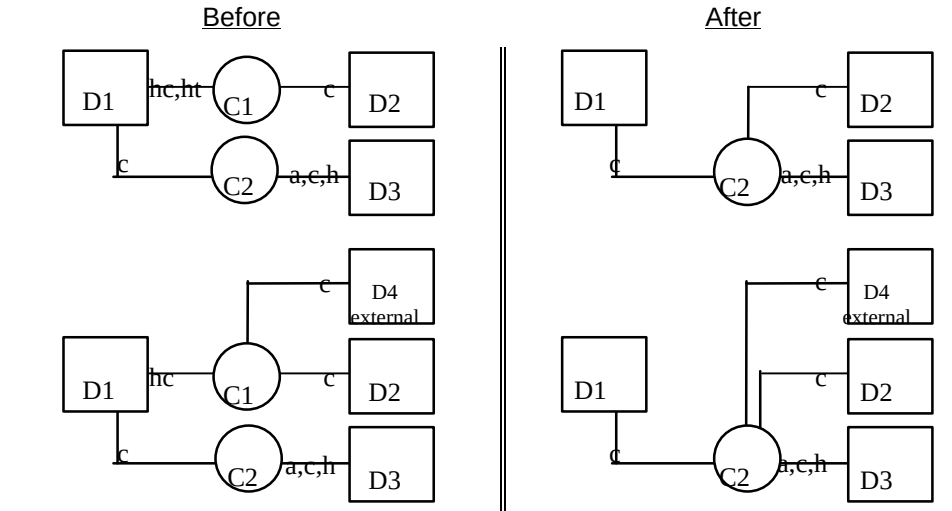


Figure 4-3. cstaConferenceCall( ) Scenarios

Return Values

Table 4-11. cstaConferenceCall( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted.

Confirmation Event - *CSTAConferenceCallConfEvent*

The deviceID in the ***newCall*** parameter in the confirmation event will be the deviceID of the conferencing extension (the common deviceID in ***heldCall*** and ***activeCall***).

In MERLIN LEGEND and MERLIN MAGIX CTI, the callID in the ***newCall*** will be the callID from the ***activeCall***. The application designer should not, however, use this fact in designing an application. As the switch supports more types of extensions and calls in the future, this may not continue to be the case.

The ***CSTAConferenceCallConfEvent*** indicates that the switch has accepted the request, validated the parameters, performed necessary call processing, and signaled the extension to merge the connections. Application(s) monitoring any of the extensions participating in the conference call will receive a ***CSTAConferencedEvent*** when the conference occurs.

Table 4-12. CSTAConferenceCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_CONFERENCE_CALL_CONF`
invokeID	identifies service request within stream
newCall	connectionID containing DeviceID and CallID of the resulting conference call at the conferencing extension
connList	The MERLIN LEGEND and MERLIN MAGIX switches do not provide this optional TSAPI parameter. In the `ConnectionList_t` structure, `count` is set to zero and the `connection` pointer is set to `NULL`.
privateData	`NULL`, no private data present

CSTA Universal Failure Event Error Values

If the ***alertingCall*** and ***heldCall*** cannot be conferenced, MERLIN LEGEND/ MERLIN MAGIX CTI returns one of the errors below. For all ***error*** values except `GENERIC_UNSPECIFIED`, the MERLIN LEGEND and MERLIN MAGIX switches leave the ***alertingCall*** and ***heldCall*** connections in the state that they were in before the ***cstaConferenceCall( )*** request was processed. `GENERIC_UNSPECIFIED` will, in most instances, also leave the connections in their initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaConferenceCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when:

- ***activeCall*** is Senderized.
- ***activeCall*** is in DGC queue.
- It attempts to complete a conference to a busy extension.
- the callID in ***activeCall*** or ***heldCall*** is not present on a supported button type at the extension.
  - — For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), only SA buttons are supported; SSA, Coverage, Line and Pool buttons are not supported.
  - — Beginning with MERLIN MAGIX Release 2.0, SA, Coverage, Line and Pool buttons are supported; only SSA buttons are not supported.

- Processing the request would exceed the limit for the number of parties on a conference call permitted by the MERLIN LEGEND or MERLIN MAGIX switch.
- The ***activeCall*** or ***heldCall*** specifies an observed call at the station of a service observer.
- The ***activeCall*** and ***heldCall*** connections could not be answered for some reason other than the more specific reasons given below.

`GENERIC_OPERATION` – The deviceIDs in ***activeCall*** and ***heldCall*** are not identical (they must be identical since the conference must occur at an extension common to the two calls.)

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***activeCall*** or ***heldCall*** is not valid. Some possible reasons are:

- No callID in ***activeCall*** or ***heldCall***.
- The callID in ***activeCall*** or ***heldCall*** does not exist in MERLIN LEGEND or MERLIN MAGIX switch.
- the callID in ***activeCall*** is not present at the deviceID in ***activeCall***.
- the callID in ***heldCall*** is not present at the deviceID in ***heldCall***.
- Invalid deviceID in ***activeCall*** or ***heldCall***. One of the following may have occurred:
  - deviceID is unknown or has a null value.
  - deviceID is configured as a QCC.
- The deviceID in ***activeCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The deviceID in ***heldCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The application supplied a dynamic device identifier (the MERLIN LEGEND and MERLIN MAGIX switches do not use dynamic device identifiers).

`INVALID_OBJECT_STATE` – The ***activeCall*** and ***heldCall*** connections are valid (the calls are present at the extension) and one of the following conditions occurred:

- The callID in ***activeCall*** is present at the deviceID in ***activeCall***, but the connection is not the active connection at the extension. It is on hold or in some other state.
- The deviceID in ***activeCall*** is Responding, but is not in Normal Mode.
- The callID in ***activeCall*** is a conference call and the deviceID in ***activeCall*** is not the conference originator (only the conference originator can add additional parties to a conference call).

- The callID in ***heldCall*** is present at the deviceID in ***heldCall***, but the connection is not held-for-conference or held-for-transfer. It is in some other state. This occurs if the ***heldCall*** is on regular hold.
- The deviceID in ***heldCall*** is Responding, but is not in Normal Mode.
- The last party added to the call was not added on an SA button and a ***cstaConferenceCall( )*** request was made.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED – Processing the ***cstaConferenceCall( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

RESOURCE_BUSY – A needed resource is not in service. Possible causes include:

- The switch is processing another TSAPI request for the extension in ***alertingCall***. Services such as ***cstaMakeCall( )*** and ***cstaConsultationCall( )*** may be in progress when a ***cstaConferenceCall( )*** request arrives.
- The application requested ***cstaConferenceCall( )*** before the MERLIN LEGEND or MERLIN MAGIX switch sent the confirmation to ***cstaConsultationCall( )***.

REQUEST_TIMEOUT_REJECTION - The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server, MERLIN LEGEND PBX driver, or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaConferenceCall (ACSHandle_t  acsHandle,     /* INPUT */
                    InvokeID_t   invokeID,      /* INPUT */
                    ConnectionID_t  *heldCall,  /* INPUT */
                    ConnectionID_t  *activeCall,/* INPUT */
                    PrivateData_t   *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;
```

```
typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTACConferenceCallConfEvent_t   conferenceCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAConferenceCallConfEvent_t {
   ConnectionID_t    newCall;
   ConnectionList_t  connList;
} CSTAConferenceCallConfEvent_t;
```

Important Feature Interactions

Auto Answer All - AAA (ATL Only) – MERLIN LEGEND and MERLIN MAGIX 1.0 only

The ***cstaConferenceCall( )*** service will successfully complete when the party answering ***activeCall*** has used Auto Answer All to answer ***activeCall***.

Auto Answer Intercom - AAI (ATL Only) - MERLIN LEGEND and MERLIN MAGIX 1.0 only

The ***cstaConferenceCall( )*** service will successfully complete when the party answering ***activeCall*** has used Auto Answer Intercom to answer ***activeCall***.

Call Screening

An application may use the ***cstaConferenceCall( )*** service to complete a conference operation at a station that is participating in a screened call.

An application may not use the ***cstaConferenceCall( )*** service to complete a conference operation at a station that is screening a call.

Call Waiting

Call Waiting may queue the consultation call at the destination and the ***cstaConferenceCall( )*** will successfully complete the conference.

The ***cstaConferenceCall( )*** service will not conference a Call Waiting call with another call.

Conferencing

The ***cstaConferenceCall( )*** service operates the same way as manual conference completion (the second press of the CONFERENCE button). Refer to the *MERLIN LEGEND Advanced Communications System Feature Reference* or *MERLIN MAGIX Integrated System Feature Reference* for complete information.

The ***activeCall*** connection may, itself, be a conference call so long as the conferencing user is the conference originator of that call. When ***activeCall*** is a conference, that conference must appear on at least one SA button on the conferencing extension. Further, the last party added to the ***activeCall*** conference must have been added on an SA button. The ***cstaConferenceCall( )*** request will fail if the last party was added on a non-SA button in MERLIN LEGEND and MERLIN MAGIX Releases 1.0 and 1.5. Beginning with MERLIN MAGIX Release 2.0 this restriction is lifted.

Callback Queuing (CBQ)

The ***cstaConferenceCall( )*** service will not conference a Callback call with another call.

Coverage

The ***cstaConferenceCall( )*** service will successfully conference a call where the far end has the call appearing on a COVER button.

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, if the ***activeCall*** appears only on a Primary, Secondary or Group COVER button at an extension, ***cstaConferenceCall( )*** cannot conference that call on behalf of that extension.

Beginning with MERLIN MAGIX Release 2.0, ***cstaConferenceCall( )*** will successfully conference a call when the ***activeCall*** or ***heldCall*** is on a Primary, Secondary or Group COVER button.

Direct Facility Termination/Personal Lines

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, if the ***activeCall*** appears only on a DFT button at an extension, then the ***cstaConferenceCall( )*** service cannot conference that call on behalf of that extension.

Beginning with MERLIN MAGIX Release 2.0, the ***cstaConferenceCall( )*** service will successfully conference a call when the ***activeCall*** or ***heldCall*** is on a DFT button.

When ***activeCall*** appears on both a DFT button and an SA button at an extension (***activeCall*** may already be a conference) then ***cstaConferenceCall( )*** can conference the call on behalf of that extension.

Group Calling (DGC)

The ***cstaConferenceCall( )*** service will fail if ***activeCall*** is queued (same as manual conference completion operation).

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX Release 1.0 environment, if an application attempts to make a consultation call to a Calling Group as a means to conference another call with that Calling Group, the ***cstaConsultationCall( )*** request will be denied.

Beginning with MERLIN MAGIX Release 1.5, if an application attempts to make a consultation call to a Calling Group as a means to conference another call with that Calling Group, the ***cstaConsultationCall( )*** request will be granted.

Networking

In a MERLIN LEGEND (Release 6.0 and later) and MERLIN MAGIX (Release 1.0) environment, if an application attempts to make a consultation call as a means to conference a call with a station on another MERLIN LEGEND or MERLIN MAGIX switch in the private network, the ***cstaConsultationCall( )*** request will be denied. The user may make the consultation call and the conference call using manual operations at the station set.

Beginning with MERLIN MAGIX Release 1.5, if an application attempts to make a consultation call as a means to conference a call with a station on another MERLIN LEGEND or MERLIN MAGIX switch in the private network, the ***cstaConsultationCall( )*** request will be granted.

Pools

In a MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) environment, if the ***activeCall*** appears only on a DPT button at an extension, then ***cstaConferenceCall( )*** cannot conference that call on behalf of that extension.

Beginning with MERLIN MAGIX Release 2.0, the ***cstaConferenceCall( )*** service will successfully conference a call when the ***activeCall*** or ***heldCall*** is on a DPT button.

Senderized Operation

The ***cstaConferenceCall( )*** service will fail if ***activeCall*** is Senderized (same as manual conference completion operation).

Service Observing

An application may use ***cstaConferenceCall( )*** to complete a conference operation at a station that is being observed.

The ***cstaConferenceCall( )*** service will fail when either ***activeCall*** or ***heldCall*** specifies an observed call at the station of a service observer; a service observer may not use the Conferencing feature with an observed call.

Single Line Sets

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not use ***cstaConferenceCall( )*** to complete a conference operation at a Single Line Set.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***csta-ConferenceCall( )*** to complete a conference operation at a Single Line Set.

System Access (SA)/Shared System Access (SSA) Buttons

The ***cstaConferenceCall( )*** service will successfully conference a call where the far end has the call appearing on an SSA button.

The ***cstaConferenceCall( )*** service will fail when the ***activeCall*** is on an SSA button

Voice Announce

When a consultation call arrives at an extension with Voice Announce, the arriving call is answered on speaker and there is no ***CSTADeliveredEvent*** for the arriving call. There is a ***CSTAEstablishedEvent***. When a conference operation joins the consultation call with the held call, the Voice Announce call clears (monitoring applications will see ***CSTAConnectionClearedEvent***s) and the newly joined call alerts at the consultation destination. Monitors will receive a ***CSTADeliveredEvent*** for the newly alerting call. The connection identifier for the call may contain a call identifier that is different than that of the call answered on Voice Announce.

See Chapter 12 for an example of a Voice Announce event flow.

cstaConsultationCall( )

The ***cstaConsultationCall( )*** service places the active connection on hold-for-transfer at an extension and makes a consultation call from that extension to another device. Specifically, ***activeCall*** specifies the extension making the consultation call and the active call that is to be placed on hold-for-transfer. The ***activeCall*** cannot be a conference call.

To facilitate the programming of consultation scenarios in call center and customer service applications, the MERLIN LEGEND and MERLIN MAGIX switches pass information about the original call on the ***activeCall*** to applications monitoring the ***calledDevice*** in the events resulting from the ***cstaConsultationCall( )*** service. This original call information allows an application monitoring the extension receiving the consultation call to pop a screen at that extension using the original call's information as the consultation call alerts (or is answered). The ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** carry the Original Call information as private data. Refer to the MERLIN LEGEND Private Data Library and Original Call Information" and "MERLIN MAGIX Private Data Library and Original Call Information" in Chapter 2 for more details about how to make use of Original Call Information. See the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** descriptions for further information on the event contents.

The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Release 1.0) switches restrict the ***calledDevice*** to an internal extension. Further, the internal extension must not be a DGC group, Park Zone, Telephone Paging Zone, Listed Directory Number (LDN), modem pool, or a feature access code (* or # code).

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may be any valid number including; a DGC group, Park Zone, Telephone Paging Zone, LDN, modem pool, networked extension, external number or a feature access code. Note, however, that when the ***calledDevice*** contains a feature access code, feature activation is not guaranteed.

The MERLIN LEGEND and MERLIN MAGIX switches will originate the consultation call only on an SA-RING appearance, not on an SA-VOICE appearance.

Once the ***activeCall*** is on hold, the switch will attempt to originate the consultation call in the same manner as it attempts to originate a call for ***cstaMakeCall( )***.

NOTE
If a consultation call has been made and then the application needs to retrieve the held call, the consultation call must be cleared or placed on hold before retrieving the held call.

NOTE:
If the far end of the ***activeCall*** is a local MERLIN LEGEND or MERLIN MAGIX switch extension that has that call held, then the MERLIN LEGEND or MERLIN MAGIX switch returns a ***CSTAUniversalFailureConfEvent*** with a cause of `GENERIC_UNSPECIFIED`. The MERLIN LEGEND or MERLIN MAGIX switch tears down the call. The consultation call is not made.

Service Request Parameters

Table 4-13. cstaConsultationCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
activeCall	active connection. Must contain deviceID and callID
calledDevice	number to call
privateData	`NULL`, not used for this service request

Scenario Diagram

Figure 4-4 illustrates various ***cstaConsultationCall( )*** scenarios where ***activeCall*** is the connection D1C1 and ***calledDevice*** is the device D3.

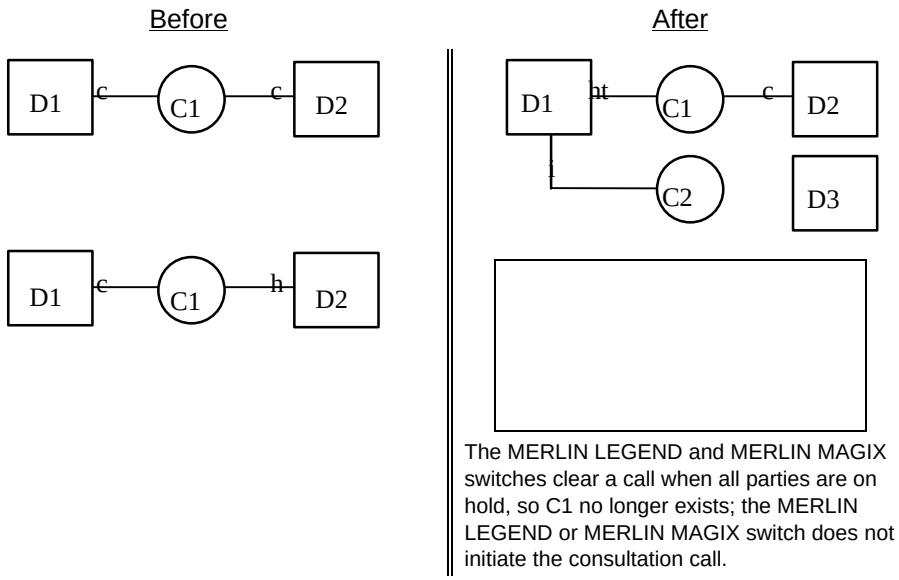


Figure 4-4. cstaConsultationCall( ) Scenarios

Return Values

Table 4-14. cstaConsultationCall( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted.

Confirmation Event - *CSTAConsultationCallConfEvent*

The deviceID in the ***newCall*** parameter in the confirmation event is the deviceID from the ***callingDevice*** request parameter.

The MERLIN LEGEND or MERLIN MAGIX switch sends the ***CSTAConsultationCallConfEvent*** after it has placed the ***activeCall*** on hold and prior to originating the consultation call. A subsequent ***CSTAServiceInitiatedEvent*** will indicate origination of the consultation call.

Table 4-15. CSTAConsultationCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_CONSULTATION_CALL_CONF`
invokeID	identifies service request within stream
newCall	connectionID containing DeviceID and CallID of the consultation call at the extension placing the consultation call
privateData	`NULL`, no private data present

CSTA Universal Failure Event Error Values

If the MERLIN LEGEND or MERLIN MAGIX switch cannot place the active call on hold and originate the consultation call, then the MERLIN LEGEND or MERLIN MAGIX switch returns one of the errors below. For all ***error*** values except `GENERIC_UNSPECIFIED`, the MERLIN LEGEND and MERLIN MAGIX switches leave the ***activeCall*** connection in the state that it was in before the ***cstaConsultationCall( )*** request[1] was processed. `GENERIC_UNSPECIFIED` will, in most instances, also leave the ***activeCall*** connection in its initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaConsultationCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when:

- ***activeCall*** is Senderized.

[1] Of course, if the far end on connection (D2C1) is held, the ***activeCall*** will be torn down if the consultation request proceeded far enough to put connection D1C1 on hold (since all parties on the call were on hold).

- ***activeCall*** connection is a conference call and the device in the ***activeCall*** connection is not the conference originator.
- ***activeCall*** is on hold at the far end connection.
- the callID in ***activeCall*** is not present on a supported button type at the deviceID in ***activeCall***.
  - — For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), only SA buttons are supported; SSA, Coverage, Line and Pool buttons are not supported.
  - — Beginning with MERLIN MAGIX Release 2.0, SA, Coverage, Line and Pool buttons are supported; only SSA buttons are not supported.
- ***activeCall*** specifies a screened call at the station of a Call Screener.
- ***activeCall*** specifies an observed call at the station of a Service Observer.
- The consultation call could not be made for some reason other than those shown below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CALLED_DEVICE` – The deviceID in ***calledDevice*** is invalid. This may be because:

- For releases prior to MERLIN MAGIX Release 1.5, the deviceID specified by ***calledDevice*** is not a local extension number.
- The deviceID in ***calledDevice*** specifies the same extension as the deviceID in ***activeCall***. (An extension may not consult to itself.)

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***activeCall*** is not valid. Some possible reasons are:

- No callID in ***activeCall***.
- The callID in ***activeCall*** does not exist in the MERLIN LEGEND or MERLIN MAGIX switch.
- callID in ***activeCall*** is not present at the deviceID in ***activeCall***.
- Invalid deviceID in ***activeCall***. One of the following may have occurred:
  - — deviceID is unknown or has a null value.
  - — deviceID is configured as a QCC.
- The deviceID in ***activeCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The application supplied a dynamic device identifier (the MERLIN LEGEND and MERLIN MAGIX switches do not use dynamic device identifiers).

INVALID_OBJECT_STATE – The ***activeCall*** connection is a valid connection identifier (the call is present at the extension) and one of the following conditions occurred:

- The ***activeCall*** connection is a conference call and the deviceID specified in ***activeCall*** is the conference originator.
- The callID in ***activeCall*** is present at the deviceID in ***activeCall***, but it is not in the active state.
- The deviceID in ***activeCall*** is active on another call.
- The deviceID in ***activeCall*** is Responding, but is not in Normal Mode.
- There is not an SA button available to place the consultation call on the deviceID specified in ***activeCall***.
- There is already another call on hold-for-transfer or on hold-for-conference at the deviceID specified in ***activeCall***.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED – Processing the ***cstaConsultationCall( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

RESOURCE_BUSY – A needed resource is busy. Possible causes include:

- The switch is processing another TSAPI request for the extension in ***activeCall***. Services such as ***cstaMakeCall( )*** and ***cstaConsultationCall( )*** may be in progress when a ***cstaConsultationCall( )*** request arrives.

REQUEST_TIMEOUT_REJECTION - The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server, MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaConsultationCall (  ACSHandle_t    acsHandle,     /* INPUT */
                        InvokeID_t     invokeID,      /* INPUT */
                        ConnectionID_t *activeCall,   /* INPUT */
                        DeviceID_t     *calledDevice, /* INPUT */
                        PrivateData_t  *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAConsultationCallConfEvent_t  consultationCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAConsultationCallConfEvent_t {
   ConnectionID_t  newCall;
} CSTAConsultationCallConfEvent_t;
```

Important Feature Interactions

Account Code/Forced Account Code

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0 and later), the ***calledDevice*** must be a local extension and should not contain account code information.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may contain an account code. However, the account code will be ignored.

Authorization Code

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** must be a local extension, so it cannot contain an authorization code.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may contain an authorization code.

Automatic Line Selection and Ringing Line Preference

The ***cstaConsultationCall( )*** service selects an origination appearance using the same method as One Touch Transfer described in the One-Touch Transfer With Manual Completion feature. This overrides Automatic Line Selection or Ringing Line Preference administration.

Automatic Route Selection (ARS)

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** must be a local extension, so it cannot contain ARS digits.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may contain ARS digits.

Central Office Trunks

The ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request cannot be a trunk identifier.

Call Screening

An application may use ***cstaConsultationCall( )*** to make a consultation call at a station that is participating in a screened call. However, when ***activeCall*** is placed on hold-for-transfer, the Call Screener will be dropped from the call.

The ***cstaConsultationCall( )*** service will fail when ***activeCall*** specifies a screened call at the station of a Call Screener.

Call Waiting

The ***cstaConsultationCall( )*** service is successful if the consultation call waits at the ***calledDevice***.

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** must be a local extension, so it cannot contain the Call Waiting Pickup code.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may contain the Call Waiting Pickup code.

Callback Queuing (CBQ)

The ***cstaConsultationCall( )*** service is successful if the originating extension has Automatic Callback Queuing enabled.

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** cannot contain the Selective Callback feature activation code.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may contain the Selective Callback feature activation code.

Conferencing

The ***activeCall*** cannot be a conference call.

Coverage

The ***cstaConsultationCall( )*** service is successful whether or not the ***calledDevice*** is a coverage sender or has coverage off.

Prior to MERLIN MAGIX Release 2.0, the ***activeCall*** cannot be on a Primary, Secondary or Group Coverage button.

Beginning with MERLIN MAGIX Release 2.0, the ***activeCall*** can be on a Primary, Secondary or Group Coverage button.

Through DGC Group Coverage, a consultation call to an extension may be delivered to a DGC Group. If ***calledDevice*** is an extension that is out of service (not plugged in), and is a Group Coverage sender with a DGC Group as the Group Coverage receiver, then the consultation call will route to the DGC Group.

Dial Plan

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the deviceID in the ***calledDevice*** is valid if the leading digits are a valid extension in the dial plan. The switch will dial any additional digits entered after the extension.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** is valid as long as it is not the same as the deviceID in ***activeCall*** (i.e., an extension may not consult with itself).

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** must be a local extension, so it may not contain *, #, feature or programming code entries.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** can contain the *, #, feature or programming code entries.

The ***calledDevice*** may contain the operator code of 0.

Direct Facility/Pool Termination

Beginning with MERLIN MAGIX Release 2.0, the ***activeCall*** can be on a DFT or DPT button.

Prior to MERLIN MAGIX Release 2.0, the ***activeCall*** can not be on a DFT or DPT button.

Direct Voice Mail

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it may not contain the feature code for a direct call to Voice Mail.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may contain the feature code for a direct call to Voice Mail.

Do Not Disturb

The ***cstaConsultationCall( )*** service is successful whether or not the ***calledDevice*** has Do Not Disturb enabled.

End-Of-Dialing (Loop and Ground Start Trunks)

When the active call (C1 in the diagrams above) is connected on a Loop or Ground Start trunk, the users may be able to talk before the switch timers transition the connection to End-of-Dialing. If an application requests ***cstaConsultationCall( )*** before the transition to End-of-Dialing on this connection, the request will fail.

External Numbers

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it cannot be an external number.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** can be an external number.

Far End Disconnect

If the far-end party on the ***activeCall*** hangs up while the MERLIN LEGEND or MERLIN MAGIX switch is processing the ***cstaConsultationCall( )*** request, then the MERLIN LEGEND or MERLIN MAGIX switch places the consultation call and returns the ***CSTAConsultationCallConfEvent***. The application will receive call events reflecting the far end disconnect and placement of the consultation call.

Group Calling (DGC)

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so cannot be a DGC group. It may contain the extension of a DGC Group member.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** may contain a DGC group.

Hold

If the far end of the ***activeCall*** is a local extension on hold, then ***cstaConsultationCall( )*** returns `GENERIC_UNSPECIFIED` and ***activeCall*** is cleared at the consulting extension.

Idle Time-outs

If the ***calledDevice*** is not plugged in, the consulting user will hear busy tone and the ***activeCall*** (that is now on hold) remains on hold. After the time-out occurs and the consulting user's extension transitions to off-hook idle, an application may use ***cstaRetrieveCall( )*** to retrieve the ***activeCall*** from the held state. Note that the application need not wait for the time-out to occur. A user would use the ***cstaClearConnection( )*** (via the application) to clear the busy connection at the consulting device and then use ***cstaRetrieveCall( )*** to retrieve the ***activeCall*** from the held state.

Listed Directory Number (LDN)

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it cannot be the LDN.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request can be the LDN.

Modem Pool

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it cannot be a Modem Pool.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request can be a Modem Pool.

Networking

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it cannot be the extension number of a station on another switch in the private network. If an application attempts to make a consultation call to a station on another switch in the private network, the ***cstaConsultationCall( )*** request will be denied.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request can be the extension number of a station on another switch in the private network.

One-Touch Transfer with Manual Completion

Attempting the ***cstaConsultationCall( )*** service request is the same as invoking One-Touch Transfer with Manual Completion. The type of button selected is dependent on the type of transfer (i.e. automatic completion). Refer to the *MERLIN LEGEND Advanced Communications System Feature Reference* or *MERLIN MAGIX Integrated System Feature Reference* for further information.

Paging

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it cannot be a paging zone.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request can be a paging zone.

Park

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it cannot be a park zone.

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request can be a park zone.

Since an extension cannot consult to itself, an application cannot use ***cstaConsultationCall( )*** to park a call (an application may not use ***cstaConsultationCall( )*** to transfer a call to itself).

Pool Codes

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release1.0) environment, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request must be a local extension, so it cannot contain a pool dialout code (e.g. 70).

Beginning with MERLIN MAGIX Release 1.5, the ***calledDevice*** parameter in a ***cstaConsultationCall( )*** request can contain a pool dialout code.

Redial

If the ***cstaConsultationCall( )*** service originates a call, then the ***calledDevice*** becomes the Redial number.

Remote Call Forwarding

The ***cstaConsultationCall( )*** service is successful whether or not the ***calledDevice*** has Remote Call Forwarding (delayed or undelayed) enabled.

Restrictions

The restrictions for a station still apply when the ***cstaConsultationCall( )*** service is used.

Save Number Dialed

The user at the extension originating the ***cstaConsultationCall( )*** service may invoke Save Number Dialed and the MERLIN LEGEND or MERLIN MAGIX switch will retain the number in the ***calledDevice*** as the Saved Number Dialed.

Senderization

The ***cstaConsultationCall( )*** service will not be successful if ***activeCall*** is Senderized. Senderization does not correspond to End-Of-Dialing or to a talking phase. Typically senderization stops before or during End-of-Dialing, but this is not guaranteed (for example during ARS digit absorption). This interaction is equivalent to the Transfer With Manual Completion feature operation.

Service Observing

An application may use ***cstaConsultationCall( )*** to make a consultation call at a station that is being observed.

The ***cstaConsultationCall( )*** service will fail when ***activeCall*** specifies an observed call at the station for a service observer.

Shared System Access Buttons

The ***cstaConsultationCall( )*** service will fail when ***activeCall*** appears on a Shared System Access button.

Single Line Sets

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaConsultationCall( )*** to make a consultation call on behalf of a Single Line Set.

Prior to MERLIN MAGIX Release 2.0, the switch will deny a request to make a consultation call at a Single Line Set.

System Access Ring/Voice Option

The consultation call resulting from an invocation of ***cstaConsultationCall( )*** may be either a ring consultation or a voice announce consultation. The result depends on the administration and operation of the One-Touch Transfer With Manual Completion feature.

Transfer

See the feature interaction for One-Touch Transfer With Manual Completion.

Voice Announce

When a consultation call arrives at an extension with Voice Announce, the arriving call is answered on speaker and there is no ***CSTADeliveredEvent*** for the arriving call. There is a ***CSTAEstablishedEvent***. When a transfer or conference operation joins the consultation call with the held call, the Voice Announce call clears (monitoring applications will see ***CSTAConnectionClearedEvent***s) and the newly joined call alerts at the consultation destination. Monitors will receive a ***CSTADeliveredEvent*** for the newly alerting call. The connection identifier for the call may contain a call identifier that is different than that of the call answered on Voice Announce.

Chapter 12 contains an sample event flow for a Voice Announce scenario.

cstaDeflectCall( )

This service is supported beginning with MERLIN MAGIX Release 2.0.

The ***cstaDeflectCall( )*** service redirects an unanswered Calling Group call from one device to another. The call must be a Calling Group call alerting at an agent, at a Delay Announcement Unit, or in a Calling Group queue. In a MERLIN MAGIX Release 2.0 environment the call may be deflected to another Logged In agent or to a Calling Group queue. Beginning with MERLIN MAGIX Release 2.1, these restrictions are lifted and an application may deflect a call to any non-QCC station that is in normal call handling mode and has an available System Access button. A call may be deflected more than one time.

The call may be in the queue for any reason (e.g., a call was made directly to the calling group, or the call was queued as a result of Call Coverage). DGC calls alerting at an extension are eligible for the ***cstaDeflectCall( )*** service. Non-DGC calls for a Calling Group agent (i.e., the called number was the agent's extension) are not eligible for the ***cstaDeflectCall( )*** service.

An alerting DGC call may be deflected to a queue that has no members.

If the call is routed off the local MERLIN MAGIX switch, there will be no events for the call, as event reporting services are not available for non-local extensions.

The MERLIN MAGIX switch supports four possible scenarios for the ***cstaDeflectCall( )*** service:

1. Deflecting a Call From One Calling Group Queue To Another.

   An alerting call in a Calling Group Queue may be redirected to another Calling Group. When the ***cstaDeflectCall( )*** service is used, the call is removed from the original queue and placed at the end of the destination queue. Once redirected, the call is treated as if had originally come into the destination queue. The SMDR record will show the redirected queue and not the original queue. The call will now follow the rules for the new Calling Group.

2. Deflecting a Call From a Calling Group Queue To a Station

   An alerting call in a Calling Group Queue may be redirected to a station. For MERLIN MAGIX Release 2.0, the destination station must be available to receive a DGC call (i.e., logged in and idle), though the station does not need to be a member of a Calling Group.

   Beginning with MERLIN MAGIX Release 2.1, these restrictions are lifted and an application is able to deflect a call to any non-QCC station that is in normal call handling mode and has an available System Access button.

When the ***cstaDeflectCall( )*** service is used, the call is removed from the queue and begins to alert on a System Access button at the destination station. The call continues to ring until it is answered, deflected, or until the far end disconnects. The call appears as an ordinary DGC call; the station display provides the same feedback as if the station were a member of the Calling Group. Any DGC information in the SMDR record will report the *last* group where the call was queued.

3. Deflecting a Call From One Station To Another

   An alerting DGC call at a station may be deflected to another station. In MERLIN MAGIX Release 2.0 the destination station must be available to receive a DGC call (i.e., logged in and idle), though the station does not need to be a member of a Calling Group.

   Beginning with MERLIN MAGIX Release 2.1, these restrictions are lifted and an application is able to deflect a call to any non-QCC station that is in normal call handling mode and has an available System Access button.

   When the ***cstaDeflectCall( )*** service is used, the call is removed from the station where it is alerting and begins to alert on a System Access button at the destination station. The call appears as an ordinary DGC call; the destination station display provides the same feedback as if the station were a member of the last Calling Group where the call was queued. Any DGC information in the SMDR record will report the *last* group where the call was queued.

4. Deflecting a Call From Station To a Calling Group Queue

   An alerting DGC call at a station may be redirected to a Calling Group Queue. When the ***cstaDeflectCall( )*** service is used, the call is removed from the station where it is alerting and placed at the end of the destination queue. Once redirected, the call is treated as if had originally come into the destination queue. The SMDR record will show the redirected queue and not the original queue. The call will now follow the rules for the new Calling Group.

Service Request Parameters

Table 4-16. cstaDeflectCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
deflectCall	alerting connection to be redirected. Must contain DeviceID and CallID
calledDevice	destination device. This must be a local extension or calling group queue
privateData	NULL, not used for this service request

Scenario Diagram

Figure 4-5 illustrates a ***cstaDeflectCall( )*** scenario where ***deflectCall*** is the connection D1C1 and ***calledDevice*** is the device D3.

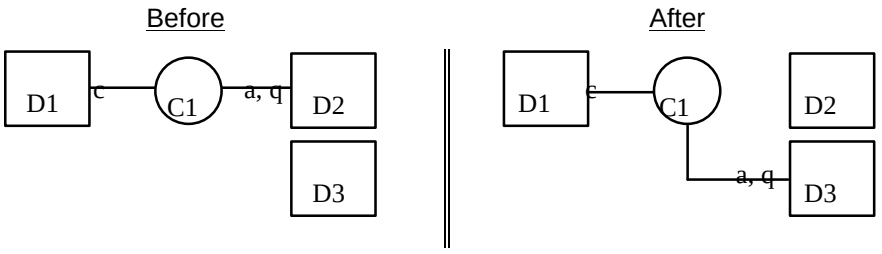


Figure 4-5. cstaDeflectCall( ) Scenarios

Return Values

Table 4-17. cstaDeflectCall( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTADeflectCallConfEvent*

A ***CSTADeflectCallConfEvent*** indicates that the ***deflectCall*** connection has been redirected to ***calledDevice***.

Table 4-18. CSTADeflectCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_DEFLECT_CALL_CONF
invokeID	identifies service request within stream
privateData	NULL, no private data present

CSTA Universal Failure Event Error Values

If the ***deflectCall*** connection cannot be redirected to ***calledDevice***, or ***calledDevice*** is not eligible to receive the call, the MERLIN MAGIX switch returns one of the errors below. For all ***error*** values except GENERIC_UNSPECIFIED, the MERLIN MAGIX switch leaves the ***deflectCall*** in the state that it was in before the switch processed the ***cstaDeflectCall( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaDeflectCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

GENERIC_UNSPECIFIED – An application will receive GENERIC_UNSPECIFIED when the ***deflectCall*** connection could not be redirected to ***calledDevice*** for some reason other than the more specific reasons given below.

RESOURCE_OUT_OF_SERVICE – The CTI link is disconnected or not in service.

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***deflectCall*** is not valid. Some possible reasons are:

- There is no callID in ***deflectCall***.
- The callID in ***deflectCall*** does not exist in the MERLIN MAGIX switch.
- The callID in ***deflectCall*** is not present at the deviceID in ***deflectCall***.
- The deviceID in ***deflectCall*** is not a local extension or Calling Group.
- The deviceID in ***deflectCall*** is configured as a QCC.
- The application supplied a dynamic device identifier (the MERLIN MAGIX switch does not use dynamic device identifiers).

`INVALID_CALLED_DEVICE` – The deviceID in ***calledDevice*** is invalid. This may be because:

- The ***calledDevice*** is an SSA or Calling Group queue.
- The ***calledDevice*** is a QCC or LDN.
- The ***calledDevice*** is the same as the deviceID in ***deflectCall***.

`INVALID_OBJECT_STATE` – The callID in ***deflectCall*** is not a Group Calling call alerting at the specific deviceID in ***deflectCall***.

`GENERIC_STATE_INCOMPATABILITY` – The ***calledDevice*** is a valid extension number, but it is not in the proper state to receive a DGC call (i.e., the extension must be on-hook, logged-in, responding mode, etc.). See the *MERLIN MAGIX Integrated System Feature Reference* for the complete list of conditions that block DGC calls.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***cstaDeflectCall( )*** exceeds the maximum number of outstanding requests permitted at either the PBX driver or the switch.

`RESOURCE_BUSY` – A needed resource is busy. Possible causes include:

- The switch is processing another TSAPI request for the extension in ***deflectCall***. Services such as ***cstaMakeCall( )*** and ***cstaConsultation-Call( )*** may be in progress when a ***cstaDeflectCall( )*** request arrives.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaDeflectCall ( ACSHandle_t    acsHandle,     /* INPUT */
                  InvokeID_t     invokeID,      /* INPUT */
                  ConnectionID_t *deflectCall,  /* INPUT */
                  DeviceID_t     *calledDevice, /* INPUT */
                  PrivateData_t  *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTADeflectCallConfEvent_t deflectCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTADeflectCallConfEvent_t {
   Nulltype    null;
} CSTADeflectCallConfEvent_t;
```

Important Feature Interactions

Bridging

If an application requests the ***cstaDeflectCall( )*** service for a Bridged call, the request is denied with CSTA Universal Failure error `INVALID_OBJECT_STATE`.

Callback Queuing (CBQ)

If an application requests the ***cstaDeflectCall( )*** service for a Callback Queuing call, the request is denied with CSTA Universal Failure error `INVALID_-OBJECT_STATE`.

Calling Information

When a call is deflected to a display station, the information on the station display reflects the last queue or DGC group that the call was in.

Camp-On Return

If an application requests the ***cstaDeflectCall( )*** service for a Camp-On return call, the request is denied with CSTA Universal Failure error `INVALID_-OBJECT_STATE`.

Coverage

If an application requests the ***cstaDeflectCall( )*** service for a call alerting on a Primary, Secondary or Group Cover button, the request is denied with CSTA Universal Failure error `INVALID_OBJECT_STATE`.

The ***cstaDeflectCall( )*** service is successful for a Coverage Call alerting in a DGC queue (i.e., if DGC Group Coverage is in use).

Delay Announcement Unit

The ***cstaDeflectCall( )*** service is successful for a call alerting at a Delay Announcement Unit or a call that has been answered at the Delay Announcement Unit. The `deviceID` component of ***deflectCall*** must contain the DGC queue and not the extension number of the Delay Announcement Unit.

Dial Plan Routing

The ***cstaDeflectCall( )*** service is successful for a call that has been routed to a Calling Group via Dial Plan Routing.

Distinctive Ring

A call that is redirected with the ***cstaDeflectCall( )*** service will receive Distinctive Ringing treatment.

Group Calling

The ***cstaDeflectCall( )*** is successful for an alerting DGC call[2] (either at a member or in the queue).

Listed Directory Number (LDN)

A call may be deflected to a DGC group that overflows to the LDN.

Park Return

If an application requests the ***cstaDeflectCall( )*** service for a Park return call, the request is denied.

---

[2] This could be an internal, external, networked, night service, QCC backup or coverage call

Reminder Service

If an application requests the ***cstaDeflectCall( )*** service for a Reminder Service call, the request is denied.

Transfer Return

If an application requests the ***cstaDeflectCall( )*** service for a transfer return call, the request is denied.

cstaHoldCall( )

The ***cstaHoldCall( )*** service puts the active connection ***activeCall*** on hold (not on hold-for-transfer or hold-for-conference.). The ***callID*** specified in the connection is held at the ***deviceID***. The connection must be in an active state at the extension.

The MERLIN LEGEND and MERLIN MAGIX switches always reserve the connection on all of their extension sets. The ***reservation*** parameter has no effect on the ***cstaHoldCall( )*** processing.

NOTE:
The MERLIN LEGEND and MERLIN MAGIX switches will tear down a call when all internal parties on the call have placed that call on hold. If an application requests ***cstaHoldCall( )*** when all other parties to that call are local extensions with the call on hold, the ***cstaHoldCall( )*** operation will succeed. The last party added to the call is placed on Hold and the MERLIN LEGEND or MERLIN MAGIX switch will then tear down the call. Applications will receive appropriate events as the call is torn down.

Service Request Parameters

Table 4-19. cstaHoldCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
activeCall	active connection. Must contain deviceID and callID
reservation	The MERLIN LEGEND and MERLIN MAGIX switches always reserve the facility for re-use and thus this parameter has no effect.
privateData	`NULL`, not used for this service request

Scenario Diagram

Figure 4-6 illustrates various ***cstaHoldCall( )*** scenarios where ***activeCall*** is the connection D1C1.

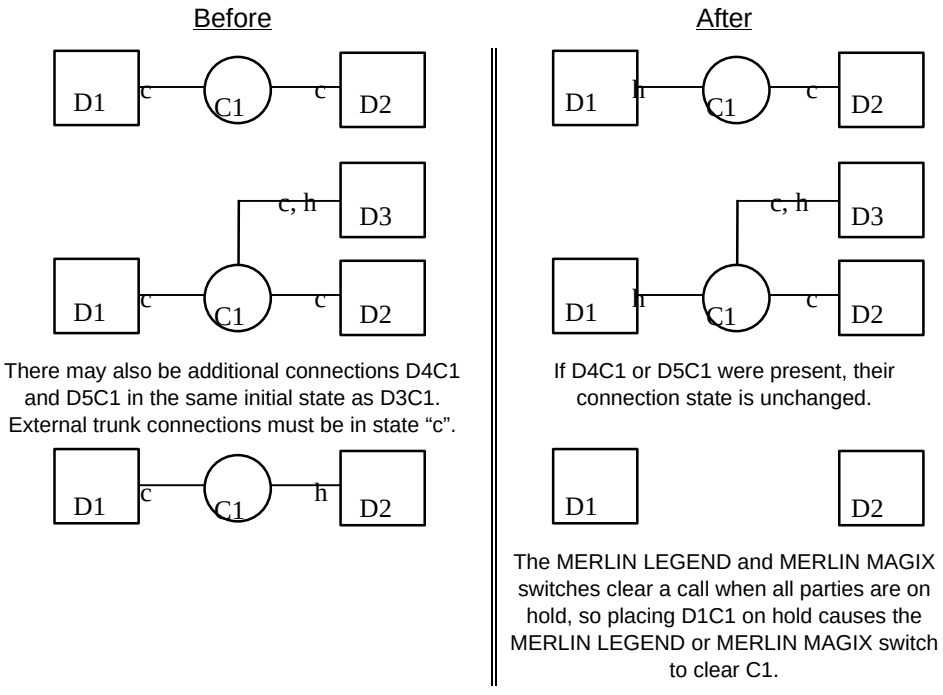


Figure 4-6. cstaHoldCall( ) Scenarios

Return Values

Table 4-20. cstaHoldCall( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTAHoldCallConfEvent*

A ***CSTAHoldCallConfEvent*** indicates that the ***activeCall*** connection has been placed on hold.

Table 4-21. CSTAHoldCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_HOLD_CALL_CONF
invokeID	identifies service request within stream
privateData	NULL, no private data present

CSTA Universal Failure Event Error Values

If the ***activeCall*** connection cannot be held, the MERLIN LEGEND or MERLIN MAGIX switch returns one of the errors below. For all ***error*** values except GENERIC_UNSPECIFIED, the MERLIN LEGEND and MERLIN MAGIX switches leave the ***activeCall*** connection in the state that it was in before the switch processed the ***cstaHoldCall( )*** request. GENERIC_UNSPECIFIED will, in most instances, also leave the ***activeCall*** connection in its initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaHoldCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

GENERIC_UNSPECIFIED – An application will receive GENERIC_UNSPECIFIED when:

- ***activeCall*** specifies an active connection at a 4400D station, but there is already a call on hold, hold-for-conference or hold-for-transfer at that station
- callID in ***activeCall*** is not present on a supported button type at the deviceID in ***activeCall***.
  - For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), only SA buttons are supported; SSA, Coverage, Line and Pool buttons are not supported.
  - Beginning with MERLIN MAGIX Release 2.0, SA, Coverage, Line and Pool buttons are supported; only SSA buttons are not supported.
- ***activeCall*** specifies a screened call at the station of a Call Screener.
- ***activeCall*** specifies an observed call at the station of a Service Observer.
- The ***activeCall*** connection could not be held for some reason other than the more specific reasons listed below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***activeCall*** is not valid. Some possible reasons are:

- No callID in ***activeCall***.
- The callID in ***activeCall*** does not exist in the MERLIN LEGEND or MERLIN MAGIX switch.
- callID in ***activeCall*** is not present at the deviceID in ***activeCall***.
- Invalid deviceID in ***activeCall***. One of the following may have occurred:
  - deviceID is unknown or has a null value.
  - deviceID is configured as a QCC.
- The deviceID in ***activeCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The application supplied a dynamic device identifier (the MERLIN LEGEND and MERLIN MAGIX switches do not use dynamic device identifiers).

`INVALID_OBJECT_STATE` – The ***activeCall*** connection is a valid connection identifier (the call is present at the extension) and one of the following conditions occurred:

- The callID in ***activeCall*** is present at the deviceID in ***activeCall***, but it is not in the active state.
- The deviceID in ***activeCall*** is active on another call.
- The deviceID in ***activeCall*** is Responding, but is not in Normal Mode.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED – Processing the ***cstaHoldCall( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

RESOURCE_BUSY – A needed resource is busy. Possible causes include:

- The switch is processing another TSAPI request for the extension in ***activeCall***. Services such as ***cstaMakeCall( )*** and ***cstaConsultationCall( )*** may be in progress when a ***cstaHoldCall( )*** request arrives.

REQUEST_TIMEOUT_REJECTION - The MERLIN LEGEND or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server, MERLIN LEGEND PBX driver, or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaHoldCall ( ACSHandle_t    acsHandle,     /* INPUT */
               InvokeID_t     invokeID,      /* INPUT */
               ConnectionID_t *activeCall,   /* INPUT */
               Boolean        reservation,   /* INPUT */
               PrivateData_t  *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAHoldCallConfEvent_t holdCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAHoldCallConfEvent_t {
   Nulltype    null;
} CSTAHoldCallConfEvent_t;
```

Important Feature Interactions

4400D Hold

The Hold button on the 4400D sets acts differently than any other set in the system. The first press of the Hold button on an active call places the call on Hold; a second press of the Hold button retrieves the held call. While a call is on hold, the user may make another call (after a switch-hook depression). If an application requests the ***cstaHoldCall( )*** service for an active call at a 4400D set while another call is already on hold, the request is denied.

Call Screening

An application may use the ***cstaHoldCall( )*** service to place a call on hold at a station that is participating in a screened call.

The ***cstaHoldCall( )*** service will fail when ***activeCall*** specifies a screened call at the station of a Call Screener.

Conference

The ***cstaHoldCall( )*** service places a call on hold, not hold-for-conference.

Prior to MERLIN MAGIX Release 2.0, the ***cstaHoldCall( )*** service will place a conference call on hold if the last party added to conference was added on an SA button.

Beginning with MERLIN MAGIX Release 2.0, the ***cstaHoldCall( )*** service will place a conference call on hold only if the last party added to conference was added on a supported button type.

Coverage

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not use ***cstaHoldCall( )*** to hold a call on a Primary, Secondary or Group Coverage button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaHoldCall( )*** to hold a call on a Primary, Secondary or Group Coverage button.

Direct Facility Termination and Direct Pool Termination (DFT/DPT)

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not use ***cstaHoldCall( )*** to hold a call on a DFT or DPT button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***cstaHoldCall( )*** to hold a call on a DFT or DPT button.

End-Of-Dialing (Loop and Ground Start Trunks)

When an incoming call arrives on a Loop or Ground Start trunk, the users may be able to talk before the switch timers transition the trunk connection to End-of-Dialing. If an application requests ***cstaHoldCall( )*** before the transition to End-of-Dialing on this connection, the request will fail.

Intercom - Voice Announce

The ***cstaHoldCall( )*** service can hold a call that arrived as a Voice Announce Call.

Service Observing

An application may use the ***cstaHoldCall( )*** service to hold a call at a station that is being observed. When the call goes on hold, a monitor for the Service Obersever will receive a ***CSTAConnectionClearedEvent*** with a cause of `EC_SILENT_MONITOR`.

The ***cstaHoldCall( )*** service will fail when ***activeCall*** specifies an observed call at the station for a Service Observer.

Shared System Access Buttons

If an application requests the ***cstaHoldCall( )*** service for a call on a Shared System Access button, the request is denied.

Single Line Set

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not use ***cstaHoldCall( )*** to hold a call appearing on a Single Line Set.

Beginning with MERLIN MAGIX Release 2.0, an application may use the ***cstaHoldCall( )*** service to hold a call appearing on a Single Line Set.

Transfer

The ***cstaHoldCall( )*** service places a call on regular hold, not hold-for-transfer.

cstaMakeCall( )

The ***cstaMakeCall*** service places a call from the ***callingDevice*** to the ***calledDevice***. The confirmation event gives the connection identifier of the newly placed call at the ***callingDevice***.

The destination may be a local extension or external number. The ***calledDevice*** may contain a facilities indicator (such as “9” or a pool access code).

The ***callingDevice***, the originating extension, must be an internal MLX, ETR, 4400-series or ATL extension with an available SA button. The originating extension must be in Normal, Responding Mode.

The originating user must be either:

- off-hook on an SA button listening to dial tone
- off-hook on an SA button in the middle of dialing. (The ***calledDevice*** digits are inserted into the dialing at this point.)
- off-hook idle (no red LED appears on the extension)
- on-hook with an SA button available

If the switch can take the ***callingDevice***’s speakerphone off-hook and place the call, it will do so. If the calling extension is not already off-hook and drawing dial tone, and if the calling extension is in a suitable state for initiating a call, the switch will make the call. If the switch cannot take the extension’s speakerphone off-hook, and the extension is not in a suitable off-hook state for placing the call, then the switch will deny the request.

The MERLIN LEGEND and MERLIN MAGIX switches do not support authorization or account code entry for the ***cstaMakeCall( )*** service. Thus, the ***cstaMakeCall( )*** service will fail when ***callingDevice*** is administered for forced account code entry.

The ***calledDevice*** parameter may contain digits 0-9, *, and #. The MERLIN LEGEND and MERLIN MAGIX switches will ignore any alphabetic or pause characters. The MERLIN LEGEND and MERLIN MAGIX switches will dial all requested digits; if the requested digits do not form a valid number, then the user will hear reorder or an appropriate tone.

Service Request Parameters

Table 4-22. cstaMakeCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
callingDevice	originating extension
calledDevice	number to dial
privateData	`NULL`, not used for this service request

Scenario Diagram

Figure 4-7 illustrates a ***cstaMakeCall( )*** scenario.

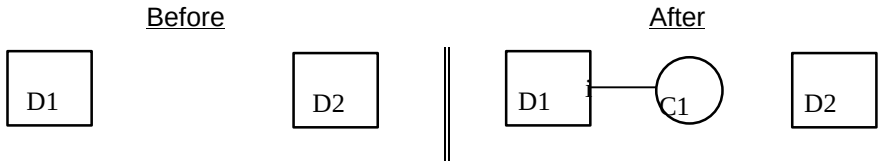


Figure 4-7. cstaMakeCall( ) Scenario

Return Values

Table 4-23. cstaMakeCall( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTAMakeCallConfEvent*

The MERLIN LEGEND or MERLIN MAGIX switch sends the ***CSTAMakeCallConfEvent*** after the call is originated and switch call processing progresses to the point that the switch allocates the connectionID for the new call. Call origination does not mean that the newly originated call has alerted at the destination. Resulting event reports (***CSTADeliveredEvent***, ***CSTANetworkReachedEvent***) will indicate alerting at the called extension or trunk seizure.

The deviceID in the ***newCall*** parameter in the confirmation event is the deviceID from the ***callingDevice*** request parameter.

Table 4-24. CSTAMakeCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_MAKE_CALL_CONF`
invokeID	identifies service request within stream
newCall	connectionID (containing both deviceID and callID) for the originated call at the ***callingDevice***
privateData	`NULL`, no private data present

CSTA Universal Failure Event Error Values

If the MERLIN LEGEND or MERLIN MAGIX switch cannot originate the call, then the MERLIN LEGEND or MERLIN MAGIX switch returns one of the errors below. For all ***error*** values except `GENERIC_UNSPECIFIED`, the MERLIN LEGEND and MERLIN MAGIX switches leave all connections at ***callingDevice*** in the states that they were in before the switch processed the ***cstaMakeCall( )*** request. `GENERIC_UNSPECIFIED` will, in most instances, also leave the connections in their initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaMakeCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when the call could not be originated for some reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CALLING_DEVICE` – The ***callingDevice*** is invalid. This may be because:

- The deviceID in ***callingDevice*** is not a valid extension in the dial plan.
- ***callingDevice*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- ***callingDevice*** is unknown or has a null value.
- ***callingDevice*** is configured as a QCC.
- ***callingDevice*** does not have an SA button available to originate the call.

INVALID_CALLED_DEVICE – The deviceID in ***calledDevice*** is invalid. This may be because:

- The deviceID in ***calledDevice*** specifies the same device as the deviceID in ***callingDevice***. (An extension cannot place a call to itself.)

INVALID_OBJECT_STATE – One of the following conditions occurred:

- ***callingDevice*** is active on another call (cannot originate a call while active on another call).
- ***callingDevice*** is not in a suitable initial state. A suitable initial state is:
  - — off-hook on an SA button and hearing dial tone or in the midst of dialing.
  - — off-hook on an SA button and dial tone has timed out, which results in the user becoming aware of silence. This is sometimes termed “high & dry”.
  - — on hook (the switch can force the speaker off hook.)
- ***callingDevice*** is Responding, but is not in Normal Mode.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED – Processing the ***cstaMakeCall( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

RESOURCE_BUSY – A needed resource is busy. Possible causes include:

- The switch is processing another CTI request for the extension in ***callingDevice***. Services such as ***cstaMakeCall( )*** (perhaps from another application) and ***cstaConsultationCall( )*** may be in progress when a ***cstaMakeCall( )*** request arrives.

REQUEST_TIMEOUT_REJECTION - The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server, MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaMakeCall ( ACSHandle_t    acsHandle,        /* INPUT */
               InvokeID_t     invokeID,         /* INPUT */
               DeviceID_t     *callingDevice,   /* INPUT */
               DeviceID_t     *calledDevice,    /* INPUT */
               PrivateData_t  *privateData);    /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAMakeCallConfEvent_t makeCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAMakeCallConfEvent_t {
   ConnectionID_t newCall;
} CSTAMakeCallConfEvent_t;
```

Important Feature Interactions

Auto Dial

The ***cstaMakeCall( ) service*** will not preempt the processing of a pending Auto Dial.

Automatic Line Selection (ALS)

If the originating extension is off-hook or if the red LED is lit at an SA button, then ***cstaMakeCall( )*** originates the call from that button.

Bridged Appearances

The ***cstaMakeCall( )*** service will not originate a call on a bridged appearance (SSA, DFT, etc.). The ***cstaMakeCall( )*** service will originate only on System Access Voice, System Access Ring, or System Access Originate Only buttons.

Group Page

The ***calledDevice*** may be a Group Page extension.

The ***cstaMakeCall( )*** service will not originate a call from a member of a group page that is active on a page call.

Redial

The ***calledDevice*** from a ***cstaMakeCall( )*** is retained as the Redial number.

Restrictions

The ***cstaMakeCall( )*** service will honor any restrictions administered for the originating and destination extensions. If the originating extension is toll restricted and the ***calledDevice*** is a toll number, the MERLIN LEGEND or MERLIN MAGIX switch will not originate the call.

Save Number Dial

The ***calledDevice*** from a ***cstaMakeCall( )*** may be retained as the Save Number Dialed.

Service Observing

An application may use ***cstaMakeCall( )*** to make a call at a station that is being observed.

An observer may activate Service Observing by pressing the Service Observing button on the station and then invoking the ***cstaMakeCall( )*** service with ***callingDevice*** set to the extension number of the service observer and ***calledDevice*** set to the extension number of the station being observed.

cstaRetrieveCall( )

The ***cstaRetrieveCall( )*** service retrieves a held, held-for-transfer, or held-for-conference connection ***heldCall*** at an extension. ***cstaRetrieveCall( )*** will not retrieve an associative held connection. Specifically, the ***heldCall***'s callID is retrieved at the ***heldCall***'s deviceID. The ***heldCall*** connection must be in a held state at the extension. The ***cstaRetrieveCall( )*** service will not drop another connection to retrieve a call. Thus, the retrieving extension cannot be active on another call for ***cstaRetrieveCall( )*** to be successful.

Service Request Parameters

Table 4-25. cstaRetrieveCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
heldCall	held connection containing both deviceID and callID
privateData	NULL, not used for this service request

Scenario Diagram

Figure 4-8 illustrates various ***cstaRetrieveCall( )*** scenarios where ***heldCall*** is the connection D1C1.

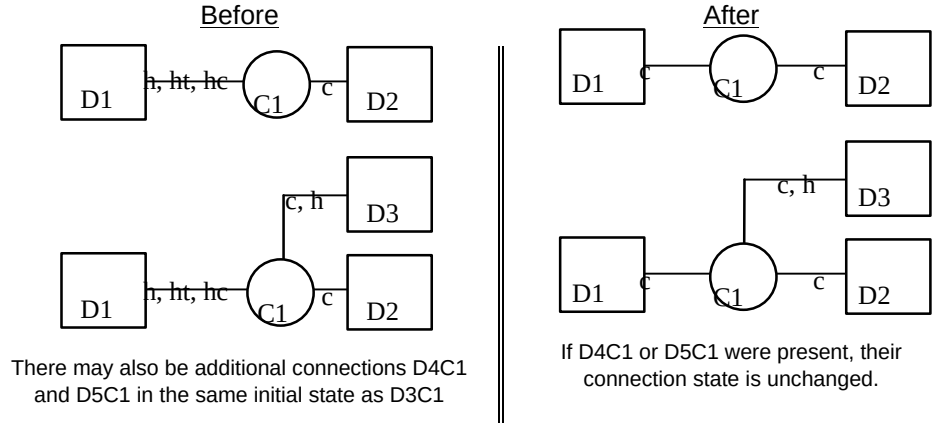


Figure 4-8. cstaRetrieveCall( ) Scenarios

Return Values

Table 4-26. cstaRetrieveCall( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTARetrieveCallConfEvent*

A ***CSTARetrieveCallConfEvent*** indicates that the switch has accepted the request, validated the parameters, and signaled the extension to retrieve the call. Application(s) monitoring the extension will receive a ***CSTARetrievedEvent*** when the connection has been retrieved.

Table 4-27. CSTARetrieveCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_RETRIEVE_CALL_CONF
invokeID	identifies service request within stream
privateData	NULL, no private data present

CSTA Universal Failure Event Error Values

If the ***heldCall*** connection cannot be retrieved, the MERLIN LEGEND or MERLIN MAGIX switch returns one of the errors below. For all ***error*** values except GENERIC_UNSPECIFIED, the MERLIN LEGEND and MERLIN MAGIX switches leave the ***heldCall*** connection in the state that it was in before the switch processed the ***cstaRetrieveCall( )*** request. GENERIC_UNSPECIFIED will, in most instances, also leave the connections in their initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaRetrieveCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when:

- callID in ***heldCall*** is not present on a supported button type at the deviceID in ***heldCall***.
  - — For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), only SA buttons are supported; SSA, Coverage, Line and Pool buttons are not supported.
  - — Beginning with MERLIN MAGIX Release 2.0, SA, Coverage, Line and Pool buttons are supported; only SSA buttons are not supported.
- The ***heldCall*** specifies an observed call at the station of a service observer.
- The ***heldCall*** connection could not be retrieved for some reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***heldCall*** is not valid. Some possible reasons are:

- No callID in ***heldCall.***
- The callID in ***heldCall*** does not exist in the MERLIN LEGEND or MERLIN MAGIX switch.
- callID in ***heldCall*** is not present at the deviceID in ***heldCall.***
- Invalid deviceID in ***heldCall***. One of the following may have occurred:
  - — deviceID is unknown or has a null value.
  - — deviceID is configured as a QCC.
- The deviceID in ***heldCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The application supplied a dynamic device identifier (the MERLIN LEGEND and MERLIN MAGIX switches do not use dynamic device identifiers).

`INVALID_OBJECT_STATE` – The ***heldCall*** connection is a valid connection identifier (the call is present at the extension) and one of the following conditions occurred:

- ***heldCall*** is not in the held state.
- The deviceID in ***heldCall*** is active on another call.
- The deviceID in ***heldCall*** is Responding, but is not in Normal Mode.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***cstaRetrieveCall( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

RESOURCE_BUSY – A needed resource is busy. Possible causes include:

- The switch is processing another TSAPI request for the extension in ***heldCall***. Services such as ***cstaMakeCall( )*** and ***cstaConsultationCall( )*** may be in progress when a ***cstaRetrieveCall( )*** request arrives.

REQUEST_TIMEOUT_REJECTION - The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server, MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaRetrieveCall (   ACSHandle_t acsHandle,     /* INPUT */
                     InvokeID_t        invokeID,      /* INPUT */
                     ConnectionID_t    *heldCall,     /* INPUT */
                     PrivateData_t     *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTARetrieveCallConfEvent_t   retrieveCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTARetrieveCallConfEvent_t {
   Nulltype null;
} CSTARetrieveCallConfEvent_t;
```

Important Feature Interactions

Call Screening

An application may use the ***cstaRetrieveCall( )*** service to retrieve a held call at a station that is participating in a screened call.

Callback Queuing (CBQ)

If a user has invoked the Callback Queuing feature for a call and then either hung up or post-selected away from that call, then the call is in associative hold. An application may not use ***cstaRetrieveCall( )*** to retrieve a call on associative hold, including a CBQ call.

Conference

Prior to MERLIN MAGIX Release 2.0, the ***cstaRetrieveCall( )*** service will connect to a held conference call so long as the conference call appears on at least one SA button at the extension.

Beginning with MERLIN MAGIX Release 2.0, the ***cstaRetrieveCall( )*** service will connect to a held-for-conference call as long as the conference call appears on at least one supported button type at the extension.

Coverage

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not use ***cstaRetrieveCall( )*** to retrieve a held call on a Primary, Secondary or Group Coverage button.

Beginning with MERLIN MAGIX Release 2.0, an application may use the ***cstaRetrieveCall( )*** service to retrieve a held call on a call on a Primary, Secondary or Group Coverage button.

Direct Facility Termination and Direct Pool Termination (DFT/DPT)

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not use ***cstaRetrieveCall( )*** to retrieve a held call on a DFT or DPT button.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***csta-RetrieveCall( )*** to retrieve a held call on a call on a DFT or DPT button.

Service Observing

An application may use the ***cstaRetrieveCall( )*** service to retrieve a call at a station that is being observed.

Shared System Access Buttons

If an application requests the ***cstaRetrieveCall( )*** service for a held call on a Shared System Access button, the request is denied.

Single Line Set

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not use ***cstaRetrieveCall( )*** to retrieve a held call at a Single Line Set.

Beginning with MERLIN MAGIX Release 2.0, an application may use ***csta-RetrieveCall( )*** to retrieve a held call on a supported button type at a Single Line Set.

Transfer

An application may use ***cstaRetrieveCall( )*** to connect to a call that is on-hold-for-transfer. The request will be successful whether or not the call is ringing at the transfer destination.

cstaTransferCall( )

The ***cstaTransferCall( )*** service transfers a held connection ***heldCall*** to an active connection ***activeCall*** at a common extension. The deviceID in the ***heldCall*** and ***activeCall*** must specify the common extension.

The ***heldCall*** must be on hold-for-transfer. The ***cstaTransferCall( )*** service will fail otherwise.

Prior to MERLIN MAGIX Release 2.0, the ***cstaTransferCall( )*** service will fail if the ***activeCall*** is a call to an external party.

An application may request the ***cstaTransferCall( )*** service after a successful invocation of the ***cstaConsultationCall( )*** service and thereby transfer the held call (held by ***cstaConsultationCall( )***) with the consultation call (originated by ***cstaConsultationCall( )***).

The MERLIN LEGEND and MERLIN MAGIX switches will deny an application request to transfer a held call after successful execution of ***cstaHoldCall( )*** and ***cstaMakeCall( )*** since the ***cstaHoldCall( )*** will not put the call on hold-for-transfer.

NOTE:
On a MERLIN LEGEND or MERLIN MAGIX switch, a transferred call may, or may not, remain at the transferring party. Whether or not the call remains at the transferring party depends on such factors as whether the transfer destination answers. The application should not infer from a successful transfer request that the call no longer appears at the transferring extension. In some situations, the application will receive a ***CSTATransferredEvent*** and then the call can still appear at or return to the transferring extension. The application will receive a ***CSTADeliveredEvent*** if the call returns and alerts.

The MERLIN LEGEND and MERLIN MAGIX switches will permit the interleaving of manual and CTI operations to effect a transfer as follows:

- Prerequisite: The user has an active connection and the application has a connectionID for that connection. This may occur when:
  - the user manually answers an incoming call (application has connectionID from Delivered and Established events),
  - the application uses ***cstaAnswerCall( )*** to answer an incoming call,
  - the application uses ***cstaMakeCall( )*** to make a call, or
  - the user manually places a call to another extension.
- the user manually presses TRANSFER button. The previously active connection is now on hold-for-transfer.

- The transferring user becomes connected on a second call either through using ***cstaMakeCall( )*** to make a call, or by answering an incoming call (manually or using ***cstaAnswerCall( )***). The application now has the connectionIDs for the active call and the held call.
- the application makes a ***cstaTransferCall( )*** request giving the connectionIDs for the held and active calls.

If the transfer cannot be done, then the switch leaves the ***heldCall*** and ***activeCall*** connections in the states that they were in before the switch began processing the ***cstaTransferCall( )*** call request.

Service Request Parameters

Table 4-28. cstaTransferCall( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
heldCall	held connection. Must contain deviceID and callID
activeCall	active connection. Must contain deviceID and callID
privateData	NULL, not used for this service request

Scenario Diagram

Figure 4-9 illustrates various ***cstaTransferCall( )*** scenarios where ***heldCall*** is the connection D1C1 and ***activeCall*** is the connection D1C2.

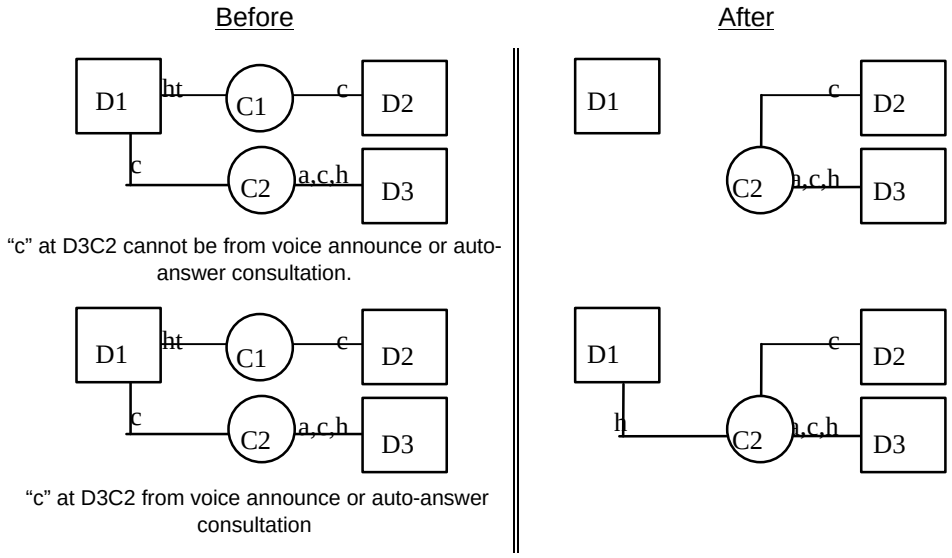


Figure 4-9. cstaTransferCall( ) Scenarios

Return Values

Table 4-29. cstaTransferCall( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted.

Confirmation Event - *CSTATransferCallConfEvent*

For a MERLIN LEGEND or MERLIN MAGIX switch, the call ID in the ***newCall*** will be the callID from the ***activeCall***. The application designer should not, however, use this fact in designing an application. As the switch supports more types of extensions and calls in the future, this may not continue to be the case.

The ***CSTATransferCallConfEvent*** indicates that the switch has accepted the request, validated the parameters, performed necessary call processing, and signaled the extension to transfer the call. Application(s) monitoring the extension will receive a ***CSTATransferredEvent*** when the transfer occurs.

Table 4-30. CSTATransferCallConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_TRANSFER_CALL_CONF`
invokeID	identifies service request within stream
newCall	connectionID containing DeviceID and CallID of the resulting call at the transfer destination
connList	The MERLIN LEGEND and MERLIN MAGIX switches do not provide this optional TSAPI parameter. In the `ConnectionList_t` structure, `count` is set to zero and the `connection` pointer is set to `NULL`.
privateData	`NULL`, no private data present

CSTA Universal Failure Event Error Values

If the ***activeCall*** and ***heldCall*** cannot be transferred, the MERLIN LEGEND or MERLIN MAGIX switch returns one of the errors below. For all ***error*** values except `GENERIC_UNSPECIFIED`, the MERLIN LEGEND and MERLIN MAGIX switches leave the ***activeCall*** and ***heldCall*** connections in the state that they were in before the switch processed the ***cstaTransferCall( )*** request. `GENERIC_UNSPECIFIED` will, in most instances, also leave the connections in their initial state, but there are a few circumstances where this cannot be guaranteed.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaTransferCall( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when:

- ***activeCall*** is Senderized.
- ***activeCall*** is in a DGC queue.
- ***activeCall*** is a conference call and the device in the ***activeCall*** connection is not the conference controller.
- callID in ***activeCall*** or ***heldCall*** is not present on a supported button type at the extension.

- For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), only SA buttons are supported; SSA, Coverage, Line and Pool buttons are not supported.
- Beginning with MERLIN MAGIX Release 2.0, SA, Coverage, Line and Pool buttons are supported; only SSA buttons are not supported.

- Either the ***activeCall*** or ***heldCall*** specifies an observed call at the station of a service observer.
- The ***activeCall*** and ***heldCall*** connections could not be transferred for some reason other than the more specific reasons given below.

`GENERIC_OPERATION` – The deviceIDs in ***activeCall*** and ***heldCall*** are not identical. (They must be identical since the transfer must occur at an extension common to the two calls.)

`INVALID_CSTA_CONNECTION_IDENTIFIER` – The connection identifier ***activeCall*** or ***heldCall*** is not valid. Some possible reasons are:

- No callID in activeCall or heldCall.
- The callID in activeCall or heldCall does not exist in the MERLIN LEGEND or MERLIN MAGIX switch.
- the callID in activeCall is not present at the deviceID in activeCall.
- the callID in heldCall is not present at the deviceID in heldCall.
- Invalid deviceID in activeCall or heldCall. One of the following may have occurred:
  - deviceID is unknown or has a null value.
  - deviceID is configured as a QCC.
- The deviceID in ***activeCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The deviceID in ***heldCall*** is not a supported extension set type in Responding Mode. (The extension may be out of service.)
- The application supplied a dynamic device identifier (the MERLIN LEGEND and MERLIN MAGIX switches do not use dynamic device identifiers).

`INVALID_OBJECT_STATE` – The ***activeCall*** and ***heldCall*** connections are valid connection identifiers (the call is present at the extension) and one of the following conditions occurred:

- The callID in ***activeCall*** is present at the deviceID in ***activeCall***, but the connection is not the active connection at the extension. It is on hold or in some other state.
- The deviceID in ***activeCall*** is Responding, but is not in Normal Mode.

- The callID in ***heldCall*** is present at the deviceID in ***heldCall***, but the connection is not held-for-transfer. It is in some other state. This occurs if the ***heldCall*** is on regular hold or hold-for-transfer.
- The deviceID in ***heldCall*** is Responding, but is not in Normal Mode.
- The ***heldCall*** is a conference call.
- The callID in ***activeCall*** is a conference call and the device in ***activeCall*** is the conference controller.
- An attempt was made to transfer to an external party.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***cstaTransferCall( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_BUSY` – A needed resource is busy. Possible causes include:

- The switch is processing another TSAPI request for the transferring extension. Services such as ***cstaMakeCall( )*** and ***cstaConsultationCall( )*** may be in progress when a ***cstaTransferCall( )*** request arrives.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server, MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

Request Syntax

```
cstaTransferCall (ACSHandle_t    acsHandle,     /* INPUT */
                  InvokeID_t     invokeID,      /* INPUT */
                  ConnectionID_t *heldCall,     /* INPUT */
                  ConnectionID_t *activeCall,   /* INPUT */
                  PrivateData_t  *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTATransferCallConfEvent_t   transferCall;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTATransferCallConfEvent_t {
   ConnectionID_t    newCall;
   ConnectionList_t  connList;
} CSTATransferCallConfEvent_t;
```

Important Feature Interactions

Auto Answer All - AAA (ATL Only – MERLIN LEGEND and MERLIN MAGIX 1.0)

The ***cstaTransferCall( )*** service will successfully complete when the transfer destination uses Auto Answer All to answer the consultation call.

Auto Answer Intercom - AAI (ATL Only – MERLIN LEGEND and MERLIN MAGIX 1.0)

The ***cstaTransferCall( )*** service will successfully complete when the transfer destination uses Auto Answer Intercom to answer the consultation call.

Bridged Appearances (SSA)

The ***cstaTransferCall( )*** service will successfully transfer a call when the consultation call is answered at a Shared SA (SSA) button. Note that in this case, the original transfer destination did not connect to the call. The resulting ***CSTATransferredEvent*** will contain the extension of the extension that bridged onto the call.

Call Screening

An application may use the ***cstaTransferCall( )*** service to complete a transfer operation at a station that is partipating in a screened call.

An application may not use the ***cstaTransferCall( )*** service to complete a transfer operation at a station that is screening a call.

Call Waiting

Call Waiting may queue the consultation call at the destination and ***cstaTransferCall( )*** will successfully complete the transfer.

Callback Queuing (CBQ)

The ***cstaTransferCall( )*** service will succeed if Callback (Automatic or Selective) queues the consultation call.

Conference

A user cannot press the conference button, place another call, and then use the ***cstaTransferCall( )*** service from an application to complete a transfer.

Coverage

The ***cstaTransferCall( )*** service will successfully transfer a call where the far end has the call appearing on a COVER button.

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Release 1.0) environment, if the ***activeCall*** appears on a Primary, Secondary or Group COVER button at an extension, the ***cstaTransferCall( )*** service cannot complete a transfer of that call on behalf of that extension.

Beginning with MERLIN MAGIX Release 2.0, the ***cstaTransferCall( )*** service will successfully transfer a call when the ***activeCall*** or ***heldCall*** is on a Primary, Secondary or Group COVER button.

Direct Facility Termination/Personal Lines

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, if the ***activeCall*** appears only on a DFT button at an extension, then ***cstaTransferCall( )*** cannot transfer that call on behalf of that extension.

Beginning with MERLIN MAGIX 2.0, the ***cstaTransferCall( )*** will successfully transfer a call when the ***activeCall*** or ***heldCall*** is on a DFT button.

Group Calling (DGC)

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, if an application attempts to make a consultation call to a Calling Group as a means to transfer another call to that Calling Group, the ***cstaConsultationCall( )*** request will be denied.

Beginning with MERLIN MAGIX 1.5, if an application attempts to make a consultation call to a Calling Group as a means to transfer another call to that Calling Group, the ***cstaConsultationCall( )*** request will be granted.

Forward/Follow Me

The ***cstaTransferCall( )*** service will successfully transfer a call when the consultation call forwards from the transfer destination to another extension.

Hands Free Answer on Intercom (HFAI)

The ***cstaTransferCall( )*** service will successfully transfer a call where the far end used HFAI to connect to the call.

Networking

In a MERLIN LEGEND (Release 6.0 and later) and MERLIN MAGIX (Release 1.0) environment, if an application attempts to make a consultation call as a means to transfer a call to a station on another MERLIN LEGEND or MERLIN MAGIX switch in the private network, the ***cstaConsultationCall( )*** request will be denied. The user may make the consultation call and transfer the call by using the station set.

Beginning with MERLIN MAGIX Release 1.5, if an application attempts to make a consultation call as a means to transfer a call with a station on another MERLIN LEGEND or MERLIN MAGIX switch in the private network, the ***csta-ConsultationCall( )*** request will be granted.

Park

The ***cstaTransferCall( )*** service cannot be used to park a call by transferring the call from an extension to itself (in the manner of manual Park operation).

Senderized Operation

The ***cstaTransferCall( )*** service will fail if ***activeCall*** is Senderized (same as manual transfer completion operation).

Service Observing

An application may use the ***cstaTransferCall( )*** service to complete a transfer operation at a station that is being observed.

The ***cstaTransferCall( )*** service will fail when either the ***activeCall*** or the ***heldCall*** specifies an observed call at the station of a service observer; a service observer may not use the Transfer feature with an observed call.

System Access (SA)/Shared System Access (SSA) Buttons

The ***cstaTransferCall( )*** service will successfully transfer a call where the far end has the call appearing on an SSA button.

Transfer

The ***cstaTransferCall( )*** service operates the same way as transfer completion (the second press of the TRANSFER button). Refer to the *MERLIN LEGEND Advanced Communication System Feature Reference* or *MERLIN MAGIX Integrated System Feature Reference* for complete information.

The MERLIN LEGEND or MERLIN MAGIX Transfer Return feature may cause the ***activeCall*** to return to the transferring extension and re-alert. If this occurs, an application monitoring the transferring extension will receive a ***CSTADeliveredEvent***.

The ***activeCall*** in ***cstaTransferCall( )*** may not be a conference call since the MERLIN LEGEND and MERLIN MAGIX switches will not permit the conference call controller to transfer a conference call.

When an unsupervised transfer is done on a MERLIN LEGEND or MERLIN MAGIX switch, an appearance of the call remains at the transferring extension until the transfer destination answers the transferred call. At that time, the appearance disappears from the transferring extension. The ***CSTATransferredEvent*** does not list the transferring party in its connection list. An application monitoring the transferring party will not receive a ***CSTAConnectionClearedEvent*** when the transfer destination answers and the appearance disappears.

Voice Announce

When a consultation call is automatically answered at the speaker, the arriving call is answered on speaker and there is no ***CSTADeliveredEvent*** for the arriving call. There is a ***CSTAEstablishedEvent***. When a transfer operation joins the consultation call with the held call, the Voice Announce call clears (monitoring applications will see ***CSTAConnectionClearedEvent***s) and the newly joined call alerts at the consultation destination. Monitors will receive a ***CSTADeliveredEvent*** for the newly alerting call. The connection identifier for the call may contain a call identifier that is different than that of the call of the Voice Announce.

Supplementary Services

5

Contents

Contents

Supplementary Services

5

Applications use Supplementary Services to access switch features. MERLIN MAGIX CTI Supplementary Services allow an application to:

- Set an agent's login state (MERLIN MAGIX Release 1.5 and later)
- Query the state of an agent (MERLIN MAGIX Release 2.0 and later)
- Change the status of Do Not Disturb feature at an extension (MERLIN MAGIX Release 2.1 and later)
- Query the Do Not Disturb status of a extension (MERLIN MAGIX Release 2.1 and later)
- Set and clear the Message Waiting Lamp at an extension (MERLIN MAGIX Release 2.1 and later)
- Query the status of an extension's Message Waiting Lamp (MERLIN MAGIX Release 2.1 and later)

Table 5-1 shows the TSAPI Supplementary Services and confirmation events that the MERLIN MAGIX switch provides.

Table 5-1. MERLIN MAGIX CTI Support for TSAPI Supplementary Services

	TSAPI Supplementary Services and Events - MERLIN MAGIX Release 1.5
Ö	cstaSetMsgWaitingInd( ) & CSTASetMwiConfEvent cstaSetDoNotDisturb( ) & CSTASetDndConfEvent cstaSetForwarding( ) & CSTASetFwdConfEvent cstaSetAgentState( ) & CSTASetAgentStateConfEvent cstaQueryMsgWaitingInd( ) & CSTAQueryMwiConfEvent cstaQueryDoNotDisturb( ) & CSTAQueryDndConfEvent cstaQueryFwd( ) & CSTAQueryFwdConfEvent cstaQueryAgentState( ) & CSTAQueryAgentStateConfEvent cstaQueryLastNumber( ) & CSTAQueryLastNumberConfEvent cstaQueryDeviceInfo( ) & CSTAQueryDeviceInfoConfEvent

TSAPI Supplementary Services and Events - MERLIN MAGIX Release 2.0

cstaSetMsgWaitingInd( ) & CSTASetMwiConfEvent
cstaSetDoNotDisturb( ) & CSTASetDndConfEvent
cstaSetForwarding( ) & CSTASetFwdConfEvent
- cstaSetAgentState( ) & CSTASetAgentStateConfEvent

cstaQueryMsgWaitingInd( ) & CSTAQueryMwiConfEvent
cstaQueryDoNotDisturb( ) & CSTAQueryDndConfEvent
cstaQueryFwd( ) & CSTAQueryFwdConfEvent
- cstaQueryAgentState( ) & CSTAQueryAgentStateConfEvent

cstaQueryLastNumber( ) & CSTAQueryLastNumberConfEvent
cstaQueryDeviceInfo( ) & CSTAQueryDeviceInfoConfEvent

TSAPI Supplementary Services and Events - MERLIN MAGIX Release 2.1 and later

- cstaSetMsgWaitingInd( ) & CSTASetMwiConfEvent
- cstaSetDoNotDisturb( ) & CSTASetDndConfEvent

cstaSetForwarding( ) & CSTASetFwdConfEvent
- cstaSetAgentState( ) & CSTASetAgentStateConfEvent
- cstaQueryMsgWaitingInd( ) & CSTAQueryMwiConfEvent
- cstaQueryDoNotDisturb( ) & CSTAQueryDndConfEvent

cstaQueryFwd( ) & CSTAQueryFwdConfEvent
- cstaQueryAgentState( ) & CSTAQueryAgentStateConfEvent

cstaQueryLastNumber( ) & CSTAQueryLastNumberConfEvent
cstaQueryDeviceInfo( ) & CSTAQueryDeviceInfoConfEvent

CAUTION:
When designing an application, be aware that the MERLIN MAGIX switch may not support all of the optional TSAPI supplementary service parameters. The pages describing each supplementary service show all of the TSAPI parameters and indicate those that the MERLIN MAGIX switch supports.

Sending Supplementary Service Requests and Receiving Confirmations

Each Supplementary Service request has an associated confirmation event. This book presents information about each service's confirmation event under the heading for the service.

An application must receive the confirmation event on the stream where it sends the Supplementary Service request. "Receiving Events" in Chapter 3 describes how applications receive confirmation events.

In general, it is recommended that an application monitor the extension it is controlling so that it receives Agent and Feature Status Events reflecting activity at the extension. Chapter 6 describes the Monitoring Services.

Supplementary Service Request Failures

If the service request fails for some reason, the application will receive a ***CSTAUniversalFailureConfEvent*** in place of the service confirmation. Each service description includes a list of the ***error*** values that the ***CSTAUniversalFailureConfEvent*** may carry for that service as well as the meanings of those values in the context of that service. Since the ***CSTAUniversalFailureConfEvent*** applies to other services, as well as Supplementary Services, its description is found in the section ***CSTAUniversalFailureConfEvent*** in Chapter 3.

Supplementary Service Page Format

The pages describing each TSAPI supplementary service contain the following sections, as appropriate:

Service Name and Description

The service name appears first. A description of that service immediately follows the name.

Service Request Parameters

A table lists the service request parameters and summarizes their use.

Return Values

A table lists the return values for the service request.

In all function returns, success values follow the TSAPI rules. If the requesting application generated the ***invokeID*** value, then a successful function call returns zero. If the TSAPI library generates the ***invokeID*** value, then a successful function call returns the value of the ***invokeID***. This is not explicitly re-stated for each service. “Sending TSAPI Requests and Receiving Confirmations” in Chapter 3 describes ***invokeID*** usage in more detail.

Confirmation Event

This section names the TSAPI confirmation event for the service and contains a table describing the confirmation event parameters.

CSTA Universal Failure Confirmation Event Error Values

This section lists error values that the ***CSTAUniversalFailureConfEvent*** may return to an application when a service request fails. Items in all capitals are #defines from the TSAPI header files (acs.h, acsdefs.h, csta.h, and cstadefs.h).

Request Syntax

This section contains C coding information for the service request.

Confirmation Event Syntax

This section contains C coding information for the service's confirmation event.

Important Feature Interactions

This section describes important interactions between the supplementary service and MERLIN MAGIX switch features.

cstaQueryAgentState( )

The ***cstaQueryAgentState( )*** service provides the agent state of an extension. This service is available beginning with MERLIN MAGIX Release 2.0.

Beginning with MERLIN MAGIX Release 2.1, an extension may be a member of multiple Calling Groups. The ***cstaQueryAgentState( )*** service can be called with a Calling Group provided in private data to obtain the agent status of an extension for that particular group. Tables 5-20 and 5-21 provide the ***agentState*** values in order of precedence, along with the corresponding MERLIN MAGIX state.

Table 5-2. MERLIN MAGIX CTI Agent States - Calling Group Not Specified

agentState	MERLIN MAGIX Agent State
AG_WORK_NOT_READY	Extension is in the After Call Work State
AG_NULL	Extension is logged out (Unavailable)
AG_NOT_READY	Extension is logged in (Available) but is *not* ready to accept Calling Group calls
AG_READY	Extension is logged in (Available) and is ready to accept Calling Group calls

Table 5-3. MERLIN MAGIX CTI Agent States - Calling Group Specified in Private Data

agentState	MERLIN MAGIX Agent State
AG_NULL	Extension is logged out of the specified group. No indication is provided as to whether Auxiliary Work Time or After Call Work is active.
AG_WORK_NOT_READY	Extension is in Auxiliary Work Time or After Call Work state, and agent extension is logged into the specified group.
AG_NOT_READY	Extension is logged into the specified group and neither Auxiliary Work Time nor After Call Work is active, but the agent station is unavailable to take a DGC call for some other reason.
AG_READY	Extension is logged in to the specified group and is ready to take a DGC call. Neither Auxiliary Work Time nor After Call Work is active.

This service is valid for all non-QCC station types. The station does not have to be a member of a DGC Group.

Service Request Parameters

Table 5-4. cstaQueryAgentState( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
device	the extension number of a telephone in this MERLIN MAGIX system.

Private Service Request Parameters

Table 5-5. cstaQueryAgentState( ) Private Service Request Parameters in MERLIN MAGIX Release 2.1

dgcID	identifies the DGC Group for the query

Return Values

Table 5-6. cstaQueryAgentState( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTAQueryAgentStateConfEvent*

The ***CSTAQueryAgentStateConfEvent*** indicates that the switch is able to provide the Agent State of ***device***.

Table 5-7 CSTAQueryAgentStateConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_QUERY_AGENT_STATE_CONF
invokeID	identifies service request within stream
agentState	identifies the state of the agent
privateData	NULL, no private data present

CSTA Universal Failure Confirmation Event Errors

If the Agent State of ***device*** cannot be provided, MERLIN MAGIX CTI returns one of the errors below. The MERLIN MAGIX switch leaves the ***device*** in the state that it was in before the switch processed the ***cstaQueryAgentState( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaQueryAgentState( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

GENERIC_UNSPECIFIED - An application will receive GENERIC_UNSPECIFIED when the ***device*** could not be queried for some reason other than the more specific reasons given below.

RESOURCE_OUT_OF_SERVICE - The CTI link is disconnected or not in service.

INVALID_FEATURE – The CTI link is connected to a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Release 1.0 or 1.5) switch.

INVALID_CSTA_DEVICE_IDENTIFIER - The device identifier ***device*** is not valid. Some possible reasons are:

- The ***device*** is configured as a QCC.
- The ***device*** is not an extension on the MERLIN MAGIX system.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED - Processing the ***cstaQueryAgentState( )*** service requests exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

REQUEST_TIMEOUT_REJECTION - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlQueryAgentState (MLPrivateData_t     *privateData);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;


cstaQueryAgentState (ACSHandle_t acsHandle,     /* INPUT */
               InvokeID_t        invokeID,      /* INPUT */
               DeviceID_t        *device,       /* INPUT */
               PrivateData_t     *privateData); /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAQueryAgentStateConfEvent_t   queryAgentState;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAQueryAgentStateConfEvent_t {
   AgentState_t      agentState;
} CSTAQueryAgentStateConfEvent_t;

typedef enum AgentState_t = {
   AG_NOT_READY = 0,
   AG_NULL = 1,
   AG_READY = 2,
   AG_WORK_NOT_READY = 3,
   AG_WORK_READY = 4
} AgentState_t;
```

Important Feature Interactions

Call States

If the ***device*** is on a call, the state of the call will not be affected by the ***cstaQueryAgentState( )*** service request.

DGC Membership

The ***cstaQueryAgentState( )*** service request will be granted even if the ***device*** is not a member of a Calling Group.

Extension Status Mode

The ***cstaQueryAgentState( )*** service is available in both Hotel/Motel and Group Calling Supervisor mode.

agentMode `AM_LOG_OUT` corresponds to Extension Status 0.

agentMode `AM_WORK_NOT_READY` corresponds to Extension Status 1.

agentMode `AM_LOG_IN` corresponds to Extension Status 2.

cstaQueryDoNotDisturb( )

The ***cstaQueryDoNotDisturb( )*** service allows an application to get the current status of the Do Not Disturb feature at an extension. This service is available beginning with MERLIN MAGIX Release 2.1.

This service is valid for all non-QCC station types.

Service Request Parameters

Table 5-8. cstaQueryDoNotDisturb( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
device	the extension number of a telephone in this MERLIN MAGIX system
privateData	`NULL`, not used for this service request

Return Values

Table 5-9. cstaQueryDoNotDisturb( ) Return Values

zero or positive value	Success
`ACSERR_BADHDL`	***acsHandle*** is not a valid stream identifier
`ACSERR_STREAM_FAILED`	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTAQueryDndConfEvent*

The ***CSTAQueryDndConfEvent*** indicates that the switch has accepted the request, validated the parameters, and obtained the Do Not Disturb feature status.

Table 5-10. CSTAQueryDndConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_QUERY_DND_CONF
invokeID	identifies service request within stream
doNotDisturb	Indicates whether the Do Not Disturb feature is active (TRUE) or inactive (FALSE)
privateData	NULL, no private data present

CSTA Universal Failure Confirmation Event Error Values

If the ***device*** cannot be queried, MERLIN MAGIX CTI returns one of the errors below. The MERLIN MAGIX switch leaves the *device* in the state it was in before the switch processed the ***cstaQueryDoNotDisturb( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaQueryDoNotDisturb( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

GENERIC_UNSPECIFIED – An application will receive GENERIC_UNSPECIFIED when the state of device could not be obtained for some reason other than the more specific reasons given below.

RESOURCE_OUT_OF_SERVICE – The CTI link is disconnected or not in service.

INVALID_CSTA_DEVICE_IDENTIFIER – The device identifier ***device*** is not valid. Some possible reasons are:

- The ***device*** is configured as a QCC.
- The ***device*** is not a local extension on the MERLIN MAGIX system.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED – Processing the ***cstaQueryDoNotDisturb( )*** request would exceed the maximum number of outstanding requests permitted at either the driver or the switch.

REQUEST_TIMEOUT_REJECTION – The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION – A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaQueryDoNotDisturb (ACSHandle_t  acsHandle,        /* INPUT */
               InvokeID_t           invokeID,         /* INPUT */
               DeviceID_t           *device,          /* INPUT */
               PrivateData_t        *privateData);    /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAQueryDndConfEvent_t queryDnd;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAQueryDndConfEvent_t {
   boolean     doNotDisturb;
} CSTAQueryDndConfEvent_t;
```

Important Feature Interactions

Call States

If the ***device*** is on a call, the state of the call will not be affected by the ***cstaQueryDoNotDisturb( )*** service request.

cstaQueryMsgWaitingInd( )

The ***cstaQueryMsgWaitingInd( )*** service allows an application to get the current status of an extension's Message Waiting Lamp. This service is available beginning with MERLIN MAGIX Release 2.1.

This service is valid for all non-QCC station types.

Service Request Parameters

Table 5-11. cstaQueryMsgWaitingInd( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
device	the extension number of a telephone in this MERLIN MAGIX system
privateData	`NULL`, not used for this service request

Return Values

Table 5-12. cstaQueryMsgWaitingInd( ) Return ValuesQueryMsgWaitingInd

zero or positive value	Success
`ACSERR_BADHDL`	***acsHandle*** is not a valid stream identifier
`ACSERR_STREAM_FAILED`	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTAQueryMwiConfEvent*

The ***CSTAQueryMwiConfEvent*** provides the status of the Message Waiting Lamp at ***device***.

Table 5-13. CSTAQueryMwiConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_QUERY_MWI_CONF
invokeID	identifies service request within stream
messages	indicates whether the Message Waiting Lamp is on (TRUE) or off (FALSE)
privateData	NULL, no private data present

CSTA Universal Failure Confirmation Event Error Values

If the ***device*** cannot be queried, MERLIN MAGIX CTI returns one of the errors below. The MERLIN MAGIX switch leaves the ***device*** in the state it was in before the switch processed the ***cstaQueryMsgWaitingInd( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaQueryMsgWaitingInd( )*** request, the ***CSTAUniversalFailureConf-Event*** will contain one of the following values in the ***error*** parameter:

GENERIC_UNSPECIFIED – An application will receive GENERIC_UNSPECIFIED when the state of device could not be provided for some reason other than the more specific reasons given below.

RESOURCE_OUT_OF_SERVICE – The CTI link is disconnected or not in service.

INVALID_CSTA_DEVICE_IDENTIFIER – The device identifier device is not valid. Some possible reasons are:

- The ***device*** is configured as a QCC.
- The ***device*** is not a local extension on the MERLIN MAGIX system.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED – Processing the ***cstaQueryMsgWaitingInd( )*** service request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

REQUEST_TIMEOUT_REJECTION – The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION – A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaQueryMsgWaitingInd (ACSHandle_t acsHandle,        /* INPUT */
               InvokeID_t           invokeID,         /* INPUT */
               DeviceID_t           *device,          /* INPUT */
               PrivateData_t        *privateData);    /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAQueryMwiConfEvent_t queryMwi;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAQueryMwiConfEvent_t {
     Boolean         messages;
} CSTAQueryMwiConfEvent_t;
```

Important Feature Interactions

Leave Word Calling

The ***cstaQueryMsgWaitingInd( )*** service will return TRUE if the station being queried has one or more Leave Word Calling messages in its mailbox.

Fax Message Waiting

The ***cstaQueryMsgWaitingInd( )*** service will return TRUE if the station being queried is administered as a Fax Message Waiting receiver and a fax has been received.

Voice Mail

The ***cstaQueryMsgWaitingInd( )*** service will return TRUE if the Voice Mail system has activated the Message Waiting Lamp at the station to indicate the presence of new voice mail.

cstaSetAgentState( )

The ***cstaSetAgentState( )*** service sets the state (***agentMode***) of an extension. The effect of the ***cstaSetAgentState( )*** is equivalent to the agent pressing various programmed buttons (Login, Logout, Agent Work Time, After Call Work, and Available) at his/her extension. This service is available beginning with MERLIN MAGIX Release 1.5.

MERLIN MAGIX Release 2.0 CTI supports the ***agentMode*** values listed in Table 5-14.

Table 5-14. MERLIN MAGIX CTI Supported Agent Modes in Release 2.0

agentMode	**MERLIN MAGIX Agent Mode**
AM_LOG_IN	Log agent in (Available)
AM_LOG_OUT	Log agent out (Unavailable)
AM_WORK_NOT_READY	Place agent in the After Call Work State

MERLIN MAGIX Release 2.1 CTI supports the ***agentMode*** values listed in Table 5-16.

Table 5-15. MERLIN MAGIX CTI Supported Agent Modes in Release 2.1

agentMode	**MERLIN MAGIX Agent Mode**
AM_LOG_IN	Log agent into one or more groups
AM_LOG_OUT	Log agent out of one or more groups
AM_WORK_NOT_READY	Place agent in Auxiliary Work Time or After Call Work State to be unavailable
AM_WORK_READY	Take agent out of Auxiliary Work Time or After Call Work state to be available

This service is valid for all non-QCC station types. The station does not have to be a member of a DGC Group.

Service Request Parameters

Table 5-16. cstaSetAgentState( ) Parameters for MERLIN MAGIX Releases 1.5 and 2.0

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
device	the extension number of a telephone in this MERLIN MAGIX system.
agentMode	the new agent state: `AM_LOGGED_IN`, `AM_LOGGED_OUT`, or `AM_WORK_NOT_READY`
agentID	This parameter has no effect.
agentGroup	This parameter has no effect.
agentPassword	This parameter has no effect.
privateData	`NULL`, not used for this service request

Table 5-17. cstaSetAgentState( ) Parameters for MERLIN MAGIX Release 2.1 and later

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
device	the extension number of a telephone in this MERLIN MAGIX system.
agentMode	the new agent state: `AM_LOGGED_IN`, `AM_LOGGED_OUT`, `AM_WORK_NOT_READY` or `AM_WORK_READY`
agentID	This parameter has no effect.
agentGroup	The Calling Group ID, or null.
agentPassword	This parameter has no effect.
privateData	`NULL`, not used for this service request

Beginning with MERLIN MAGIX Release 2.1, an extension may be a member of multiple Calling Groups. An application has the ability to perform Login or Logout operations for either an individual group or for multiple groups when making a ***cstaSetAgentState( )*** service request:

- If ***agentGroup*** is specified and the ***agentMode*** value is `AM_LOGGED_IN` or `AM_LOGGED_OUT`, then the Login/Logout operation affects the login status only for the specified ***agentGroup***.

- If ***agentGroup*** is not specified and the ***agentMode*** is AM_LOGGED_IN, then the ***device*** is Logged In to all groups of which it is a member. If, however the ***device*** is not a member of any Calling Group, then the device is Logged In to all Calling Groups in the system.
- If ***agentGroup*** is not specified and the ***agentMode*** is AM_LOGGED_OUT, then the ***device*** is Logged Out of all Calling Groups.

The ***cstaSetAgentState( )*** service may also be used by an application to move an extension to either the AM_WORK_READY mode or the AM_WORK_NOT_READY mode. Beginning with MERLIN MAGIX Release 2.1, these modes are independent of an extension's Login/Logout status. Therefore, the ***agentGroup*** parameter has no effect when requesting an extension be moved to either AM_WORK_READY or AM_WORK_NOT_READY.

Return Values

Table 5-18. cstaSetAgentState( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTASetAgentStateConfEvent*

The ***CSTASetAgentStateConfEvent*** indicates that the switch has accepted the request, validated the parameters, and signaled the extension to change states. Application(s) monitoring the extension will receive a ***CSTALoggedOnEvent, CSTALoggedOffEvent***, ***CSTAWorkNotReadyEvent*** or ***CSTAWorkReady-Event*** (the ***CSTAWorkReadyEvent*** is available beginning with MERLIN MAGIX Release 2.1)**,** when the extension changes states.

Table 5-19. CSTASetAgentStateConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_SET_AGENT_STATE_CONF`
invokeID	identifies service request within stream
privateData	`NULL`, no private data present

CSTA Universal Failure Confirmation Event Error Values

If the agent status for ***device*** cannot be changed, MERLIN MAGIX CTI returns one of the errors below. For all ***error*** values except `GENERIC_UNSPECIFIED`, the MERLIN MAGIX switch leaves the ***device*** in the state that it was in before the switch processed the ***cstaSetAgentState( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaSetAgentState( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_UNSPECIFIED` when the state of device could not be changed for some reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_FEATURE` - An application will receive `INVALID_FEATURE` when:

- For MERLIN MAGIX Release 2.1 or later, the requested ***agentMode*** is not `AM_LOG_IN`, `AM_LOG_OUT`, `AM_WORK_READY` or `AM_WORK_NOT_READY`.
- For MERLIN MAGIX Release 2.0, the requested ***agentMode*** is not `AM_LOG_IN`, `AM_LOG_OUT` or `AM_WORK_NOT_READY`.
- The CTI link is connected to a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX Release 1.0 switch.

INVALID_CSTA_DEVICE_IDENTIFIER - The device identifier ***device*** is not valid. Some possible reasons are:

- The ***device*** is configured as a QCC.
- The ***device*** is not a local extension on the MERLIN MAGIX system.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED - Processing the ***cstaSetAgentState( )*** service request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

REQUEST_TIMEOUT_REJECTION - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaSetAgentState (ACSHandle_t   acsHandle,          /* INPUT */
                   InvokeID_t    invokeID,           /* INPUT */
                   DeviceID_t    *device,            /* INPUT */
                   AgentMode_t   agentMode,          /* INPUT */
                   AgentID_t     *agentID,           /* INPUT */
                   AgentGroup_t  *agentGroup,        /* INPUT */
                   AgentPassword_t *agentPassword,   /* INPUT */
                   PrivateData_t *privateData);      /* INPUT */

typedef enum AgentMode_t = {
   AM_LOG_IN = 0,
   AM_LOG_OUT = 1,
   AM_NOT_READY = 2,
   AM_READY = 3,
   AM_WORK_NOT_READY = 4,
   AM_WORK_READY = 5
} AgentMode_t;

typedef char          AgentID_t[32];

typedef DeviceID_t    AgentGroup_t;

typedef char          AgentPassword_t[32];
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTASetAgentStateConfEvent_t  setAgentState;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTASetAgentStateConfEvent_t {
   Nulltype    null;
} CSTASetAgentStateConfEvent_t;
```

Important Feature Interactions

Call States

If the ***device*** is on a call, the state of the call will not be affected by the ***cstaSetAgentState( )*** service request.

Calling Group Membership

The ***cstaSetAgentState( )*** service requested will be granted even if the ***device*** is not a member of a Calling Group.

Extension Status Mode

The ***cstaSetAgentState( )*** service is available in both Hotel/Motel and Group Calling Supervisor mode.

Agent mode AM_LOG_IN corresponds to Extension Status 2. Agent mode AM_LOG_OUT corresponds to Extension Status 0 and agent mode AM_WORK_NOT_READY corresponds to Extension Status 1.

cstaSetDoNotDisturb( )

The **csta*SetDoNotDisturb( )*** service allows the application to activate or deactivate the Do Not Disturb feature at an extension. This service is available beginning with MERLIN MAGIX Release 2.1.

This service is valid for all extensions that have a programmed Do Not Disturb button.

This service is not valid for QCC, Single Line Set, and Multi-Function Module extensions.

Service Request Parameters

Table 5-20. cstaSetDoNotDisturb( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
device	the extension number of a telephone in this MERLIN MAGIX system
doNotDisturb	Activate (TRUE) or deactivate (FALSE) the Do Not Disturb feature
privateData	NULL, not used for this service request

Return Values

Table 5-21. cstaSetDoNotDisturb( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTASetDndConfEvent*

The ***CSTASetDndConfEvent*** indicates that the switch has accepted the request, validated the parameters, and signaled the extension to change Do Not Disturb feature status. An application monitoring the extension will receive a ***CSTADo-NotDisturbEvent*** if the extension changes its Do Not Disturb status.

Table 5-22. CSTASetDndConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_SET_DND_CONF`
invokeID	identifies service request within stream
privateData	`NULL`, no private data present

CSTA Universal Failure Confirmation Event Error Values

If the Do Not Disturb feature cannot be activated/deactivated at the ***device***, the MERLIN MAGIX CTI returns one of the errors below. The MERLIN MAGIX switch leaves the ***device*** in the state it was in before the switch processed the ***cstaSet-DoNotDisturb( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaSetDoNotDisturb( )*** request, the ***CSTAUniversalFailureConfEvent*** contains one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when the state of device could not be changed for some reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` – The device identifier ***device*** is not valid. Some possible reasons are:

- The ***device*** is configured as a QCC.
- The ***device*** is a Single Line Set.
- The ***device*** is an MFM.
- The ***device*** is not a local extension on the MERLIN MAGIX system.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***cstaSetDoNotDisturb( )*** request would exceed the maximum number of outstanding requests permitted at either the driver or the switch.

INVALID_OBJECT_STATE – The extension is not in normal responding mode.

REQUEST_TIMEOUT_REJECTION – The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION – A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

GENERIC_SUBSCRIBED_RESOURCE_AVAILABILITY – The extension does not have a Do Not Disturb button.

Request Syntax

```
cstaSetDoNotDisturb (ACSHandle_t acsHandle,           /* INPUT */
                     InvokeID_t          invokeID,    /* INPUT */
                     DeviceID_t          *device,     /* INPUT */
                     Boolean             doNotDisturb,/* INPUT */
                     PrivateData_t       *privateData);/* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTASetDndConfEvent_t   setDnd;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTASetDndConfEvent_t {
   Nulltype    null;
} CSTASetDndConfEvent_t;
```

Important Feature Interactions

Do Not Disturb

A Do Not Disturb button must be programmed on device in order for the ***cstaSet-DoNotDisturb( )*** service request to succeed.

Normal, Responding Mode

The ***cstaSetDoNotDisturb( )*** service request will fail when ***device*** is not in normal, responding mode.

Station Types

Single Line Sets and MFM's are not eligible for this service (Do Not Disturb is not supported on these types of sets).

cstaSetMsgWaitingInd( )

The ***cstaSetMsgWaitingInd( )*** service allows an application to set or clear an extension's Message Waiting Lamp. This service is available beginning with MERLIN MAGIX Release 2.1.

The MERLIN MAGIX switch maintains a Mailbox for each extension. The Mailbox may contain up to ten messages. A user is able to view these messages using his/her station display and soft keys.

When the ***cstaSetMsgWaitingInd( )*** service is used to turn on the Message Waiting Lamp, a CTI message is inserted into the station's mailbox, analogous to a voice mail message. Even if several requests are made to turn on the Message Waiting Lamp, only one CTI message will appear in the Mailbox. The existing CTI message will be overwritten, with the new message having a new time stamp and unread message flag.

If the application sends a request to turn off the Message Waiting Lamp while it is on due to a prior application request, the existing CTI message in the Mailbox will be deleted. If there are no other messages in the Mailbox the Message Waiting Lamp will be turned off. The ***cstaSetMsgWaitingInd( )*** service will be considered successful whether or not the Message Waiting Lamp is turned off as a result of the request.

This service is valid for all non-QCC station types.

Service Request Parameters

Table 5-23. cstaSetMsgWaitingInd( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
device	the extension number of a telephone in this MERLIN MAGIX system
messages	Set Message Waiting Lamp on (TRUE) or off (FALSE)
privateData	NULL, not used for this service request

Return Values

Table 5-24. cstaSetMsgWaitingInd( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTASetMwiConfEvent*

The ***CSTASetMwiConfEvent*** indicates that the switch has accepted the request, validated the parameters, and signaled the extension to change the state of the Message Waiting Lamp.

Table 5-25. CSTASetMwiConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_SET_MWI_CONF
invokeID	identifies service request within stream
privateData	NULL, no private data present

CSTA Universal Failure Confirmation Event Error Values

If the status of the Message Waiting Indicator cannot be set at ***device***, MERLIN MAGIX CTI returns one of the errors below. The MERLIN MAGIX switch leaves the ***device*** in the state it was in before the switch processed the ***cstaSetMsgWaitingInd( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaSetMsgWaitingInd( )*** request, the ***CSTAUniversalFailureConfEvent*** contains one of the following values in the ***error*** parameter:

GENERIC_UNSPECIFIED – An application will receive GENERIC_UNSPECIFIED when the state of device could not be changed for some reason other than the more specific reasons given below.

RESOURCE_OUT_OF_SERVICE – The CTI link is disconnected or not in service.

INVALID_CSTA_DEVICE_IDENTIFIER – The device identifier ***device*** is not valid. Some possible reasons are:

- The ***device*** is configured as a QCC.
- The ***device*** is not a local extension on the MERLIN MAGIX system.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***cstaSetMsgWaitingInd( )*** request would exceed the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` – The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` – A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request. For example, the extension's mailbox is full.

Request Syntax

```
cstaSetMsgWaitingInd (ACSHandle_t   acsHandle,        /* INPUT */
              InvokeID_t            invokeID,         /* INPUT */
              DeviceID_t            *device,          /* INPUT */
              Boolean               messages,         /* INPUT */
              PrivateData_t         *privateData);    /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTASetMwiConfEvent_t   setMwi;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTASetMwiConfEvent_t {
   Nulltype    null;
} CSTASetMwiConfEvent_t;
```

Important Feature Interactions

Messaging

A user may delete the CTI message as they would any other type of message in the extension's mailbox.

If a successful ***cstaSetMsgWaitingInd( )*** service request is made with the messages parameter set to FALSE**,** the extension's Message Waiting indicator may remain on due to other messages in the extension's mailbox.

Normal, Responding Mode

An application may use the ***cstaSetMsgWaitingInd( )*** service to set the status of an extension's Message Waiting Lamp even when the station is not in normal, responding mode.

Station Types

Non-display sets (including single line sets) are eligible to receive CTI messages. Without a display, a user will not be able to accurately determine why the Message Waiting Lamp is on.

Monitoring

6

Contents

Contents

Monitoring

6

Applications use Monitoring services to monitor devices and receive Call Events when call activity occurs at a monitored device. “Switch Environment” in Chapter 2 details the devices that an application may monitor. Applications use the Monitoring Services to establish a monitor. They use the event reception services (included in Chapter 3) to receive the resulting events (Chapter 8).

Table 6-1 shows the TSAPI Monitoring Services and Events that the MERLIN LEGEND and MERLIN MAGIX switches provide. Note that the MERLIN LEGEND and MERLIN MAGIX switches do not provide all of the TSAPI Monitoring Services and Events.

Table 6-1. MERLIN LEGEND/MERLIN MAGIX CTI Support for TSAPI Monitoring Services and Events

	TSAPI Monitoring Services and Events
Ö	cstaMonitorDevice cstaMonitorCall cstaMonitorCallsViaDevice
Ö	CSTAMonitorConfEvent
Ö	cstaMonitorStop and CSTAMonitorStopConfEvent cstaChangeMonitorFilter and CSTAChangeMonitorFilterConfEvent
Ö	CSTAMonitorEndedEvent

CAUTION:
When designing an application, be aware of not only the services and events that the MERLIN LEGEND and MERLIN MAGIX switches provide but also the parameters within those services and events. The MERLIN LEGEND and MERLIN MAGIX switches do not provide all of the optional TSAPI service and event parameters. The event manual pages list all of the TSAPI parameters and indicate those that the MERLIN LEGEND and MERLIN MAGIX switches provide.

NOTE:
Applications should always be event driven, and use Call Events to react to changes in connection states. An application, especially an application that is intended to be switch-independent, cannot anticipate how all of the various switch-specific features on the various switch vendors' products will affect connection states. Many switches have connection states (such as the MERLIN LEGEND and MERLIN MAGIX Associative Active state) that the CSTA connection state model does not adequately incorporate. Vendors reflect these using the TSAPI events in various ways.

Monitor Types

The MERLIN LEGEND and MERLIN MAGIX switches support device monitoring. The MERLIN LEGEND and MERLIN MAGIX switches do not support "TSAPI Call Monitoring" or "TSAPI Monitor Calls Via Device."

A device monitor provides an application with call events for calls that appear on supported button types (see "Button Types" in Chapter 2) at a monitored extension, Agent Status Events (in MERLIN MAGIX Release 1.5) or, beginning with MERLIN MAGIX Release 2.0, with call events for calls in a Calling Group queue, Feature and Agent Status Events. Generally, when a call no longer appears on a monitored button type at a monitored extension, the application receives a ***CSTAConnectionClearedEvent***.[1] The call may continue to appear at other extensions in the system (and may be reflected in Call Events for those extensions), but an application will not receive any further events on the monitor for the extension from which the call disappeared.

An application may request monitors on multiple devices.

Event Filtering

When an application requests a monitor, it may specify an event filter. The event filter specifies that the application is to receive only certain events (a subset of the set that the MERLIN LEGEND or MERLIN MAGIX switch provides.) The MERLIN LEGEND PBX Driver and MERLIN MAGIX PBX Driver do not permit an application to change the event filter for an active monitor.

1 The application will receive a ***CSTATransferredEvent*** (rather that a ***CSTAConnectionClearedEvent***) when a call leaves a device as a result of a transfer operation.

cstaMonitorDevice( )

The ***cstaMonitorDevice( )*** service provides Call Events reflecting telephone activity at a device. An application uses the ***monitorFilter*** parameter to request that the driver filter out events that it does not wish to receive. When an application makes a successful ***cstaMonitorDevice( )*** request, the MERLIN LEGEND switch will

- report call events for calls that originate from or arrive at the device after the ***CSTAMonitorConfEvent***.
- report call events for a call in progress at the device beginning with the next call event for that call at the monitored device. Note that until the monitored device takes some action on the call, the application will not receive any events for other parties on such a call. Once the monitored device takes some action on the call, the application will receive events for all parties on the call.

The MERLIN MAGIX switch will report the above and in addition,

- report Agent Status events for a device (in this case a station) after the ***CSTAMonitorConfEvent***.
- report Feature events for a device (in this case a station) after the ***CSTAMonitorConfEvent***.
- report queue events for calls that enter or leave a Calling Group Queue

Once an application begins receiving events about a call, the application will continue to receive events pertaining to the call so long as the call remains at the monitored device. When the call leaves the monitored device, the application will not receive further events pertaining to the call (on the monitor for that device.)

Chapter 2 details the extension and button types on which the MERLIN LEGEND and MERLIN MAGIX switches provide monitoring.

The PBX driver will request no more than one monitor on a device at any time across a MERLIN LEGEND or MERLIN MAGIX CTI link. If multiple streams monitor the same device, then the driver replicates the events for each of the monitoring applications. Thus, even though the switch limits the number of monitors on a device, the driver design allows a greater number of monitors.

Unlike call control requests, a telephone does not need to be in Normal, Responding Mode for an application to successfully establish a monitor on that telephone. A monitor will continue even if the telephone transitions to/from Normal, Responding mode. The MERLIN LEGEND or MERLIN MAGIX switch will generate events for a telephone regardless of whether it is in a Normal, Responding Mode.

Service Request Parameters

Table 6-2. cstaMonitorDevice( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
deviceID	monitor this station[2] or (beginning in MERLIN MAGIX Release 1.5) Calling Group queue
monitorFilter	optionally specifies a subset of events the application will receive
privateData	`NULL`, not used for this service request

Return Values

Table 6-3. cstaMonitorDevice( ) Return Values

zero or positive value	Success
`ACSERR_BADHDL`	***acsHandle*** is not a valid stream identifier.
`ACSERR_STREAM_FAILED`	***acsHandle*** is not valid. The stream may have been closed or aborted.

Confirmation Event - *CSTAMonitorConfEvent*

The driver assigns a monitor cross-reference ID when it successfully enables a monitor on a device. The ***monitorCrossRefID*** will be present in any following subsequent events sent to the requesting application. Each monitor within the stream has a unique ***monitorCrossRefID***.

[2] For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX Releases 1.0, monitoring a station means that the application will receive events for calls appearing on SA buttons at that station. Beginning with MERLIN MAGIX Release 1.5, the application will also receive agent status events for the station. Beginning with MERLIN MAGIX Release 2.0, the application will also receive events for calls appearing at Line, Pool and Coverage buttons at the station.

Table 6-4. CSTAMonitorConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_MONITOR_CONF`
invokeID	identifies service request within stream
monitorCrossRefID	associates subsequent events with this monitor
monitorFilter	structure indicating the event set that the monitor will provide
privateData	`NULL`, no private data present

CSTA Universal Failure Event Error Values

`GENERIC_UNSPECIFIED` – The monitor could not be started for a reason other than the more specific reasons given below.

`INVALID_CSTA_DEVICE_IDENTIFIER` – The deviceID does not identify a station of a type that may be monitored, or (beginning with MERLIN MAGIX Release 1.5) a Calling Group queue. This value is returned if an application tries to monitor a QCC or MFM. Chapter 2 discusses the device types that may be monitored.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the service request exceeds the maximum number of outstanding requests permitted at either the PBX driver or the switch.

`OVERALL_MONITOR_LIMIT_EXCEEDED` – This monitor would exceed the switch's or PBX driver's limit for monitors (across all devices).

`REQUEST_TIMEOUT_REJECTION` – The MERLIN LEGEND PBX driver or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` – A Telephony Server, MERLIN LEGEND PBX driver , or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

Request Syntax

```
cstaMonitorDevice (  ACSHandle_t       acsHandle,     /* INPUT */
                     InvokeID_t        invokeID,      /* INPUT */
                     DeviceID_t        *deviceID,     /* INPUT */
                     CSTAMonitorFilter_t *monitorFilter,/* INPUT */
                     PrivateData_t     *privateData); /* INPUT */
```

Confirmation Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union
   {
      CSTAMonitorConfEvent_t        monitorStart;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAMonitorConfEvent_t {
   CSTAMonitorCrossRefID_t monitorCrossRefID;
   CSTAMonitorFilter_t     monitorFilter;
} CSTAMonitorConfEvent_t;
```

```
typedef unsigned short  CSTACallFilter_t;
#define                     CF_CALL_CLEARED          0x8000
#define                     CF_CONFERENCED           0x4000
#define                     CF_CONNECTION_CLEARED    0x2000
#define                     CF_DELIVERED             0x1000
#define                     CF_DIVERTED              0x0800
#define                     CF_ESTABLISHED           0x0400
#define                     CF_FAILED                0x0200
#define                     CF_HELD                  0x0100
#define                     CF_NETWORK_REACHED       0x0080
#define                     CF_ORIGINATED            0x0040
#define                     CF_QUEUED                0x0020
#define                     CF_RETRIEVED             0x0010
#define                     CF_SERVICE_INITIATED     0x0008
#define                     CF_TRANSFERRED           0x0004

typedef unsigned char   CSTAFeatureFilter_t;
#define                     FF_CALL_INFORMATION      0x80
#define                     FF_DO_NOT_DISTURB        0x40
#define                     FF_FORWARDING            0x20
#define                     FF_MESSAGE_WAITING       0x10

typedef unsigned char   CSTAAgentFilter_t;
#define                     AF_LOGGED_ON             0x80
#define                     AF_LOGGED_OFF            0x40
#define                     AF_NOT_READY             0x20
#define                     AF_READY                 0x10
#define                     AF_WORK_NOT_READY        0x08
#define                     AF_WORK_READY            0x04

typedef unsigned char   CSTAMaintenanceFilter_t;
#define                     MF_BACK_IN_SERVICE       0x80
#define                     MF_OUT_OF_SERVICE        0x40

typedef struct CSTAMonitorFilter_t {
   CSTACallFilter_t        call;
   CSTAFeatureFilter_t     feature;
   CSTAAgentFilter_t       agent;
   CSTAMaintenanceFilter_t maintenance;
   long                    privateFilter;
} CSTAMonitorFilter_t;
```

If the application does not apply any event filtering, then the ***monitorFilter*** in the Confirmation Event will indicate that the MERLIN LEGEND or MERLIN MAGIX switch will provide the following default set of events:

Table 6-5. Events Provided With No Event Filtering

	TSAPI Call Events for Monitored Stations - MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX Release 1.0
	CSTACallClearedEvent
Ö	CSTAConferencedEvent
Ö	CSTAConnectionClearedEvent
Ö	CSTADeliveredEvent
	CSTADivertedEvent
Ö	CSTAEstablishedEvent
	CSTAFailedEvent
Ö	CSTAHeldEvent
Ö	CSTANetworkReachedEvent
	CSTAOriginatedEvent
	CSTAQueuedEvent
Ö	CSTARetrievedEvent
Ö	CSTAServiceInitiatedEvent
Ö	CSTATransferredEvent

	TSAPI Call Events for Monitored Stations - MERLIN MAGIX (Releases 1.5 and later)
	CSTACallClearedEvent
Ö	CSTAConferencedEvent
Ö	CSTAConnectionClearedEvent
Ö	CSTADeliveredEvent
Ö	CSTADivertedEvent
Ö	CSTAEstablishedEvent
	CSTAFailedEvent
Ö	CSTAHeldEvent
Ö	CSTANetworkReachedEvent
	CSTAOriginatedEvent
Ö	CSTAQueuedEvent
Ö	CSTARetrievedEvent
Ö	CSTAServiceInitiatedEvent
Ö	CSTATransferredEvent

TSAPI Call Events for Monitored Calling Group Queues - MERLIN MAGIX (Releases 1.5 and later)

CSTACallClearedEvent
CSTAConferencedEvent
- CSTAConnectionClearedEvent

CSTADeliveredEvent
- CSTADivertedEvent

CSTAEstablishedEvent
CSTAFailedEvent
CSTAHeldEvent
CSTANetworkReachedEvent
CSTAOriginatedEvent
- CSTAQueuedEvent

CSTARetrievedEvent
CSTAServiceInitiatedEvent
CSTATransferredEvent

TSAPI Feature Event Reports for Monitored Stations - MERLIN MAGIX Release 2.0

CSTACallInfoEventEvent
- CSTADoNotDisturbEvent

CSTAForwardingEvent
CSTAMessageWaitingEvent

TSAPI Feature Event Reports for Monitored Stations - MERLIN MAGIX Release 2.1 and later

- CSTACallInfoEventEvent
- CSTADoNotDisturbEvent

CSTAForwardingEvent
CSTAMessageWaitingEvent

TSAPI Agent Status Events for Monitored Stations - MERLIN MAGIX Release 1.5

- CSTALoggedOnEvent
- CSTALoggedOffEvent

CSTANotReadyEvent
CSTAReadyEvent
- CSTAWorkNotReadyEvent

CSTAWorkReadyEvent

TSAPI Agent Status Events for Monitored Stations - MERLIN MAGIX Release 2.0

- CSTALoggedOnEvent
- CSTALoggedOffEvent
- CSTANotReadyEvent
- CSTAReadyEvent
- CSTAWorkNotReadyEvent
- CSTAWorkReadyEvent

TSAPI Agent Status Events for Monitored Stations - MERLIN MAGIX Release 2.1 and later

- CSTALoggedOnEvent
- CSTALoggedOffEvent
- CSTANotReadyEvent
- CSTAReadyEvent
- CSTAWorkNotReadyEvent
- CSTAWorkReadyEvent

Important Feature Interactions

An application may only monitor supported station types (see “Switch Environment - Extension Types” in Chapter 2), or, beginning with MERLIN MAGIX Release 1.5, Calling Group queues.

Group Calling

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX Release 1.0 environment, an application may not monitor a Calling Group queue.

Beginning with MERLIN MAGIX Release 1.5, an application may monitor a Calling Group queue.

MFM

An application may not monitor an MFM.

Single Line Sets

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application may not monitor a Single Line Set.

Beginning with MERLIN MAGIX Release 2.0, an application may monitor a Single Line Set.

QCC

An application may not monitor a QCC.

Voice Response Port

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX Releases 1.0 and 1.5 environments, an application may not monitor a Voice Response Unit.

Beginning with MERLIN MAGIX Release 2.0, an application may monitor a Voice Response Unit.

cstaMonitorStop( )

The ***cstaMonitorStop*** service terminates the event reporting for a device on this stream.

Service Request Parameters

Table 6-6. cstaMonitorStop( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
monitorCrossRefID	stop this monitor
privateData	`NULL`, not used for this service request

Return Values

Table 6-7. cstaMonitorStop( ) Return Values

zero or positive value	Success
`ACSERR_BADHDL`	***acsHandle*** is not a valid stream identifier.
`ACSERR_STREAM_FAILED`	***acsHandle*** is not valid. The stream may have been closed or aborted.

Confirmation Event - *CSTAMonitorStopConfEvent*

Table 6-8. CSTAMonitorStopConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_MONITOR_STOP_CONF`
invokeID	identifies service request within stream
privateData	`NULL`, no private data present

CSTA Universal Failure Event Error Values

`GENERIC_UNSPECIFIED` – The monitor could not be stopped for some reason other than the more specific reasons below.

`INVALID_CROSS_REF_ID` – The ***monitorCrossRefID*** is invalid.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the service request exceeds the maximum number of outstanding requests permitted at either the PBX driver or the switch.

`REQUEST_TIMEOUT_REJECTION` – The MERLIN LEGEND or MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` – A Telephony Server, MERLIN LEGEND PBX driver, or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

Request Syntax

```
cstaMonitorStop ( ACSHandle_t          acsHandle,        /* INPUT */
                  InvokeID_t           invokeID,         /* INPUT */
                  CSTAMonitorCrossRefID_t monitorCrossRefID,/* INPUT */
                  PrivateData_t        *privateData);    /* INPUT */
```

Confirmation Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct  {
   ACSEventHeader_t  eventHeader;
   union  {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct  {
   InvokeID_t  invokeID;
   union {
      CSTAMonitorStop_t    monitorStop;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAMonitorStop_t {
   CSTAMonitorCrossRefID_t monitorCrossRefID;
} CSTAMonitorStop_t;
```

CSTAMonitorEndedEvent

This event indicates that the MERLIN LEGEND or MERLIN MAGIX switch terminated the event reporting for a device.

An application must be prepared to receive a Monitor Ended event for any monitored device at any time. The CSTAMonitorEndedEvent may be the result of a transient problem (e.g. the CTI link was reset or temporarily disconnected). A robust application will implement a strategy for re-establishing a device monitor when this event is received.

Event Parameters

Table 6-9. CSTAMonitorEndedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_MONITOR_ENDED`
monitorCrossRefID	Monitor that ended
cause	reason monitor ended
privateData	`NULL`, none present in Monitor Ended event

Event Causes

Table 6-10. CSTAMonitorEndedEvent Causes

`EC_NETWORK_NOT_OBTAINABLE`	CTI link failed.

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAMonitorEndedEvent_t monitorEnded;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAMonitorEndedEvent_t {
   CSTAEventCause_t  cause;
} CSTAMonitorEndedEvent_t;
```

Important Feature Interactions

Busy-Out

When a port or board for a monitored device is busied-out, application will receive a ***CSTAMonitorEndedEvent*** for that device.

Cold Start

When the MERLIN LEGEND or MERLIN MAGIX switch goes through a cold start, the application will receive a ***CSTAMonitorEndedEvent*** for all monitored devices.

Server Time Change

When the Telephony Server's time is changed, the application may receive a ***CSTAMonitorEndedEvent*** for all monitored devices.

Snapshot Services

7

Contents

Contents

Snapshot Services

7

An application uses Snapshot Services to query the current state of a call or device. MERLIN MAGIX CTI Snapshot Services allow an application to determine information about calls associated with an extension. The information includes a list of Calls associated with the given extension and the Connection State of each Call.

Table 7-1 shows the TSAPI Snapshot Services and confirmation events that the MERLIN MAGIX switch provides beginning with MERLIN MAGIX Release 2.1.

Table 7-1. MERLIN MAGIX CTI Support for TSAPI Snapshot Services

	TSAPI Snapshot Services - MERLIN MAGIX Release 2.1 and later
	cstaSnapshotCallReq( ) & CSTASnapshotCallConfEvent
Ö	cstaSnapshotDeviceReq( ) & CSTASnapshotDeviceConfEvent

Sending Snapshot Service Requests and Receiving Confirmations

Each Snapshot Service request has an associated confirmation event. This book presents information about each service's confirmation event under the heading for the service.

An application must receive the confirmation event on the stream where it sends the Snapshot Service request. "Receiving Events" in Chapter 3 describes how applications receive confirmation events.

Confirmations have different meanings for various services. Refer to the manual page for each service when coding applications so as to use the service confirmations properly. In general, it is recommended that an application monitor the extension it is controlling so that it receives events reflecting the call activity at the extension. Chapter 6 describes the Monitoring Services.

Snapshot Service Request Failures

If the service request fails for some reason, the application will receive a ***CSTAUniversalFailureConfEvent*** in place of the service confirmation. Each service description includes a list of the ***error*** values that the ***CSTAUniversalFailureConfEvent*** may carry for that service as well as the meanings of those values in the context of that service. Since the ***CSTAUniversalFailureConfEvent*** applies to other services, as well as Snapshot Services, its description is found in the section pertaining to ***CSTAUniversalFailureConfEvent*** in Chapter 3.

Snapshot Service Page Format

The pages describing each TSAPI snapshot service contain the following sections, as appropriate:

Service Name and Description

The service name appears first. A description of that service immediately follows the name.

Service Request Parameters

A table lists the service request parameters and summarizes their use.

Return Values

A table lists the return values for the service request.

In all function returns, success values follow the TSAPI rules. If the requesting application generated the ***invokeID*** value, then a successful function call returns zero. If the TSAPI library generates the ***invokeID*** value, then a successful function call returns the value of the ***invokeID***. This is not explicitly re-stated for each service. “Sending TSAPI Requests and Receiving Confirmations” in Chapter 3 describes ***invokeID*** usage in more detail.

Confirmation Event

This section names the TSAPI confirmation event for the service and contains a table describing the confirmation event parameters.

CSTA Universal Failure Confirmation Event Error Values

This section lists error values that the ***CSTAUniversalFailureConfEvent*** may return to an application when a service request fails. Items in all capitals are #defines from the TSAPI header files (acs.h, acsdefs.h, csta.h, and cstadefs.h).

Request Syntax

This section contains C coding information for the service request.

Confirmation Event Syntax

This section contains C coding information for the service's confirmation event.

Important Feature Interactions

This section describes important interactions between the snapshot service and MERLIN MAGIX switch features.

cstaSnapshotDeviceReq( )

The ***cstaSnapshotDeviceReq( )*** service provides information about calls associated with an extension. The information includes the Call Identifier and Call State for each call at the station (up to ten calls are reported). The Call State is comprised of a list of Connection State values for up to five endpoints on the call, starting with the Connection State of the specified station. The information does not include the identity of the other devices on the call.

This service is available beginning with MERLIN MAGIX Release 2.1.

This service is valid for all non-QCC station types.

Service Request Parameters

Table 7-2. cstaSnapshotDeviceReq( ) Parameters

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
snapshotObj	the extension number of a telephone in this MERLIN MAGIX system
privateData	`NULL`, not used for this service request

Return Values

Table 7-3. cstaSnapshotDeviceReq( ) Return Values

zero or positive value	Success
`ACSERR_BADHDL`	***acsHandle*** is not a valid stream identifier
`ACSERR_STREAM_FAILED`	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - *CSTASnapshotDeviceConfEvent*

The ***CSTASnapshotDeviceConfEvent*** indicates that the switch has accepted the request, validated the parameters.

Table 7-4. CSTASnapshotDeviceConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_SNAPSHOT_DEVICE_CONF_EVENT`
invokeID	identifies service request within stream
snapshotData	contains call information for ***snapshotObj*** including: ***count***: the number of calls being reported (0-10) ***info[ ]***: an array of calls, where each element contains: ***callIdentifier***: Connection Identifier ***callState***: a list of local connection states for end points starting with the local connection state of ***snapshotObj***
privateData	`NULL`, no private data present

CSTA Universal Failure Confirmation Event Error Values

If ***snapshotObj*** cannot be queried, MERLIN MAGIX CTI returns one of the errors below. The MERLIN MAGIX switch leaves the extension in the state it was in before the switch processed the ***cstaSnapshotDeviceReq( )*** request.

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***cstaSnapshotDeviceReq( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_-UNSPECIFIED` when a snapshot of ***snapshotObj*** could not be provided for some reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` - The device identifier ***snapshotObj*** is not valid. Some possible reasons are:

- The ***snapshotObj*** is configured as a QCC.
- The ***snapshotObj*** is not a local extension on the MERLIN MAGIX system.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED - Processing the ***cstaSnapshotDeviceReq( )*** service request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

REQUEST_TIMEOUT_REJECTION - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
cstaSnapshotDeviceReq (ACSHandle_t  acsHandle,        /* INPUT */
               InvokeID_t           invokeID,         /* INPUT */
               DeviceID_t           *snapshotObj,     /* INPUT */
               PrivateData_t        *privateData);    /* INPUT */
```

Confirmation Event Syntax

```
typedef struct  {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      struct
      {
         InvokeID_t  invokeID;
         union
         {
            CSTASnapshotDeviceConfEvent_t snapshotDevice;
         } u;
      } cstaConfirmation;
   } event;
} CSTAEvent_t;


typedef struct CSTASnapshotDeviceConfEvent_t {
   CSTASnapshotDeviceData_t   snapshotData;
} CSTASnapshotDeviceConfEvent_t;

typedef struct CSTASnapshotDeviceData_t {
   int   count;
   struct CSTASnapshotDeviceResponseInfo_t   *info;
} CSTASnapshotDeviceData_t;

typedef struct CSTASnapshotDeviceResponseInfo_t {
    ConnectionID_t      callIdentifier;
    CSTACallState_t     localCallState;
} CSTASnapshotDeviceResponseInfo_t;

typedef struct CSTACallState_t {
    int                         count;
    LocalConnectionState_t    *state;
} CSTACallState_t;

typedef enum LocalConnectionState_t {
   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;
```

Important Feature Interactions

Busy Calls

If a call being reported is an internal call to a busy station without coverage or forwarding (i.e., the calling party hears busy), the Local Connection State of ***snapshotObj*** will be reported as CS_INITIATE, and no other endpoints on the call will be reported.

Call Screening

When the ***cstaSnapshotDeviceReq( )*** service is requested for a station that is screening a call, all eligible calls will be reported, including the screened call.

When the ***cstaSnapshotDeviceReq( )*** service is requested for a station on a call that is being screened by another station, the status of the call at the Call Screening station will not be reported.

Connection States

The ***snapshotObj*** may be active on a call and the service will not affect the state of the station or any endpoint on the call.

For incoming calls alerting at ***snapshotObj***, the Local Connection State of the external endpoint is reported as CS_CONNECT. It remains CS_CONNECT for the duration of the call or until the endpoint is removed from the call.

For external calls originated from ***snapshotObj***, the Local Connection State of the external endpoint is reported as either CS_NULL, CS_ALERTING, or CS_CONNECT, depending on the call progress and type of facility used.

The Local Connection State of ***snapshotObj*** may be reported as CS_INITIATE, CS_ALERTING, CS_CONNECT, or CS_HOLD.

The Local Connection State of internal extension endpoints may be reported as CS_NULL, CS_ALERTING, CS_CONNECT, or CS_HOLD.

When reporting the Local Connection States of internal extension endpoints (other than ***snapshotObj***), no filtering of data is performed based on the button type at the endpoint. For example, if station A calls station B and station C answers the call on an SSA button for station B, a Snapshot of A will include the CS_CONNECT Local Connection State for station C, even though C answered the call on a SSA button.

The Local Connection State of a DGC endpoint is reported as CS_QUEUED.

While a DPT call is alerting at ***snapshotObj***, a Snapshot of the station will include the Local Connection State of the call at ***snapshotObj*** (CS_ALERTING), but will not include the Local Connection State of the call at any other station.

Coverage

While a coverage call is alerting:

- A Snapshot of the coverage sender will include the Local Connection State of the call at ***snapshotObj***, but will not include the Local Connection State of the call at any of the coverage receivers.
- A Snapshot of a coverage receiver will include the Local Connection State of the call at ***snapshotObj***. The Snapshot will also include the Local Connection State of the call at the coverage sender if and only if the call is alerting at the coverage sender station. The Snapshot will not include the Local Connection State of the call at any other coverage receiver.

Direct Facility Termination (DFT) Buttons

Calls appearing on DFT buttons at ***snapshotObj*** will not be reported by the ***cstaSnapshotDeviceReq( )*** service.

Forwarding

While a forwarded call is alerting:

- A Snapshot of the forwarding station will include the Local Connection State of the call at that station (`CS_ALERTING`), but will not include the Connection State of the call at the forwarding destination station.
- A Snapshot of the forwarding destination station will include the Local Connection State of the call at that station (`CS_ALERTING`). The Snapshot will include the Local Connection State of the call at the forwarding station if and only if the call is alerting at the forwarding station.

Group Calling (DGC)

The Local Connection State of a DGC endpoint is reported as `CS_QUEUED`.

Reminder Service Calls

Reminder calls received at ***snapshotObj*** will not be reported by the ***cstaSnapshotDeviceReq( )*** service.

Service Observing

When an application uses the ***cstaSnapshotDeviceReq( )*** service to take a snapshot of a Service Observing station, all eligible calls will be reported including Service Observing calls.

When an application uses the ***cstaSnapshotDeviceReq( )*** service to take a snapshot of a station that is being observed by another station, the status of the call at the Service Observing station will not be reported.

Shared System Access (SSA) Buttons

Calls appearing on SSA buttons at ***snapshotObj*** will not be reported by the ***cstaSnapshotDeviceReq( )*** service.

Call Events

8

Contents

Contents

Contents

Contents

Contents

Contents

Call Events

8

Call Events track telephony activity occurring at a device. Telephony activity may occur as a result of user activity at the device, call activity at the device (for example, an incoming call or the far-end party dropping from a call), or the activity of a CTI application (for example an application dropping a device from a call).

Applications use Call Events to track the activity of a connection, device, or call that is of interest to the application. Since telephony activity can occur at any time, these messages are asynchronous. An application that needs to receive Call Events for a device must:

- Open a stream using the Control Services (Chapter 3);
- Monitor that device using the Monitor Services (Chapter 6);
- Receive events using the Control Services (Chapter 3).

NOTE:
Applications should always be event driven and use TSAPI events to react to telephony activity. An application should never presume a specific call state model. The MERLIN LEGEND and MERLIN MAGIX switches provide many features, and simple call state models will not take into account all feature interactions that may occur. In addition, future releases of the switch may contain new features that interact in ways that the application's call state model did not anticipate.

Table 8-1 shows the TSAPI Call Events that the MERLIN LEGEND and MERLIN MAGIX switches provide. Note that the MERLIN LEGEND and MERLIN MAGIX switches do not provide all of the TSAPI Call Events.

Table 8-1. MERLIN LEGEND and MERLIN MAGIX CTI Support for TSAPI Call Events

	TSAPI Call Events - MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX Release 1.0
	CSTACallClearedEvent
Ö	CSTAConferencedEvent
Ö	CSTAConnectionClearedEvent
Ö	CSTADeliveredEvent
	CSTADivertedEvent
Ö	CSTAEstablishedEvent
	CSTAFailedEvent
Ö	CSTAHeldEvent
Ö	CSTANetworkReachedEvent
	CSTAOriginatedEvent
	CSTAQueuedEvent
Ö	CSTARetrievedEvent
Ö	CSTAServiceInitiatedEvent
Ö	CSTATransferredEvent

	TSAPI Call Events - MERLIN MAGIX (Release 1.5 and later)
	CSTACallClearedEvent
Ö	CSTAConferencedEvent
Ö	CSTAConnectionClearedEvent
Ö	CSTADeliveredEvent
Ö	CSTADivertedEvent
Ö	CSTAEstablishedEvent
	CSTAFailedEvent
Ö	CSTAHeldEvent
Ö	CSTANetworkReachedEvent
	CSTAOriginatedEvent
Ö	CSTAQueuedEvent
Ö	CSTARetrievedEvent
Ö	CSTAServiceInitiatedEvent
Ö	CSTATransferredEvent

CAUTION:

When designing an application, be aware of the event parameters that the MERLIN LEGEND and MERLIN MAGIX switches provide. The MERLIN LEGEND and MERLIN MAGIX switches do not provide all of the optional TSAPI event parameters. Note that the MERLIN LEGEND and MERLIN MAGIX switches do not provide the optional local connection state information. The event manual pages list all of the TSAPI parameters and indicate those that the MERLIN LEGEND and MERLIN MAGIX switches provide.

Many of the call events contain mandatory TSAPI parameters that identify devices. In some situations the device may be a trunk or a Calling Group queue.

- When the MERLIN LEGEND and MERLIN MAGIX switches supply a trunk facility identifier in a call event, the identifier takes the form of the letter “T” followed by the facility identifier for the trunk (for example, “T801”.)
- When the MERLIN MAGIX switch supplies a Calling Group queue identifier in a call event, the identifier takes the form of the letter “Q” followed by the extension number for the Calling Group (for example, “Q770”.)

General Call Event Feature Interactions

There are several important attributes of event reporting that application developers must consider:

- Programming for “busy” conditions (Chapter 2);
- Bridging and Shared Facility Interactions (summarized below and covered more thoroughly in Chapter 2);
- Events that flow when a call is manually originated (below).

Bridging, Coverage and Shared Facility Interactions

For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), Call Event reporting does not track activity on, or interaction with, bridged, coverage or shared appearances. An application monitoring a device where a call alerts on a monitored station and is then answered using a shared facility or cover button from another station is treated as cleared from the station where it initially alerted. Application designers should be aware that interaction with a shared facility may remove such a connection from their control. Whenever this happens, the application receives a ***CSTAConnectionClearedEvent***.

Beginning with MERLIN MAGIX Release 2.0, Call Event reporting will track activity on all buttons except Shared System Access buttons. A call that alerts on a Cover, DFT or DPT button will receive events, including the ***CSTADeliveredEvent***. However, a call that alerts at an SA button on a monitored station and is then answered using a Shared System Access button from another station is treated as cleared from the station where it initially alerted, as in earlier MERLIN LEGEND and MERLIN MAGIX releases.

Chapter 2 provides additional details and Chapter 12 provides example event flows.

Call Event Distribution in MERLIN MAGIX Release 2.0 and later

In general the rules for distributing call events are:

- If a device is a participant on a call, then a monitor for that device will receive events for that call with the following exceptions DFT and DPT Buttons:
  - — A device monitor for a station where a call appears on a DFT or DPT (Pool) button will generally not receive events for call activity at other DFT or DPT buttons.
  - — A device monitor for a station that has received a call as a result of coverage will generally not receive events for call activity at other coverage receivers.

  (These two exceptions were implemented to prevent an application from being bombarded with events that were probably not relevant to the device monitor.)
- If a device appears as event parameter value, then a monitor for that device will receive the event.

Beginning with MERLIN MAGIX Release 2.0, a ***CSTADeliveredEvent*** is generated when a call is delivered to a DFT or DPT button. When there are multiple events generated for a particular call, a monitored station will receive the ***CSTADeliveredEvent*** for its own station. When the call is answered at a station, all other monitored extensions with an appearance of that call will receive a ***CSTAConnectionClearedEvent*** for the appearance for their device and not other devices. When the call is answered, the answering station will receive a ***CSTAEstablishedEvent*** for itself, but this will not be propagated to other monitored stations with the call appearance.

Coverage Buttons

Beginning with MERLIN MAGIX Release 2.0, a ***CSTADeliveredEvent*** when a call is delivered to Cover buttons. When the sender is monitored, it will receive its own ***CSTADeliveredEvent*** and the ***CSTADeliveredEvent*** for all receivers. A receiver will only receive a ***CSTADeliveredEvent*** for itself and not the other receivers.

- If the far end disconnects, the sender will receive a ***CSTAConnection-ClearedEvent*** for itself and all receivers. A receiver will receive a ***CSTAConnectionClearedEvent*** for itself and not other receivers.
- If the call is answered at a receiver, the answering receiver and sender will receive a ***CSTAEstablishedEvent***, but this will not be propagated to other receivers. All other receivers will receive one ***CSTAConnectionCleared-Event*** for the call clearing from their button. This event will also go to the sender.

- If the call is answered at the sender, the sender will receive a ***CSTAEstablishedEvent.*** Receivers will receive one ***CSTAConnection-ClearedEvent*** for their device. The sender will also receive the ***CSTAConnectionClearedEvent*** for each receiver.

When forwarding is active, both the forward-from and in MERLIN MAGIX Release 2.0, Original Caller Information (OCI) is provided in the ***CSTADeliveredEvent***, ***CSTAEstablishedEvent*** and ***CSTAQueuedEvent***. The rules for the parameter generation for these events are similar. These parameters appear in the privateData field of the events.

Event Page Format

The following pages in this chapter present the TSAPI call events that the MERLIN LEGEND and MERLIN MAGIX switches provide to applications. Each TSAPI event description contains the following sections, as appropriate:

Event Name and Description

The event name appears first on the pages describing that event. A description of that event immediately follows the name.

Event Parameters

A table lists the event parameters and summarizes their use.

Event Scenario Diagram

A figure shows the devices, connections, and calls before and after the event. In the diagrams, squares are devices and are labeled D1, D2, etc. Circles are calls and are labeled C1, C2, etc. Lines are connections and their label identifies the device and the call (for example D1C2 would be the connection of device D1 to call C2). The diagrams use the connection state symbols shown in Table 8-2.

Table 8-2. Symbols Used in Event Scenario Figures

Symbol	Connection State
i	Initiated (the extension is hearing dial tone, is in the process of dialing, or has completed dialing but the call has not yet originated)
a	Alerting (often audible ringing, but not necessarily)
c	Connected
h	Held
ht, hc	Held for Transfer, Held for Conference - these used when necessary to distinguish from Held
q	Queued
*	Any non-null state (the call appears at the device, and may be connected, held, held-for-conference, held-for-transfer)

Event Causes

This section lists call event causes that may be present in the event giving the cause of the event. Items in all capitals are #defines from the TSAPI header files acs.h, acsdefs.h, csta.h, and cstadefs.h.

Event Syntax

This section contains C coding information for the event.

Private Data Syntax

This section contains C coding information for any private data that the event may carry. This section is not present if the event may not carry private data.

Important Feature Interactions

This section describes important interactions with the MERLIN LEGEND and/or MERLIN MAGIX switch features that produce the event.

CSTAConferencedEvent

The ***CSTAConferencedEvent*** indicates that station ***confController*** has conferenced a new connection onto a call. Specifically, the ***confController*** station had the connection ***primaryOldCall*** on hold and the connection ***secondaryOldCall*** active and then conferenced those two connections together. The MERLIN LEGEND and MERLIN MAGIX switches provide the ***CSTAConferencedEvent*** event when a conference operation occurs at any facility type on a monitored station.

In a typical conference call scenario, the ***confController*** places the ***primaryOldCall*** on hold-for-conference, then originates a call (the ***secondaryOldCall***) to ***addedParty***, and then conferences the calls.

Event Parameters

Table 8-3. CSTAConferencedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_CONFERENCED`
monitorCrossRefID	event occurred on this monitor
primaryOldCall	connection that was held for conference
secondaryOldCall	connection that was active for conference
confController	conferencing device
addedParty	Device being added. If the deviceID for the device being added is not known, then the `deviceIDStatus` component has a value of `ID_NOT_KNOWN`.
conferenceConnections	List of connections on the conference call. Each connection contains a device identifier and a call identifier.
localConnectionInfo	`CS_NONE`, none provided
cause	reason for Conferenced event
privateData	`NULL`, not used for this event

addedParty is a device identifier giving the device added to the call. When the newly added party is a station, ***addedParty*** contains the extension for that station. When the newly added party is a trunk connection, ***addedParty*** contains the MERLIN LEGEND or MERLIN MAGIX switch Facility Identifier for the trunk or the dialed digits. The MERLIN LEGEND switch always supplies the trunk identifier or dialed digits, never a pool or DGC identifier.

conferenceConnections provides applications with information so that they may continue to track calls when call identifiers change as conferencing merges calls together. When a trunk connection is a party to the conference, the ***conferenceConnections*** contains the MERLIN LEGEND or MERLIN MAGIX switch Facility Identifier for the trunk, the dialed digits, ICLID or DNIS information. The MERLIN LEGEND and MERLIN MAGIX switches always supply the trunk identifier, never a pool or DGC identifier. Each ***conferenceConnections*** list member contains:

- a device identifier for a party on the call,
- the connection identifier for the call at that device after the conference occurred.

NOTE:
An application should always check ***conferenceConnections*** to track connection and call identifiers as conferences occur. Currently, in MERLIN LEGEND and MERLIN MAGIX CTI, a conference of the ***primaryOldCall*** and the ***secondaryOldCall*** always results in the ***secondaryOldCall*** being the call identifier for the resulting conference call; there is no guarantee that this will continue to be true in future releases. In addition, not all switches operate in this manner, so a switch-independent application must use the ***conferenceConnections*** to track connection identifiers and call identifiers.

Event Scenario Diagram

Figure 8-1 illustrates one possible ***CSTAConferencedEvent*** scenario.

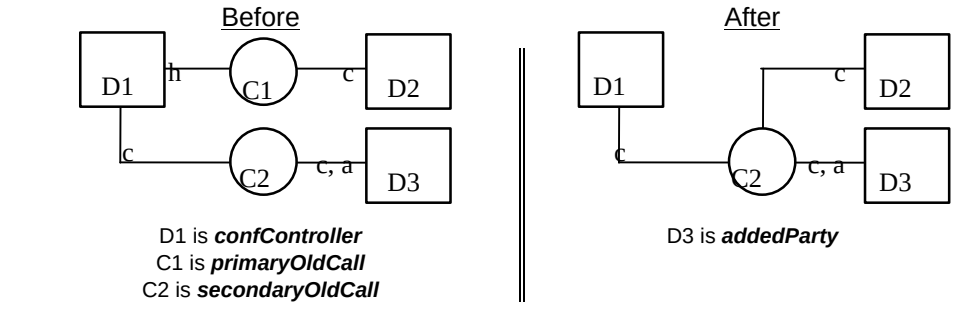


Figure 8-1. CSTAConferencedEvent Scenario

Event Causes

Table 8-4. CSTAConferencedEvent Causes

EC_NEW_CALL	The MERLIN LEGEND and MERLIN MAGIX switches provide EC_NEW_CALL on all ***CSTAConferencedEvents***.

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAConferencedEvent_t  conferenced;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAConferencedEvent_t {
   ConnectionID_t          primaryOldCall;
   ConnectionID_t          secondaryOldCall;
   SubjectDeviceID_t       confController;
   SubjectDeviceID_t       addedParty;
   ConnectionList_t        conferenceConnections;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTAConferencedEvent_t;
```

Important Feature Interactions

Barge-In

Barge-In is a form of bridging operation and does not generate a ***CSTAConferencedEvent***. A station that has Barged-In prior to a conference operation is not included in the connection list.

Bridging

Bridging operations of any type do not generate a ***CSTAConferencedEvent***.

Call Screening

An application monitoring a station will not receive a ***CSTAConferencedEvent*** when a Call Screener joins an existing call.

A station that is screening a call prior to a conference operation is not included in the connection list.

Coverage

When an alerting call to a Coverage sender is added to a conference, the extension number of the Coverage sender is included in Conference Connections List. The extension numbers of the Coverage receivers where the call is also alerting are not included in the Conference Connections List.

Group Calling (DGC)

When ***secondaryOldCall*** is a call to a Calling Group and the call is delivered directly to a Calling Group member (without being queued), the ***addedParty*** parameter in the ***CSTAConferencedEvent*** will contain the extension of the Calling Group member.

A call in a Calling Group queue may not be conferenced.

Networking

An application monitoring the conference originator when the added party is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network will receive a ***CSTAConferencedEvent*** identifying the connections on the conference call.

An application monitoring the added party when the conference originator is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network, will receive a ***CSTADeliveredEvent*** that does not contain Original Call Information. The application will not receive a ***CSTAConferencedEvent***.

Pool

When a user conferences with a call on a Pool button, the ***addedParty*** parameter in the ***CSTAConferencedEvent*** will always contain an individual trunk identifier or dialed digits for an outgoing call, not the Pool extension. Similarly, the ***conferenceConnections*** parameter contains an individual trunk identifier, not the Pool extension.

QCC

When user conferences a QCC onto a call, the ***conferenceConnections*** parameter in the ***CSTAConferencedEvent*** will not contain the QCC.

Service Observing

An application monitoring a station that is being observed will not receive a ***CSTAConferencedEvent*** when the service observer joins an existing call.

A station that has observed a device prior to a conference operation is not included in the connection list.

CSTAConnectionClearedEvent

The ***CSTAConnectionClearedEvent*** indicates that station ***releasingDevice*** disconnected from ***droppedConnection***.

- For MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5), so long as a call remains at one SA button at a monitored device, the switch does not send this event for the call (with respect to the monitored device). The switch will send the ***CSTAConnectionClearedEvent*** only after call clears from all the SA buttons. The ***CSTAConnectionClearedEvent*** is sent even though the call may continue to appear at the device on some other button type such as a DFT, DPT or Cover button.
- For Conference calls, if a user at a monitored station creates a three-way conference call (the call appears on two SA buttons) and one far-end devices drops from the call, the application monitoring the Conference originating station will receive a ***CSTAConnectionClearedEvent*** only for the device that dropped from the call. The monitoring application will receive another ***CSTAConnectionClearedEvent*** when the monitored device drops from the call.

In MERLIN LEGEND and MERLIN MAGIX CTI, a connection ID contains a callID that uniquely identifies a call within the switch. Similarly, a deviceID uniquely identifies a device within the switch. Since ***droppedConnection*** is a connectionID (containing both callID and deviceID), the ***releasingDevice*** parameter is redundant. However, both of these parameters are mandatory in CSTA, so they appear in the event.

NOTE:
Not all switches use static, unique device identifiers. Use the ***releasingDevice*** parameter, not the deviceID within the ***droppedConnection*** parameter to obtain the deviceID of the device that has been disconnected. This will assist in making the application switch-independent.

Prior to MERLIN MAGIX Release 2.2, the MERLIN LEGEND and MERLIN MAGIX CTI implementations do not generate a ***CSTAConnectionClearedEvent*** when a trunk drops off of a call. If a trunk drop results in a call being torn down at a monitored station, then the MERLIN LEGEND or MERLIN MAGIX switch will generate a ***CSTAConnectionClearedEvent*** when the connection is cleared at the monitored station.

Beginning with MERLIN MAGIX Release 2.2, a ***CSTAConnectionClearedEvent*** is provided when a trunk with disconnect supervision drops off of a conference call, or when the Selective Drop feature is used to drop an external party from a conference call.

NOTE:
The ***cstaGetAPICaps*** query does not distinguish between providing this event for local monitored stations and trunk endpoints. The ***cstaGetAPICaps*** response indicates that the MERLIN LEGEND and MERLIN MAGIX switches provide this event. Programmers must understand the limitation in the ***cstaGetAPICaps*** response and not program applications to expect a ***CSTAConnectionClearedEvent*** for the far end on an outbound trunk call.

NOTE:
In this chapter, the paragraph titled "Call Event Distribution in MERLIN MAGIX Releases 2.0 and later" explains how the ***CSTAConnection-ClearedEvent*** is generated for calls appearing on multiple stations.

The MERLIN LEGEND and MERLIN MAGIX switches will send a ***CSTA-ConnectionClearedEvent*** for a monitored station from any facility type except a Loop button.

Event Parameters

Table 8-5. CSTAConnectionClearedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_CONNECTION_CLEARED`
monitorCrossRefID	event occurred on this monitor
droppedConnection	connection that cleared (contains deviceID and callID)
releasingDevice	device where connection cleared
localConnectionInfo	`CS_NONE`, none provided
cause	reason for Connection Cleared event
privateData	(private data version 2 only) may contain an account code

Beginning with MERLIN MAGIX Release 2.1, it is recommended that applications obtain account code information using the ***CSTACallInfoEvent*** rather than the Private Data in the ***CSTAConnectionClearedEvent***.

Event Scenario Diagram

Figure 8-2 illustrates one possible ***CSTAConnectionClearedEvent*** scenario.

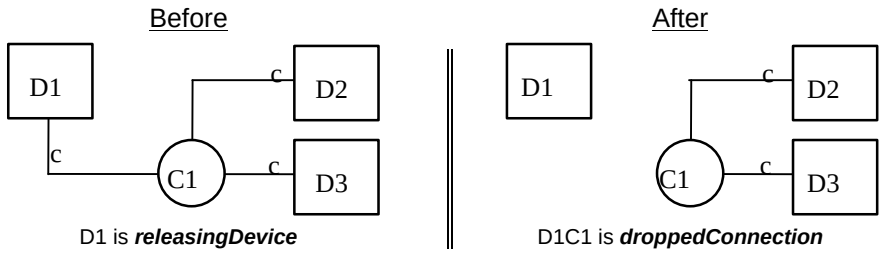


Figure 8-2. CSTAConnectionClearedEvent Scenario

Event Causes

Table 8-6. MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) CSTAConnectionClearedEvent Causes

EC_CALL_CANCELLED	The connection has dropped from the monitored device and the monitored device is not on hook.
EC_NONE	The connection has dropped at the monitored device and the monitored device is on hook.

Table 8-7. MERLIN MAGIX Release 2.0 and Later CSTAConnectionClearedEvent Causes

EC_CALL_CANCELLED	Remote end hangs up.
EC_CALL_NOT_ANSWERED	The connection is a refused Calling Group call (i.e., the call was alerting at a Calling Group member but was returned to the queue), or was redirected from a station via the ***cstaDeflectCall( )*** service.
EC_NONE	The user hangs up and the monitored device is on hook.
EC_SILENT_MONITOR	The connection has dropped from a monitored Call Screener or Service Observer on the call.

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {

      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAConnectionClearedEvent_t  connectionCleared;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAConnectionClearedEvent_t {
   ConnectionID_t          droppedConnection;
   SubjectDeviceID_t       releasingDevice;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTAConnectionClearedEvent_t;
```

Private Data Versions 2 and 3 Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the MERLIN MAGIX private data header files (mlpriv.h and mlpdefs.h) for a complete description.

```
typedef struct {
   MLEventType_t  eventType;     /* ML_CONNECTION_CLEARED */
   union {
      /* Only the pertinent union element is shown */
      MLConnectionClearedEvent_t    connectionClearedEvent;
   } u;
} MLEvent_t;

typedef struct MLConnectionClearedEvent_t {
   char     accountCode[17];
} MLConnectionClearedEvent_t;
```

Important Feature Interactions

Once the application receives a ***CSTAConnectionClearedEvent*** for a call at a device, the ***cstaAnswerCall( )*** service may not be used to answer another appearance of that call at the device.

Account Code

An application monitoring an extension that has made or received an external call (i.e. one involving a trunk) where an account code has been entered will receive the account code (beginning with Private Data version 2) in the ***CSTAConnectionClearedEvent***.

Beginning with MERLIN MAGIX Release 2.1, it is recommended that applications obtain account code information using the ***CSTACallInfoEvent*** rather than the Private Data in the ***CSTAConnectionClearedEvent***.

Call Pickup

Beginning with MERLIN MAGIX Release 1.5, an application monitoring an extension from which a call is picked up will receive a ***CSTAConnectionCleared-Event*** for the call.

An application monitoring an extension that performs a call pick (extension, line or group) will receive a ***CSTAConnectionClearedEvent*** for the “call” that was used to invoke the Pickup feature.

Call Screening

A device monitor for the extension of a Call Screener will receive a ***CSTA-ConnectionClearedEvent*** when the Call Screener drops off of a screened call. The ***cause*** in the event will be `EC_SILENT_MONITOR`. Device monitors for other extensions on the call will not receive this event.

An application monitoring a Voice Mail port will receive a ***CSTAConnection-ClearedEvent*** when a Call Screener joins the call as a regular call participant, causing the Voice Mail port to be dropped from the call. The ***cause*** in the event will be `EC_CALL_CANCELLED`.

Conferencing

An application monitoring an extension with a connection to a conference call will receive a ***CSTAConnectionClearedEvent*** when an internal conference participant drops from the conference.

An application monitoring an extension with a connection to a conference call that is not the conference originator will receive a ***CSTAConnectionClearedEvent*** when the monitored extension drops from the conference.

Prior to MERLIN MAGIX Release 2.2, an application monitoring an extension with a connection to a conference call will not receive a ***CSTAConnectionCleared-Event*** when an external conference participant drops from the conference.

Beginning with MERLIN MAGIX Release 2.2, an application monitoring an extension with a connection to a conference call will receive a ***CSTAConnection-ClearedEvent*** when an external conference participant drops from the conference if the trunk used by the external conference participant provides disconnect supervision.

The conference originator's station for a multi-party conference call will have appearances of that conference call on multiple buttons. An application monitoring the conference originator's station will receive a ***CSTAConnectionClearedEvent*** when the last appearance for the conference call clears.

Coverage

When a call is alerting on an SA button at a monitored station, and another station answers that call using a COVER button, then an application monitoring the station where the call is alerting on the SA button receives a ***CSTAConnection-ClearedEvent***.

Beginning with MERLIN MAGIX Release 2.0, when a call is alerting on a COVER button at a monitored station, and an SA or COVER button at some other station answers the call, the call is cleared from the COVER button and an application monitoring the station with the COVER button receives a ***CSTAConnection-ClearedEvent***.

When an application is monitoring a Cover sender, the application will receive call events for the Cover sender as well as call events for each of the Coverage receivers. An application monitoring a Coverage receiver will only receive call events for the Coverage receiver.

Delay Announcement Unit

Beginning in MERLIN MAGIX Release 2.0, when a call that is alerting at a monitored Delay Announcement Unit is returned to the queue or is redirected to a Calling Group member, an application monitoring the Delay Announcement Unit receives a ***CSTAConnectionClearedEvent*** with a ***cause*** of `EC_CALL_NOT_ANSWERED`.

DFT/DPT

When a call alerts on an SA button at a monitored station, and a DFT or DPT button at some other station answers the call, then an application monitoring either the station with the SA button or the station with the DFT/DPT button receives a ***CSTAConnectionClearedEvent***.

Beginning with MERLIN MAGIX Release 2.0, when a call alerts on a DFT or DPT button at a monitored station, and the call is answered at another station on a supported button, then the application monitoring the station with the DFT or DPT button receives a ***CSTAConnectionClearedEvent***.

When a call appears at an extension on a DFT or DPT, there are cases where the call can also appear at the extension on other buttons (SA button or COVER button).

When a call alerts on an SA button of a monitored station, and also alerts on a DFT or DPT button at the same station two ***CSTADeliveredEvents*** are generated for the same call. When the call is eventually cleared at the station only one ***CSTAConnectionClearedEvent*** is generated.

Direct Voice Mail

When an external call is transferred to a station's mailbox using Direct Voice Mail, the ***CSTATransferredEvent*** contains the extension number of the station in the list of transferred connections even though the station is not on the call. A ***CSTAConnectionClearedEvent*** is then generated to indicate that the station is not on the call.

Drop

An application monitoring an extension with a connection to a conference call will receive a ***CSTAConnectionClearedEvent*** when the Selective Drop feature is used to drop an internal conference participant from the call.

Prior to MERLIN MAGIX Release 2.2, an application monitoring an extension with a connection to a conference call will not receive a ***CSTAConnectionCleared-Event*** when the Selective Drop feature is used to drop an external conference participant from the call.

Beginning with MERLIN MAGIX Release 2.2, an application monitoring an extension with a connection to a conference call will receive a ***CSTAConnection-ClearedEvent*** when the Selective Drop feature is used to drop an external conference participant from the call.

Forward/Follow Me

Beginning with MERLIN MAGIX Release 2.0, when a forwarded call is alerting at the forwarded-from station, and the forward-to station connects to that call, the call is cleared from the forwarded-from station and an application monitoring any station on the call receives a ***CSTAConnectionClearedEvent***.

Beginning with MERLIN MAGIX Release 2.0, when a forwarded call is alerting at the forward-to station, and the forwarded-from station answers the call, the call is cleared from the forwarded-to station and an application monitoring any station on the call receives a ***CSTAConnectionClearedEvent***.

Group Calling (DGC)

Beginning with MERLIN MAGIX Release 1.5, when a call is in a monitored Calling Group queue and the far end disconnects, then the application monitoring the queue receives a ***CSTAConnectionClearedEvent.*** The ***releasingDevice*** in the event is the Calling Group number.

Beginning with MERLIN MAGIX Release 2.0, when a call that had been alerting at a monitored Calling Group member returns to the queue (i.e., it is a refused call) or is redirected through the ***cstaDeflectCall( )*** service, an application monitoring the Calling Group member receives a ***CSTAConnectionCleared-Event*** with a ***cause*** of `EC_CALL_NOT_ANSWERED`.

QCC

An application cannot monitor a QCC. The MERLIN LEGEND and MERLIN MAGIX switches do not provide the ***CSTAConnectionClearedEvent*** for QCC facilities.

Service Observing

In a MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application monitoring any station participating in an observed call will receive a ***CSTAConnectionClearedEvent*** when the observer drops off the call.

Beginning with MERLIN MAGIX Release 2.0, only an application monitoring the service observer will receive a ***CSTAConnectionClearedEvent*** when the observer drops off the call. The ***cause*** in the ***CSTAConnectionClearedEvent*** will be `EC_SILENT_MONITOR`.

An application monitoring the station of a service observer will receive a ***CSTA-ConnectionClearedEvent*** for the call associated with activating the Service Observing feature, and will also receive a ***CSTAConnectionClearedEvent*** whenever an observed call is disconnected.

Shared Facility Interactions

In a MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) environment, when a call alerting on an SA button of a monitored station is answered on some shared facility (SSA, BA, Cover, DFT, etc.), the monitoring application receives a ***CSTAConnectionClearedEvent*** for the call.

An application will not receive a ***CSTAConnectionClearedEvent*** when a call has been answered at a DFT, DPT or cover button.

Beginning with MERLIN MAGIX Release 2.0, when a call alerting on an SA, Cover, DFT, or DPT button of a monitored station is answered on some shared facility (SSA, BA, Cover, DFT, etc.) at another station, the monitoring application receives a ***CSTAConnectionClearedEvent*** for the call.

CSTADeliveredEvent

The ***CSTADeliveredEvent*** indicates that a call (possibly a consultation call) is alerting at a station.

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, the switch only provides a ***CSTADeliveredEvent*** for a call alerting on an SA button on a station. Beginning with MERLIN MAGIX Release 2.0, the switch provides a ***CSTADeliveredEvent*** for a call alerting at an SA, Cover, DFT or DPT button on a station.

Event Parameters

Table 8-8. CSTADeliveredEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_DELIVERED`
monitorCrossRefID	event occurred on this monitor
connection	delivered connection (contains deviceID and callID)
alertingDevice	device where connection delivered
callingDevice	the calling device may contain a number identifying the calling party number. See below for details.
calledDevice	the called device may contain a number identifying the called party number. See below for details.
lastRedirectionDevice	For MERLIN MAGIX Release 2.0 and later, the last redirection device for the call, when applicable. See below for details.
localConnectionInfo	`CS_NONE`, not provided
cause	reason for Delivered event (See Tables 8-9, 8-10)
privateData	may contain call prompting digits, original call information, and/or (private data version 2 and later) the trunk identifier for the call

The ***callingDevice*** parameter contains the ANI/ICLID for an external party (when the trunk provides it) or the extension for a local party. CSTA permits values indicating “unknown” for certain ***CSTADeliveredEvent*** parameters in certain circumstances. When an incoming call arrives on a trunk that does not provide ANI/ICLID, the ***callingDevice*** has a `deviceIDStatus` of `ID_NOT_KNOWN`.

IMPORTANT:
For a **CSTADeliveredEvent** *event to provide the calling number for an incoming external call, the external call must arrive on either:*

- *PRI/BRI facilities provisioned to provide ANI.*
- *trunks that have ICLID-Delay applied by the switch. Typically a call on a facility alerting into a Calling Group would be delayed being delivered to an extension until the ICLID information arrived.*

When an incoming call alerts on a PRI/BRI trunk provisioned to provide DNIS, the ***calledDevice*** parameter contains the PRI Called Number. Prior to MERLIN MAGIX Release 2.1, the ***calledDevice*** parameter matches the ***alertingDevice*** parameter for all other cases. Note that the parameter does not necessarily indicate the device called by the ***callingDevice***.

Beginning with MERLIN MAGIX Release 2.0, a ***CSTADeliveredEvent*** may be delivered to an application for outgoing calls. This will happen only if the call is a PRI call involving all digital lines. The switch populates the TSAPI ***calledDevice*** parameter to identify the device being called. An application monitoring an extension where the user makes a PRI call will receive a ***CSTADeliveredEvent*** when the switch receives a message that the far-end is alerting. The ***calledDevice*** will be the dialed number, which may or may not match the alerting device number. All other outgoing calls do not generate a ***CSTADeliveredEvent***.

Beginning with MERLIN MAGIX Release 2.1, the ***calledDevice*** parameter for incoming external calls is populated with one of the following:

- the called number from the ISDN setup message for calls over PRI facilities
- a `deviceIDStatus` of `ID_NOT_KNOWN` for DFT/DPT calls over non-PRI facilities
- the Calling Group Queue for Calling Group calls arriving non-PRI facilities and where the facilities do not also terminate on DFT/DPT buttons
- a `deviceIDStatus` of `ID_NOT_KNOWN` for DGC calls over non-PRI facilities that terminate on DFT/DPT buttons

For intercom calls, the ***calledDevice*** parameter is populated with one of the following:

- The called extension number for a simple station to station call
- The forwarding station for forwarded calls including forwarded on busy
- The coverage sender for coverage calls including Calling Group coverage
- The Calling Group Queue for DGC calls
- The station extension where a call is being picked up from, using the call pickup feature

As a call redirects (coverage, forwarding, etc.) from its original destination to other endpoints, the ***calledDevice*** for an incoming PRI or BRI call remains static, and the ***alertingDevice*** parameter contains the extension of the device where the call is alerting.

Beginning with MERLIN MAGIX Release 2.1, the ***calledDevice*** for all incoming calls remains static.

Prior to MERLIN MAGIX Release 2.0, switches do not populate the TSAPI ***lastRedirectionDevice*** parameter. This parameter always has the `deviceIDStatus` component set to `ID_NOT_KNOWN`.

In MERLIN MAGIX Release 2.0, the switch populates the TSAPI ***lastRedirectionDevice*** parameter as follows:

- If the call is a DGC call alerting at a station that is a Calling Group member, the ***lastRedirectionDevice*** contains the number of the Calling Group of which the station is a member. The Calling Group for the call and the alerting station may be different.
- Otherwise,
  - If the call is alerting at a Cover button, the ***lastRedirectionDevice*** contains the extension of the coverage sender.
  - If the call is an internal call alerting at a forward-to station, the ***lastRedirectionDevice*** contains the forward-from extension. The MERLIN MAGIX switch does not provide ***lastRedirectionDevice*** the for a call forwarded from a DFT or DPT button.
  - Otherwise, the `deviceIDStatus` component has a value of `ID_NOT_KNOWN`

Beginning with MERLIN MAGIX Release 2.1, the switch populates the TSAPI ***lastRedirectionDevice*** parameter as follows:

- If the call is a DGC call alerting at a station that is a Calling Group member, the ***lastRedirectionDevice*** contains the number of the Calling Group of which the call came into Otherwise,
  - If the call is alerting at a Cover button, the ***lastRedirectionDevice*** contains the extension of the coverage sender.
  - If the call is internal call alerting at a forward-to station, the ***lastRedirectionDevice*** contains the forward-from extension. The MERLIN MAGIX switch does not provide ***lastRedirectionDevice*** the for a call forwarded from a DFT or DPT button.
  - Otherwise, the `deviceIDStatus` component has a value of `ID_NOT_KNOWN`

Prior to MERLIN MAGIX Release 2.0, the TSAPI ***cause*** parameter is always populated with `EC_NEW_CALL`.

Beginning with MERLIN MAGIX Release 2.0, the switch populates the TSAPI ***cause*** parameter (the precedence is the presented order) as follows:

- If the call is a DGC call alerting at a station that is a Calling Group member, the ***cause*** is `EC_REDIRECTED`.
- If the call is a transfer, park or camp-on return call, the ***cause*** is `EC_RECALL`.
- If the call is alerting at a Cover button, the ***cause*** is `EC_CALL_FORWARD`.
- If the call is alerting at a forward-to station (for non-DFT/DPT calls), the ***cause*** is `EC_CALL_FORWARD`.
- If the call is an outbound PRI call, the ***cause*** is `EC_NONE`.
- If the call is alerting as a result of a Voice Announced transfer the cause is `EC_TRANSFER`.
- For all other cases, the ***cause*** is `EC_NEW_CALL`.

Beginning with MERLIN MAGIX Release 2.1 the ***cause*** parameter is populated the same as for MERLIN MAGIX Release 2.0 except that the ***cause*** is `EC_CALL_FORWARD_ALWAYS` if the call is alerting at a forward-to station for non-DFT/DPT calls (MERLIN MAGIX Release 2.0 uses `EC_CALL_FORWARD`).

The ***CSTADeliveredEvent*** may contain private data that carries:

- Any collected digits that have been associated with the call – If the call is an incoming call and has been routed through a VMI port prompted digits may have been collected
- Information about the original call – When an application uses ***cstaConsultationCall( )*** to extend a call, information about the original call is provided in private data. This “original call information” about the transfer source [1] appears in any ***CSTADeliveredEvent***s for the consultation call. An application at the desktop receiving the consultation call can use the original calling number, original PRI Called Number (DNIS), or original call prompter digits to pop an appropriate screen. See the section “MERLIN LEGEND and MERLIN MAGIX Private Data Libraries” in Chapter 2.

Beginning with private data version 2 and MERLIN MAGIX Release 2.0, private data in the ***CSTADeliveredEvent*** may also contain the trunk identifier (e.g. “T802”) associated with an external call.

---

1 The MERLIN LEGEND and MERLIN MAGIX switches use the following terms in a transfer scenario: When a call is being transferred, the party doing the transferring is the *transfer originator*. The party being transferred is the *transfer source*. The party receiving the transferred call is the *transfer destination*. Thus, the ***activeCall*** parameter in a ***cstaConsultationCall( )*** is a connection at the transfer originator for the call at the transfer source. The ***calledDevice*** parameter in a ***cstaConsultationCall*** specifies the transfer destination.

In MERLIN LEGEND and MERLIN MAGIX CTI, a connection ID contains a callID that uniquely identifies a call within the switch. Similarly, a deviceID uniquely identifies a device within the switch. Since ***connection*** is a connectionID (containing both callID and deviceID), the ***alertingDevice*** parameter is redundant. However, both of these parameters are mandatory in CSTA so they must be present.

NOTE:
Not all switches use static, unique device identifiers. Use the ***alertingDevice*** parameter, not the deviceID within the ***connection*** parameter to obtain the deviceID for the alerting device. This will assist in making the application switch-independent.

NOTE:
The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches do not provide the ***CSTADeliveredEvent*** event for the outbound leg of a call, leaving the switch on a trunk.[2] Note that the ***cstaGetAPICaps( )*** query does not distinguish between providing this event for a local monitored station and a trunk endpoint. The ***cstaGetAPICaps( )*** response indicates that the MERLIN LEGEND and MERLIN MAGIX switches provide this event. Programmers must understand the limitation in the ***cstaGetAPICaps( )*** response and not program applications to expect a ***CSTADeliveredEvent*** for the far end on an outbound trunk call.

Beginning with MERLIN MAGIX Release 2.0, the switch will provide the ***CSTADeliveredEvent*** event for the outbound leg of a call, leaving the switch on a PRI trunk when the switch receives a message that the far-end is ringing. (This requires that the call be routed only on digital facilities.)

NOTE:
The ***CSTADeliveredEvent*** event is not generated for Calling Group calls that go over the private network.

2 PRI trunking provides the switch with signaling information that the switch can use to generate this event. Generation of a Delivered event in this circumstance is provided beginning with MERLIN MAGIX Release 2.0.

Event Scenario Diagram

Figure 8-3 illustrates one possible ***CSTADeliveredEvent*** scenario.

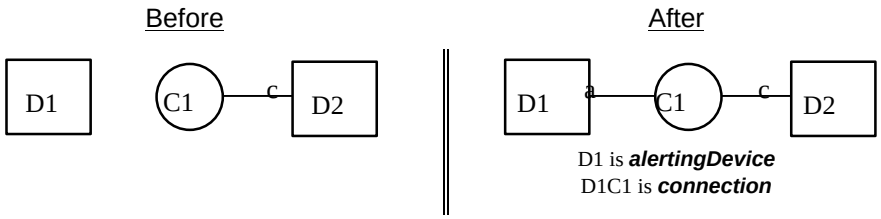


Figure 8-3. CSTADeliveredEvent Scenario

Event Causes

Table 8-9. MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) CSTADeliveredEvent Causes

EC_NONE	The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches provide this cause in all ***CSTADeliveredEvents***.

Table 8-10. MERLIN MAGIX Release 2.0 and 2.1 CSTADeliveredEvent Causes

EC_NEW_CALL	The MERLIN MAGIX Release 2.0 and 2.1 switches provides this cause in all cases except those specified below:
EC_CALL_FORWARD_ALWAYS	the call alerting at ***alertingDevice*** has been forwarded via the Call Forwarding feature (Beginning with MERLIN MAGIX Release 2.1)
EC_CALL_FORWARD	the call alerting at ***alertingDevice*** has been forwarded via the Call Forwarding (MERLIN MAGIX Release 2.0) or Coverage feature
EC_RECALL	the call alerting at ***alertingDevice*** is a transfer, park or camp-on return call
EC_REDIRECTED	***alertingDevice*** is a Calling Group member and the call is a DGC call
EC_NONE	for outgoing digital PRI calls

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTADeliveredEvent_t delivered;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTADeliveredEvent_t {
   ConnectionID_t          connection;
   SubjectDeviceID_t       alertingDevice;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   RedirectionDeviceID_t   lastRedirectionDevice;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTADeliveredEvent_t;
```

Private Data Parameters

Table 8-11. CSTADeliveredEvent Private Data Parameters

userEnteredCode	Specifies the code/digits that may have been entered by the caller through the Collected Digits feature. If the ***userEnteredCode*** type is set to "**ML_UE_NONE**", no Collected Digits are provided with this event. If the ***userEnteredCode*** type is set to "**ML_CALL_PROMPTER**," ***userEnteredCode*** Collected Digits are provided with this event. See the MERLIN LEGEND Advanced Communications System Feature Reference or MERLIN MAGIX Integrated System Feature Reference (in the CTI Link Section) for information on how to set up the switch and application for collecting ***userEnteredCode*** through the Collected Digits feature.

originalCallInfo Specifies the original call information. Note that information is not repeated in the ***originalCallInfo*** if it is already reported in the CSTA service parameters. For example, the ***callingDevice*** and ***calledDevice*** in the ***originalCallInfo*** will not be set if the ***callingDevice*** and the ***calledDevice*** in the CSTA service parameters are the original calling and called devices. The ***callingDevice*** and ***calledDevice*** in the ***originalCallInfo*** will be set only when the original devices are different from the most recent ***callingDevice*** and ***calledDevice***.

NOTE:

For the Delivered Event corresponding to the ***newCall*** of a Consultation Call, the ***originalCallInfo*** is taken from the ***activeCall*** specified in the Consultation Call request. Thus the application can pass the original call information between two calls. The ***calledDevice*** of the Consultation Call must reside on the same switch and must be monitored via the same Tserver.

- ***reason*** — the reason for the ***originalCallInfo***. The following reasons are supported.

  ML_OR_NONE — no ***originalCallInfo*** provided

  ML_OR_CONSULTATION — ***originalCallInfo*** provided

- ***callingDevice*** — the original ***callingDevice*** received by the ***activeCall***.
- ***calledDevice*** — the original ***calledDevice*** received by the ***activeCall***.

NOTE:

In MERLIN MAGIX Release 2.0, ***originalCallInfo*** is also provided for calls that have been redirected due to Forwarding or Coverage. In this case, the ***reason*** for the ***originalCallInfo*** gives no indication that the ***originalCallInfo*** is due to Forwarding or Coverage. However, the ***cause*** in the Delivered event is `EC_CALL_FORWARD`.

Beginning with MERLIN MAGIX Release 2.1, changes to the ***calledDevice*** parameter eliminate the need for ***originalCallInfo*** in Forwarding and Coverage scenarios. Also, the ***cause*** parameter provides a distinction between Coverage and Forwarded calls. For Coverage calls, the ***cause*** is `EC_CALL_FORWARD` (as before), and for Forwarded calls, the ***cause*** is `EC_CALL_FORWARD_ALWAYS`.

trunkUsed	Available beginning with private data Version 2. Contains the trunk identifier (e.g. “T801”) when the call involves a trunk

Private Data Versions 2 and 3 Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the MERLIN MAGIX private data library header files (mlpriv.h and mlpdefs.h) for a complete description.

```
typedef struct {
   MLEventType_t  eventType;     /* ML_DELIVERED */
   union {
      /* Only the pertinent union element is shown */
      MLDeliveredEvent_t   deliveredEvent;
   } u;
} MLEvent_t;

typedef struct MLDeliveredEvent_t {
   MLUserEnteredCode_t     userEnteredCode;
   MLOriginalCallInfo_t    originalCallInfo;
   DeviceID_t              trunkUsed;
} MLDeliveredEvent_t;

/*
 * Note: ML_MAX_USER_CODE is defined in mlpriv.h to be the
 * maximum length of the collected digit string.
 */

typedef struct MLUserEnteredCode_t {
   MLUserEnteredCodeType_t type;
   char                    data[ML_MAX_USER_CODE];
} MLUserEnteredCode_t;

typedef enum MLUserEnteredCodeType_t {
   ML_UE_NONE = -1,        /* no collected digits */
   ML_CALL_PROMPTER = 5    /* collected digits */
} MLUserEnteredCodeType_t;

typedef struct MLOriginalCallInfo_t {
   MLReasonForCallInfo_t   reason;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   MLUserEnteredCode_t     userEnteredCode;
} MLOriginalCallInfo_t;

typedef enum MLReasonForCallInfo_t {
   ML_OR_NONE = 0,         /* no OCI present */
   ML_OR_CONSULTATION = 1  /* OCI present */
} MLReasonForCallInfo_t;
```

Private Data Version 1 Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the MERLIN MAGIX private data library header files (mlpriv.h and mlpdefs.h) for a complete description.

```
typedef struct {
   MLEventType_t  eventType;     /* MLV1_DELIVERED */
   union {
      /* Only the pertinent union element is shown */
      MLV1DeliveredEvent_t v1deliveredEvent;
   } u;
} MLEvent_t;

typedef struct MLV1DeliveredEvent_t {
   MLUserEnteredCode_t     userEnteredCode;
   MLOriginalCallInfo_t    originalCallInfo;
} MLV1DeliveredEvent_t;

/*
 * Note: ML_MAX_USER_CODE is defined in mlpriv.h to be the
 * maximum length of the collected digit string.
 */

typedef struct MLUserEnteredCode_t {
   MLUserEnteredCodeType_t type;
   char                    data[ML_MAX_USER_CODE];
} MLUserEnteredCode_t;

typedef enum MLUserEnteredCodeType_t {
   ML_UE_NONE = -1,        /* no collected digits */
   ML_CALL_PROMPTER = 5    /* collected digits */
} MLUserEnteredCodeType_t;

typedef struct MLOriginalCallInfo_t {
   MLReasonForCallInfo_t   reason;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   MLUserEnteredCode_t     userEnteredCode;
} MLOriginalCallInfo_t;

typedef enum MLReasonForCallInfo_t {
   ML_OR_NONE = 0,         /* no OCI present */
   ML_OR_CONSULTATION = 1  /* OCI present */
} MLReasonForCallInfo_t;
```

Important Feature Interactions

Auto Answer All - AAA (ATL Only)

An application will receive a ***CSTADeliveredEvent*** when the Auto Answer All feature answers a call at a monitored device. ATL sets are discontinued beginning in MERLIN MAGIX Release 1.5.

Auto Answer Intercom - AAI (ATL Only)

An application will receive a ***CSTADeliveredEvent*** when the Auto Answer Intercom feature answers a call at a monitored device. ATL sets are discontinued beginning in MERLIN MAGIX Release 1.5.

Call Screening

An application monitoring the station of a Call Screener will not receive a ***CSTA-DeliveredEvent*** when a screened call is presented at the station because the call does not alert. The application will receive a ***CSTAEstablishedEvent***.

Call Waiting

When a call waits at a monitored station, monitoring applications will not receive a ***CSTADeliveredEvent*** when the call waits. They will receive a ***CSTADeliveredEvent*** when the waiting call begins to alert at the station. There is no special information in the ***CSTADeliveredEvent*** to identify a call as a waiting call.

Callback (CBQ)

An application monitoring a station that invokes the Callback feature will not receive a ***CSTADeliveredEvent*** when the call leaves the Callback queue and priority rings the invoking station. After the invoking station picks up and the call then alerts the destination, monitoring applications will receive a ***CSTADeliveredEvent*** for the delivery of the call to an SA button on the originally called station. There is no special information in the ***CSTADeliveredEvent*** to identify a call as a Callback call.

Camp-On

When a Camp-On call completes to an SA button on the originally called extension, monitoring applications will receive a ***CSTADeliveredEvent***.

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, there is no special information in the ***CSTADelivered-Event*** to identify a call as a Camp-On return call.

Beginning with MERLIN MAGIX Release 2.0, the ***cause*** is EC_RECALL for camp-on return calls.

Coverage

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application monitoring a station will not receive a ***CSTADeliveredEvent*** for a call alerting at a COVER button.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring a station receives a ***CSTADeliveredEvent*** for a call alerting at a COVER button. The ***lastRedirectionDevice*** is the coverage sender, and the ***cause*** is `EC_CALL_FORWARD`, indicating that the call has been forwarded due Coverage.

For MERLIN MAGIX Release 2.0 only, private data may provide ***originalCallInfo*** about the call alerting at the coverage sender.

Beginning with MERLIN MAGIX Release 2.1, changes to the ***calledDevice*** parameter eliminate the need for ***originalCallInfo*** for Coverage scenarios.

A monitoring application receives a ***CSTADeliveredEvent*** for a Calling Group member that receives a Group Coverage call. There is no special information in the ***CSTADeliveredEvent*** to identify the call as a Group Coverage call. In this case, the call is treated as a Calling Group call, so the ***lastRedirectionDevice*** is the Calling Group and the ***cause*** is `EC_REDIRECTED`.

An application monitoring a Calling Group member, that is proving Group Coverage, will receive a ***CSTADeliveredEvent*** for all calls that are receiving coverage treatment. For MERLIN MAGIX Release 2.0, in cases when the call does not alert at the coverage sender (e.g. Do Not Disturb was active), the OCI ***calledDevice*** is the Calling Group and not the Coverage Sender. Beginning with MERLIN MAGIX Release 2.1, the ***calledDevice*** is the Coverage Sender, regardless of the status of Do Not Disturb.

Direct Facility/Pool Termination

Prior to MERLIN MAGIX Release 2.0, the ***CSTADeliveredEvent*** was not generated for DFT and DPT buttons.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring a station will receive a ***CSTADeliveredEvent*** for an incoming call alerting at a DFT/DPT button. The ***trunkUsed*** (private data version 2 or later) is the trunk identifier associated with the call.

When a call appears on a DFT or DPT at an extension, there are cases where the call can appear at the extension more than once. Examples are:

- an extension is a member of a Calling Group and also has a DFT or DPT button for the line ringing into the Calling Group. The application monitoring the extension receives two ***CSTADeliveredEvents***. The Calling Group alerts the call on an SA button at the extension and sends an event with a ***cause*** of `EC_REDIRECTED`. The same call alerts on the DFT/DPT button and sends the second event with a ***cause*** of `EC_NEW_CALL`.

- an extension has an incoming call forwarded to it from another extension (alerts on an SA button) while it also has a DFT button for the facility the call comes in on. An application monitoring the extension will receive two ***CSTADeliveredEvents***. The first event will be for the alert on the DFT button and will have a ***cause*** of EC_NEW_CALL. The second will be for the alert on the SA button and will have a ***cause*** of EC_CALL_FORWARD (MERLIN MAGIX Release 2.0) or EC_CALL_FORWARD_ALWAYS (beginning with MERLIN MAGIX Release 2.1).

Direct Inward Dial (DID) Trunks

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application monitoring a station will receive a ***CSTADeliveredEvent*** for an incoming DID or unassigned DID call alerting at an SA button. There is no special information in the ***CSTADeliveredEvent*** to identify a call as a DID call.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring a station receives a ***CSTADeliveredEvent*** for an incoming DID or unassigned DID call. The ***trunkUsed*** (available beginning with private data version 2 and MERLIN MAGIX Release 2.0) is the trunk identifier associated with the call.

Direct Line Console (DLC)

When an unmonitored DLC transfers an incoming trunk call to a monitored extension, the ***callingDevice*** parameter in the resulting events appear as if the trunk call came directly to that extension. This behavior lets an unmonitored DLC transfer incoming calls to a customer service representative where an application can pop a screen using the original caller's information from the ***CSTADeliveredEvent***.

When a monitored DLC transfers an incoming CO call to a monitored station, the ***CSTADeliveredEvent*** contains the same information as if any other station extension transferred the call. If the DLC operator uses the ***cstaConsultation-Call( )*** service to transfer a call, an application running on behalf of the transfer destination can pop a screen using the OCI.

Forward on Busy

An application monitoring a station where a forward-on-busy call alerts receives a ***CSTADeliveredEvent*** for the forwarded call.

Beginning with MERLIN MAGIX Release 2.0, the ***lastRedirectionDevice*** is the forwarded-from extension, and the ***cause*** is one of two values, EC_CALL_FORWARD or EC_NEW_CALL. If the call appears on a button at the forwarding station (most likely case) ***cause*** is EC_CALL_FORWARD. If the call does not appear on a button ***cause*** is EC_NEW_CALL. Cases where the call does not appear on a button include:

- There is no button available to receive the call
- Do Not Disturb is enabled and the station has some form of Coverage

Beginning with MERLIN MAGIX 2.1, the operation is the same as that for MERLIN MAGIX Release 2.0 except that ***cause*** is assigned EC_CALL_FORWARD_ALWAYS instead of EC_CALL_FORWARD.

Forward/Follow Me

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application monitoring the station receiving a forwarded call does not receive a ***CSTADeliveredEvent***.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring a station receives a ***CSTADeliveredEvent*** for a call alerting at the forward-to extension. For calls Forwarded from an SA button, the ***lastRedirectionDevice*** is the forward-from extension, and the ***cause*** is EC_CALL_FORWARD. If the call appears on an SA button at the forward-from extension, private data may provide Original Call Information for the forwarded call. For calls forwarded from a DFT or DPT button, the ***lastRedirectionDevice*** is ID_NOT_KNOWN, and the ***cause*** is EC_NEW_CALL.

Beginning with MERLIN MAGIX Release 2.1, the operation is the same as that for MERLIN MAGIX Release 2.0 except that changes to the ***calledDevice*** parameter eliminates the need for ***originalCallInfo*** for Forwarding scenarios. Also, a distinction is made in ***cause*** between Coverage and Forwarded calls. For Coverage calls, ***cause*** is EC_CALL_FORWARD (as before) and for Forwarded calls ***cause*** is EC_CALL_FORWARD_ALWAYS.

Group Calling (DGC)

An application monitoring a station where a Calling Group call alerts on an SA button will receive a ***CSTADeliveredEvent***. This includes calls delivered to Calling Group members and the Calling Group delay announcement unit. An application monitoring a station where a Calling Group call alerts on an SA button receives a ***CSTADeliveredEvent*** even in the case where the call first alerts at the announcement unit, then alerts at one Calling Group member, then another, etc. In a MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) environment, there is no special information in the ***CSTA-DeliveredEvent*** to identify a call as a Calling Group call.

Beginning with MERLIN MAGIX Release 2.0, when a DGC call is delivered to a Calling Group member, the ***lastRedirectionDevice*** in the ***CSTADeliveredEvent*** is the Calling Group of which the station is a member (e.g. “Q770”), and the ***cause*** is EC_REDIRECTED.

Networking

An application monitoring a station where a non-local Uniform Dial Plan (UDP) call alerts receives a ***CSTADeliveredEvent***. If the call is answered, the application will also receive a ***CSTAEstablishedEvent***. The ***callingDevice*** parameter in these events will contain the extension number of the calling device on the originating MERLIN LEGEND or MERLIN MAGIX switch, provided that the call has crossed only PRI trunks.

For a non-local UDP call crossing a tie trunk, the ***callingDevice*** parameter of the ***CSTADeliveredEvent*** and the ***CSTAEstablishedEvent*** has a `deviceIDStatus` of `ID_NOT_KNOWN`.

Beginning with MERLIN MAGIX Release 2.0, the ***CSTADeliveredEvent*** is generated for a Network PRI call that is alerting at the far end.

Beginning with private data version 2 and MERLIN MAGIX Release 2.0, the ***trunkUsed*** is the trunk identifier associated with the call.

When an incoming call with collected digits is directed from the MERLIN Messaging system on one switch to another switch in the private network, an application monitoring the station on the terminating switch will not receive collected digits in the private data associated with the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent***.

An application monitoring the transfer destination when the transfer originator is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network will receive a ***CSTADeliveredEvent*** and a ***CSTAEstablishedEvent*** for the consultation call, but these events will not contain any private data for the Original Call Information. The application will not receive a ***CSTATransferredEvent***.

An application monitoring the added party when the conference originator is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network will receive a ***CSTADeliveredEvent*** and a ***CSTAEstablishedEvent*** for the consultation call, but these events will not contain any private data for the Original Call Information. The application will not receive a ***CSTAConferencedEvent***.

Night Service

An application monitoring a station where a Night Service call alerts will receive a ***CSTADeliveredEvent***. There is no special information in the ***CSTADeliveredEvent*** to identify a call as a Night Service call.

Paging

An application will not receive a ***CSTADeliveredEvent*** for incoming Speakerphone Paging calls.

Park

An application monitoring an extension where the user parks a call will not receive a ***CSTADeliveredEvent*** when the user presses the TRANSFER button to park the call.[3] If the parked call is not picked up within the Call Park Return Interval, the application will receive a ***CSTADeliveredEvent*** when the parked call returns and alerts at the parking station.

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, there is no special information in the ***CSTADelivered-Event*** to identify a call as a park return call.

Beginning with MERLIN MAGIX Release 2.0, the ***cause*** is EC_RECALL for park return calls.

PRI

An application monitoring a station where a PRI call is alerting on an SA button will receive a ***CSTADeliveredEvent***.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring an extension where the user makes a PRI call involving all digital lines will receive a ***CSTADeliveredEvent*** when the switch receives a message that the far-end is alerting. Note that the ***calledDevice*** will be the dialed number, which may not accurately identify the alerting device.

Queued Call Console (QCC)

When a QCC transfers an incoming trunk call to a monitored extension, the ***callingDevice*** parameter in the resulting events appear as if the trunk call came directly to that extension. This behavior lets a QCC transfer incoming calls to a customer service representative where an application can pop a screen using the original caller's information from the ***CSTADeliveredEvent***.

Reminder Service

An application monitoring a station where a Reminder Service call alerts will not receive ***CSTADeliveredEvent*** for the reminder call.

Service Observing

An application monitoring the station of a service observer will not receive a ***CSTADeliveredEvent*** for calls delivered to or originating from the station being observed.

However, an application monitoring the station of a service observer will receive a ***CSTAEstablishedEvent*** when a call is answered at the station being observed.

---

[3] A user parks a call by transferring the call to his/her own extension.

Transfer Return

An application monitoring a station to which the Transfer Return feature returns a call on an SA facility will receive a ***CSTADeliveredEvent*** when the transferred call returns and alerts. In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, there is no special information in the ***CSTADeliveredEvent*** to identify a call as a transfer return call.

Beginning with MERLIN MAGIX Release 2.0, the ***cause*** is EC_RECALL for transfer return calls.

Voice Announce

An application will not receive a ***CSTADeliveredEvent*** for incoming Voice Announce calls auto answered on the speakerphone.

Voice Prompting

When a VMI port transfers an incoming trunk call to a monitored extension, the ***callingDevice*** parameters in the resulting events appear as if the trunk call came directly to that extension. This behavior lets a VMI port transfer incoming calls to a customer service representative where an application can pop a screen using the original call's information from the ***CSTADeliveredEvent***.

CSTADivertedEvent

The ***CSTADivertedEvent*** indicates that a call has been redirected and is no longer present at a monitored device.

The MERLIN MAGIX switch provides this event beginning with Release 1.5.

The MERLIN MAGIX switch provides this event only when a call is redirected from a monitored Calling Group queue. This event is generated in the following scenarios:

- a queued call is redirected through the ***cstaDeflectCall( )*** service.
- a queued call is redirected to the overflow queue, the support group or the QCC Listed Directory Number.
- a queued call is delivered to an available Calling Group member.
- a queued call is picked up via Line Pickup.
- a queued call that is also alerting at a Primary or Secondary Coverage button is answered at the Coverage button.
- a queued call that is also alerting at a DFT or DPT button is answered at the DFT or DPT button.
- a queued call that is also alerting at an SA button is answered at a Shared System Access button.

It is possible to receive multiple ***CSTADivertedEvents*** for a single call.

Event Parameters

Table 8-12. CSTADivertedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_DIVERTED`
monitorCrossRefID	event occurred on this monitor
connection	queued connection in the Calling Group queue
divertingDevice	Calling Group where the call was queued before being redirected.
newDestination	device that the call was redirected to
localConnectionInfo	`CS_NONE`, not provided
cause	reason for Diverted event
privateData	NULL, not used for this event

Event Scenario Diagram

Figure 8-4 illustrates one possible ***CSTADivertedEvent*** scenario.

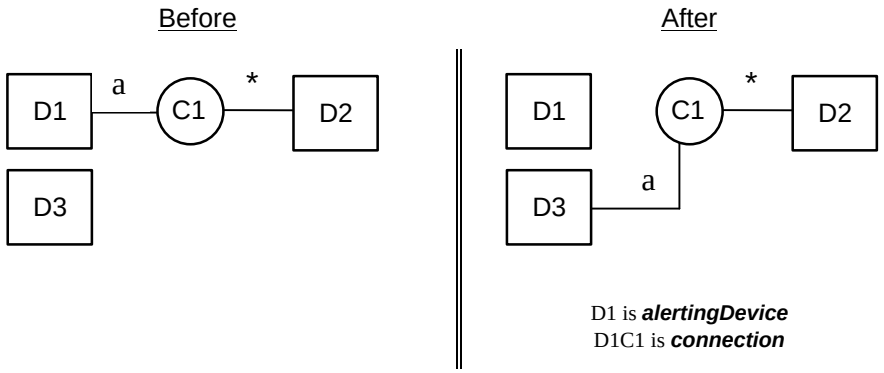


Figure 8-4. CSTADivertedEvent Scenario

Event Causes

Table 8-13. MERLIN MAGIX Release 1.5 CSTADivertedEvent Causes

EC_REDIRECTED	The MERLIN MAGIX Release 1.5 switch provides this cause in all ***CSTADivertedEvents***.

Table 8-14. MERLIN MAGIX Release 2.0 and later CSTADivertedEvent Causes

EC_REDIRECTED	The MERLIN MAGIX Release 2.0 and 2.1 switches provides this cause for all cases except those specified below:
EC_CALL_PICKUP	***connection*** has been picked up via Line Pickup.
EC_OVERFLOW	***connection*** has been redirected from a Calling Group queue to the overflow group, the support group or the QCC LDN.

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTADivertedEvent_t  diverted;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTADivertedEvent_t {
   ConnectionID_t          connection;
   SubjectDeviceID_t       divertingDevice;
   CalledDeviceID_t        newDestination;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTADivertedEvent_t;
```

Important Feature Interactions

Call Pickup

An application will receive a ***CSTADivertedEvent*** when a call in a monitored Calling Group queue is picked up at another extension. Beginning with MERLIN MAGIX Release 2.0, the ***cause*** for the event is EC_CALL_PICKUP.

Coverage

An application will receive a ***CSTADivertedEvent*** when a call in a monitored Calling Group queue is answered at a Primary or Secondary COVER button.

Direct Facility/Pool Termination

An application will receive a ***CSTADivertedEvent*** when a call in a monitored Calling Group queue is answered at a DFT or DPT button.

Group Calling (DGC)

An application will receive a ***CSTADivertedEvent*** when a call in a monitored Calling Group queue is delivered to a Calling Group member.

An application will receive a ***CSTADivertedEvent*** when a call in a monitored Calling Group queue is delivered to the Overflow or support group. Beginning with MERLIN MAGIX Release 2.0, the ***cause*** for the event is EC_OVERFLOW. The ***newDestination*** is the overflow member receiving the call.

Queued Call Console (QCC)

An application will receive a ***CSTADivertedEvent*** when a call in a monitored Calling Group queue is redirected to the QCC LDN. Beginning with MERLIN MAGIX Release 2.0, the ***cause*** for the event is EC_OVERFLOW.

CSTAEstablishedEvent

The ***CSTAEstablishedEvent*** indicates that a call (possibly a consultation call) has been answered at a station.

Unlike the ***CSTADeliveredEvent***, the MERLIN LEGEND and MERLIN MAGIX switches provide a ***CSTAEstablishedEvent*** for a call answered on any button type, including a Shared System Access button.

Event Parameters

Table 8-15. CSTAEstablishedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_ESTABLISHED`
monitorCrossRefID	event occurred on this monitor
establishedconnection	established connection (contains deviceID and call)
answeringDevice	device where connection established
callingDevice	the calling device may contain a number identifying the calling party number. See below for details.
calledDevice	the called device may contain a number identifying the called party number. See below for details
lastRedirectionDevice	For MERLIN MAGIX Release 2.0 and later, the last redirection device for the call, when applicable. See below for details
localConnectionInfo	`CS_NONE`, none provided
cause	reason for Established event (see Tables 8-16 and 8-17)
privateData	may contain call prompting digits, original call information, and/or (private data version 2 and later) the trunk identifier for the call

The ***callingDevice*** parameter contains the ANI/ICLID for a party (when the trunk provides it) or the extension for a local party. CSTA permits values indicating "unknown" for certain ***CSTAEstablishedEvent*** parameters in certain circumstances. When an incoming call arrives on a trunk that does not provide ANI/ICLID, the ***callingDevice*** has `deviceIDStatus` of `ID_NOT_KNOWN`.

IMPORTANT:

For a ***CSTAEstablishedEvent*** event to provide the calling number for an incoming external call, the external call must arrive on either:

- PRI/BRI facilities provisioned to provide ANI.
- trunks that have ICLID-Delay applied by the switch. Typically a call on a facility alerting into a Calling Group, would be delayed being delivered to an extension until the ICLID information arrived.

The switch populates the TSAPI ***calledDevice*** parameter to identify the device being called. Beginning with MERLIN MAGIX Release 2.0, an application monitoring an extension where the user makes a PRI call involving all digital lines will receive a ***CSTAEstablishedEvent*** when the switch receives a message that the far-end has connected. The ***calledDevice*** will be the dialed number, which may or may not match the answering device number. All other outgoing calls do not generate a ***CSTAEstablishedEvent***.

For incoming calls, the ***calledDevice*** parameter is populated with one of the following:

- The *called number* from the ISDN setup message for calls over PRI facilities
- `ID_NOT_KNOWN` (`deviceIDStatus`) for DFT/DPT calls over non-PRI facilities
- DGC Queue for DGC calls over non-PRI facilities and where the facilities do not terminate on DFT/DPT buttons
- `ID_NOT_KNOWN` (`deviceIDStatus`) for DGC calls over non-PRI facilities that terminate on DFT/DPT buttons
- For intercom calls, the ***calledDevice*** parameter is populated with one of the following:
- The *called extension number* for a simple station to station call
- The *forwarding station* for forwarded calls including forwarded on busy
- The *coverage sender* for coverage calls including Calling Group coverage
- The *DGC Queue* for DGC calls
- The *station extension* where a call is being picked up from, using the call pickup feature

As a call redirects (coverage, forwarding, etc.) from its original destination to other endpoints, the ***calledDevice*** for an incoming PRI or BRI call remains static. Beginning in MERLIN MAGIX Release 2.1, this is true of the ***calledDevice*** parameter for all calls.

The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches do not populate the TSAPI ***lastRedirectionDevice*** parameter. This parameter always has `deviceIDStatus` of `ID_NOT_KNOWN`. Beginning with MERLIN MAGIX Release 2.0, the switch populates the TSAPI ***lastRedirectionDevice*** parameter as follows:

- If the call is a DGC call alerting at a station that is a Calling Group member, the ***lastRedirectionDevice*** contains the number of the Calling Group of which the station is a member. The Calling Group for the call and the alerting station may be different.
- Otherwise,
  - — If the call is alerting at a Cover button, the ***lastRedirectionDevice*** contains the extension of the coverage sender
  - — If the call is internal call alerting at a forward-to station, the ***last-RedirectionDevice*** contains the forward-from extension. The MERLIN MAGIX switch does not provide ***lastRedirectionDevice*** for a call forwarded from a DFT or DPT button.

The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches always populate the TSAPI ***cause*** parameter with EC_NEW_CALL.

Beginning with MERLIN MAGIX Release 2.0, the switch will populate the TSAPI cause parameter with the following information (the precedence is the presented order):

- If the call is an observed call at the station of a Service Observer, the ***cause*** is `EC_SILENT_MONITOR`.
- If the call is a DGC call alerting at a station that is a Calling Group member, the ***cause*** is `EC_REDIRECTED`.
- If the call is a transfer, park or camp-on return call, the ***cause*** is `EC_RECALL`.
- If the call is answered at a Cover button, the ***cause*** is `EC_CALL_FORWARD`.
- If the call is answered at a forward-to station (when the call is a non-DFT/DPT call), the ***cause*** is `EC_CALL_FORWARD`. Beginning with MERLIN MAGIX Release 2.1 the ***cause*** is `EC_CALL_FORWARD_ALWAYS`.
- If the call is answered using Call Pickup or Line Pickup, the ***cause*** is `EC_CALL_PICKUP`.
- If the call is an outgoing PRI call, the ***cause*** is `EC_NONE`
- If the call was delivered to the ***answeringDevice*** through an unsupervised transfer, the ***cause*** is `EC_TRANSFER`.
- For all other cases, the ***cause*** is `EC_NEW_CALL`.

The ***CSTAEstablishedEvent*** may contain private data that carries:

- any collected digits that have been associated with the call – If the call is an incoming call that has been routed through a VMI port and prompted digits have been collected
- information about the original call – When an application uses ***cstaConsultationCall( )*** to extend a call, information about the original call is provided in private data. This "original call information" about the transfer source[4] appears in any ***CSTAEstablishedEvents*** for the consultation call. An application at the desktop receiving the consultation call can use the original calling number, original PRI Called Number (DNIS), or original call prompter digits to pop an appropriate screen.

When a user (transfer originator) makes an outbound call and then initiates a consultation transfer to another extension (transfer destination) consults to another user, the application will receive a ***CSTAEstablishedEvent*** containing Original Call Information showing the transfer originator's extension as the original calling party. However, in the case of an Unsupervised transfer, the ***CSTAEstablishedEvent*** will not contain Original Call Information because the transfer originator is no longer a party to the call.

Beginning with private data version 2 and MERLIN MAGIX Release 2.0, private data in the ***CSTAEstablishedEvent*** may also contain the trunk identifier (e.g. "T802") associated with an external call.

In MERLIN LEGEND and MERLIN MAGIX CTI, a connection ID contains a callID that uniquely identifies a call within the switch. Similarly, a deviceID uniquely identifies a device within the switch. Since ***establishedConnection*** is a connectionID (containing both callID and deviceID), the ***answeringDevice*** parameter is redundant. However, both of these parameters are mandatory in CSTA so they must be present.

NOTE:
Not all switches use static, unique device identifiers. Use the ***answeringDevice*** parameter, not the deviceID within the ***establishedConnection*** parameter to obtain the deviceID for the answering device. This will assist in making the application switch-independent.

4 The MERLIN LEGEND and MERLIN MAGIX switches use the following terms in a transfer scenario: When a call is being transferred, the party doing the transferring is the *transfer originator*. The party being transferred is the *transfer source*. The party receiving the transferred call is the *transfer destination*. Thus, the ***activeCall*** parameter in a ***cstaConsultationCall( )*** is a connection at the transfer originator for the call at the transfer source. The ***calledDevice*** parameter in a ***cstaConsultationCall*** specifies the transfer destination.

NOTE:
The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches do not provide the ***CSTAEstablishedEvent*** event for the outbound leg of a call, leaving the switch on a trunk providing answer supervision.[5] Note that the ***cstaGetAPICaps( )*** query does not distinguish between providing this event for a local monitored station calls and trunks. The ***cstaGetAPICaps( )*** response will indicate that the MERLIN LEGEND and MERLIN MAGIX switches provide this event. Programmers must understand the limitation in the ***cstaGetAPICaps( )*** response and not program applications to expect a ***CSTAEstablishedEvent*** event for the far end on an outbound trunk call.

Beginning with MERLIN MAGIX Release 2.0, the switch provides the ***CSTAEstablishedEvent*** event for the outbound leg of a call, leaving the switch on a PRI trunk when the switch receives a message that the far-end has answered. (This requires that the call be routed only on digital facilities.)

NOTE:
The ***CSTAEstablishedEvent*** event is not generated for DGC calls that go over the private network.

Event Scenario Diagram

Figure 8-5 illustrates one possible ***CSTAEstablishedEvent*** scenario.

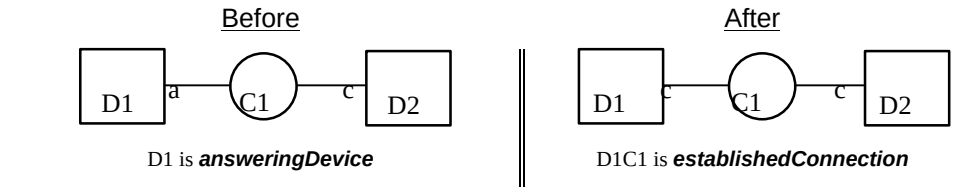


Figure 8-5. CSTAEstablishedEvent Scenario

5 PRI trunking provides the switch with signaling information that the switch can in this circumstance use to generate this event. Generation of an Established event is provided beginning with MERLIN MAGIX Release 2.0.

Event Causes

Table 8-16. MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) CSTAEstablishedEvent Causes

EC_NONE	The MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) switches provide this cause in all ***CSTAEstablishedEvents***

Table 8-17. MERLIN MAGIX Releases 2.0 and 2.1 CSTAEstablishedEvent Causes

EC_NEW_CALL	The MERLIN MAGIX Release 2.0 switch provides this cause in all cases except those specified below:
EC_CALL_FORWARD_ALWAYS	***establishedConnection*** has been forwarded via the Call Forwarding or Coverage features (beginning in MERLIN MAGIX Release 2.1)
EC_CALL_FORWARD	***establishedConnection*** has been forwarded via the Call Forwarding or Coverage features (prior to MERLIN MAGIX Release 2.1)
EC_CALL_PICKUP	***establishedConnection*** has been answered via the Line Pickup feature
EC_RECALL	***establishedConnection*** is a transfer, park or camp-on return call
EC_REDIRECTED	***answeringDevice*** is a Calling Group member and ***establishedConnection*** is a DGC call
EC_NONE	***establishedConnection*** is an outgoing PRI call that is alerting at the far end
EC_SILENT_MONITOR	***establishedConnection*** is an observed call at the station of a Service Observer
EC_TRANSFER	***establishedConnection*** was delivered to ***answeringDevice*** through an unsupervised transfer

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files (acs.h, acsdefs.h, csta.h and cstadefs.h) for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAEstablishedEvent_t  established;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAEstablishedEvent_t {
   ConnectionID_t          establishedConnection;
   SubjectDeviceID_t       answeringDevice;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   RedirectionDeviceID_t   lastRedirectionDevice;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTAEstablishedEvent_t;
```

Private Data Parameters

Table 8-18. CSTAEstablishedEvent Private Data Parameters

userEnteredCode	Specifies the code/digits that may have been entered by the caller through the Collected Digits feature. If the ***userEnteredCode*** code is set to "**ML_UE_NONE**", no ***userEnteredCode*** private data is provided with this event. If the ***userEnteredCode*** code is set to "**ML_CALL_PROMPTER**," ***userEnteredCode*** private data is provided with this event. See the *MERLIN LEGEND Advanced Communications System Feature Reference* or *MERLIN MAGIX Integrated System Feature Reference* (in the CTI Link section) for information on how to set up the switch and application for collecting ***userEnteredCode*** through the Collected Digits feature.

originalCallInfo Specifies the original call information. Note that information is not repeated in the ***originalCallInfo*** if it is already reported in the CSTA service parameters. For example, the ***callingDevice*** and ***calledDevice*** in the ***originalCallInfo*** will not be set if the ***callingDevice*** and the ***calledDevice*** in the CSTA service parameters are the original calling and called devices. The ***callingDevice*** and ***calledDevice*** in the ***originalCallInfo*** will be set only when the original devices are different from the most recent ***callingDevice*** and ***calledDevice***.

NOTE:

For the Established Event received for the ***newCall*** of a Consultation Call, the ***originalCallInfo*** is taken from the ***activeCall*** specified in the Consultation Call request. Thus the application can pass the original call information between two calls. The ***calledDevice*** of the Consultation Call must reside on the same switch and must be monitored via the same Tserver.

The original call information includes:

- ***reason*** — the reason for the ***originalCallInfo***. The following reasons are supported:

  ML_OR_NONE — no ***originalCallInfo*** provided

  ML_OR_CONSULTATION — ***originalCallInfo*** provided

- ***callingDevice*** — the original ***callingDevice*** received by the ***activeCall***
- ***calledDevice*** — the original calledDevice received by the activeCall

NOTE:

In MERLIN MAGIX Release 2.0, ***originalCallInfo*** is also provided for calls that have been redirected due to Forwarding or Coverage. In this case, the ***reason*** for the ***originalCallInfo*** gives no indication that the ***originalCallInfo*** is due to Forwarding or Coverage. However, the ***cause*** in the Delivered event is `EC_CALL_FORWARD`.

In MERLIN MAGIX Release 2.1, changes to the ***calledDevice*** parameter eliminates the need for ***originalCallInfo*** for Forwarding and Cover scenarios. Also, a distinction is made in ***cause*** between Cover and Forwarding calls. For Cover calls, ***cause*** is `EC_CALL_FORWARD` (as before) and for Forwarding calls ***cause*** is `EC_CALL_FORWARD_ALWAYS`.

trunkUsed Available beginning with private data version 2. Contains the trunk identifier (e.g. “T801”) when the call involves a trunk

Private Data Versions 2 and 3 Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the MERLIN MAGIX private data library header files (mlpriv.h and mlpdefs.h) for a complete description.

```
typedef struct {
   MLEventType_t  eventType;     /* ML_ESTABLISHED */
   union {
      /* Only the pertinent union element is shown */
      MLEstablishedEvent_t    establishedEvent;
   } u;
} MLEvent_t;

typedef struct MLEstablishedEvent_t {
   MLUserEnteredCode       userEnteredCode;
   MLOriginalCallInfo_t    originalCallInfo;
   DeviceID_t              trunkUsed;
} MLEstablishedEvent_t;

/*
 * Note: ML_MAX_USER_CODE is defined in mlpriv.h to be the
 * maximum length of the collected digit string.
 */

typedef struct MLUserEnteredCode_t {
   MLUserEnteredCodeType_t type;
   char                    data[ML_MAX_USER_CODE];
} MLUserEnteredCode_t;

typedef enum MLUserEnteredCodeType_t {
   ML_UE_NONE = -1,        /* no collected digits */
   ML_CALL_PROMPTER = 5    /* collected digits */
} MLUserEnteredCodeType_t;

typedef struct MLOriginalCallInfo_t {
   MLReasonForCallInfo_t   reason;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   MLUserEnteredCode_t     userEnteredCode;
} MLOriginalCallInfo_t;

typedef enum MLReasonForCallInfo_t {
   ML_OR_NONE = 0,         /* no OCI present*/
   ML_OR_CONSULTATION = 1  /* OCI present */
} MLReasonForCallInfo_t;
```

Private Data Version 1 Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the MERLIN MAGIX private data library header files (mlpriv.h and mlpdefs.h) for a complete description.

```
typedef struct {
   MLEventType_t     eventType;     /* MLV1_ESTABLISHED */
   union {
      /* Only the pertinent union element is shown */
      MLV1EstablishedEvent_t  v1establishedEvent;
   } u;
} MLEvent_t;

typedef struct MLV1EstablishedEvent_t {
   MLUserEnteredCode_t userEnteredCode;
   MLOriginalCallInfo_t originalCallInfo;
} MLV1EstablishedEvent_t;

/*
 * Note: ML_MAX_USER_CODE is defined in mlpriv.h to be the
 * maximum length of the collected digit string.
 */

typedef struct MLUserEnteredCode_t {
   MLUserEnteredCodeType_t type;
   char                    data[ML_MAX_USER_CODE];
} MLUserEnteredCode_t;

typedef enum MLUserEnteredCodeType_t {
   ML_UE_NONE = -1,        /* no collected digits */
   ML_CALL_PROMPTER = 5    /* collected digits */
} MLUserEnteredCodeType_t;

typedef struct MLOriginalCallInfo_t {
   MLReasonForCallInfo_t   reason;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   MLUserEnteredCode_t     userEnteredCode;
} MLOriginalCallInfo_t;

typedef enum MLReasonForCallInfo_t {
   ML_OR_NONE = 0,         /* no OCI present */
   ML_OR_CONSULTATION = 1  /* OCI present */
} MLReasonForCallInfo_t;
```

Important Feature Interactions

When, in the presence of a feature interaction, an application receives a ***CSTADeliveredEvent***, it will also receive the corresponding ***CSTAEstablishedEvent*** if the alerting connection is answered. With the exception of Auto Answer, in those feature interactions where an application does not receive a ***CSTADeliveredEvent***, it may receive a ***CSTAEstablishedEvent***. Refer to "Important Feature Interactions" pertaining to the ***CSTADeliveredEvent*** for related information.

There are cases when a ***CSTADeliveredEvent*** will contain private data, but the ***CSTAEstablishedEvent*** will not. This will happen when the calling device is no longer on the call. For, if a monitored extension makes an external call and then does an unsupervised transfer (using the services) to another extension, the ***CSTADeliveredEvent*** will contain private data with the original calling and called device, but the ***CSTAEstablishedEvent*** will not. This prevents an application doing a screen pop on the Transfer Originator, who is no longer on the call.

Auto Answer

An application will not receive a ***CSTAEstablishedEvent*** when a call is auto answered (via the headset feature) at a monitored station.

Barge-In

An application will not receive a ***CSTAEstablishedEvent*** when a monitored extension Barges-In on a call.

Call Pickup

In a MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application will not receive a ***CSTAEstablished-Event*** when a call is picked up at a monitored station.

Beginning with MERLIN MAGIX Release 2.0, an application receives a ***CSTAEstablishedEvent*** when a call is picked up at a monitored station, and the ***cause*** is `EC_CALL_PICKUP`.

Call Screening

A device monitor for the extension of a Call Screener will receive a ***CSTAEstablishedEvent*** when a screened call is presented at the extension. The ***cause*** in the ***CSTAEstablishedEvent*** is `EC_SILENT_MONITOR`. Device monitors for other extensions on the call will not receive this event.

Camp-On

When a camp-on return call is answered at a monitored station, monitoring applications will receive a ***CSTAEstablishedEvent***.

In a MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) environment, there is no special information in the ***CSTAEstablishedEvent*** to identify a call as a Camp-On return call.

Beginning with MERLIN MAGIX Release 2.0, the ***cause*** is EC_RECALL for camp-on return calls.

Coverage

In a MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application monitoring a station will not receive a ***CSTAEstablishedEvent*** for a call answered at a COVER button.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring a station receives a ***CSTAEstablishedEvent*** for a call answered at a COVER button. The ***lastRedirectionDevice*** is the coverage sender, and the ***cause*** is EC_CALL_FORWARD, indicating that the call has been forwarded due to either Coverage or Forwarding. Private data may provide Original Call Information about the call delivered to the coverage sender. Beginning with MERLIN MAGIX Release 2.1, changes to the ***calledDevice*** parameter eliminates the need for ***originalCallInfo*** for Forwarding and Cover scenarios and it is therefore no longer provided.

A monitoring application receives a ***CSTAEstablishedEvent*** for a Calling Group member which gets Group Coverage call that is answered on an SA button. There is no special information in the ***CSTAEstablishedEvent*** to identify the call as a Group Coverage call. In this case, the call is treated as a Calling Group Call, so the ***lastRedirectionDevice*** is the Calling Group and the ***cause*** is EC_REDIRECTED.

Direct Facility/Pool Termination

An application monitoring a station receives a ***CSTAEstablishedEvent*** for an incoming call answered at a DFT or DPT button. The ***trunkUsed*** (private data version 2 or later) is the trunk identifier associated with the call.

When a DFT appears at an extension, there are cases where the call can appear at the extension more than once.

An example of this is when the extension is a member of a Calling Group and also has a DFT appearance of the line that is ringing into the Calling Group. In this case, the application monitoring the extension receives two ***CSTADeliveredEvents***. One has a ***cause*** of EC_REDIRECTED and the other has a ***cause*** of EC_NEW_CALL. If the call is answered at the DFT, the ***CSTAEstablishedEvent*** has a ***cause*** of EC_NEW_CALL. If the call is answered at the SA button, the ***CSTAEstablishedEvent*** has a ***cause*** of EC_REDIRECTED.

Another example is when a forwarded-to extension is alerting (on an SA button) with a call forwarded from another extension and the extension has the same call also alerting on a DFT button. In this case, the application monitoring the extension receives two ***CSTADeliveredEvents***. One has a ***cause*** of EC_CALL_FORWARD_ALWAYS and the other has a ***cause*** of EC_NEW_CALL. If the call is answered at the DFT, the ***CSTAEstablishedEvent*** has a ***cause*** of EC_NEW_CALL. If the call is answered at the SA button, the ***CSTAEstablishedEvent*** has a ***cause*** of EC_CALL_FORWARD_ALWAYS.

Direct Line Console (DLC)

When an unmonitored DLC transfers an incoming trunk call to a monitored extension, the calling party parameters in the resulting ***CSTAEstablishedEvent*** appears as if the trunk call came directly to the monitored station.

When a monitored DLC transfers an incoming CO call to a monitored station, the ***CSTAEstablishedEvent*** contains the same information as if any other station extension transferred the call.

Direct Inward Dial (DID) Trunks

An application monitoring a station receives a ***CSTAEstablishedEvent*** when an incoming DID or unassigned DID call is answered. Beginning with MERLIN MAGIX Release 2.0, the ***trunkUsed*** (private data version 2 or later) is the trunk identifier associated with the call.

Forward on Busy

An application monitoring a station where a forward-on-busy call is answered receives a ***CSTAEstablishedEvent*** for the forwarded call.

Beginning with MERLIN MAGIX Release 2.0, the ***lastRedirectionDevice*** is the forwarded-from extension, and the ***cause*** is one of two values, EC_CALL_FORWARD or EC_NEW_CALL. If the call appears on a button at the forwarding station (most likely case) ***cause*** is EC_CALL_FORWARD. If the call does not appear on a button ***cause*** is EC_NEW_CALL.. Cases where the call does not appear on a button includes:

- There is no button available to receive the call
- Do Not Disturb is enabled and the station has some form of Coverage

Beginning with MERLIN MAGIX Release 2.1, the operation is the same as that for MERLIN MAGIX Release 2.0 except that ***cause*** is assigned EC_CALL_FORWARD_ALWAYS instead of EC_CALL_FORWARD.

Forward/Follow Me

In a MERLIN LEGEND (Release 5.0 or later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, an application monitoring the station receiving a forwarded call does not receive a ***CSTADeliveredEvent***.

Beginning with MERLIN MAGIX Release 2.0, for calls Forwarded from an SA button, the ***lastRedirectionDevice*** is the forward-from extension, and the ***cause*** is `EC_CALL_FORWARD`. If the call appears on an SA button at the forward-from extension, private data may provide Original Call Information for the forwarded call. For calls forwarded from a DFT or DPT button, the ***lastRedirectionDevice*** is `ID_NOT_KNOWN`, and the ***cause*** is `EC_NEW_CALL`.

Beginning with MERLIN MAGIX Release 2.1, the operation is the same as that for MERLIN MAGIX Release 2.0 except that changes to the ***calledDevice*** parameter eliminates the need for ***originalCallInfo*** for Forwarding scenarios. Also, a distinction is made in ***cause*** between Cover and Forwarding calls. For Cover calls, ***cause*** is `EC_CALL_FORWARD` (as before) and for Forwarding calls ***cause*** is `EC_CALL_FORWARD_ALWAYS`.

Group Calling (DGC)

An application monitoring a station where a DGC call is answered receives a ***CSTAEstablishedEvent***. Beginning with MERLIN MAGIX Release 2.0, when a DGC call is answered by a Calling Group member, the ***lastRedirectionDevice*** is the Calling Group (i.e. “Q770”), and the ***cause*** is `EC_REDIRECTED`

Networking

An application monitoring a station where a non-local Uniform Dial Plan (UDP) call is answered receives a ***CSTAEstablishedEvent***. The ***callingDevice*** parameter contains the extension number of the calling device on the originating MERLIN LEGEND or MERLIN MAGIX switch.

For a non-local UDP call crossing a tie trunk, the ***callingDevice*** parameter of the ***CSTAEstablishedEvent*** has a `deviceIDStatus` of `ID_NOT_KNOWN`.

Beginning with MERLIN MAGIX Release 2.0 and private data version 2, the ***trunkUsed*** is the trunk identifier associated with the call.

Beginning with MERLIN MAGIX Release 2.0, the ***CSTAEstablishedEvent*** is generated for a Network PRI call that is answered at the far end.

When an incoming call for which digits were collected is directed from the MERLIN Messaging system on one switch to an extension on a satellite switch, an application monitoring the extension on the terminating switch will not receive collected digits in the private data associated with the ***CSTAEstablishedEvent***.

An application monitoring the transfer destination when the transfer originator is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network receives a ***CSTADeliveredEvent*** and a ***CSTAEstablishedEvent*** for the consultation call, but these events will not contain any private data for the Original Call Information. The application will not receive a ***CSTATransferredEvent***.

An application monitoring the added party when the conference originator is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network receives a ***CSTADeliveredEvent*** and a ***CSTAEstablishedEvent*** for the consultation call, but these events will not contain any private data for the Original Call Information. The application will not receive a ***CSTAConferencedEvent***.

Paging

An application will not receive a ***CSTAEstablishedEvent*** for Speakerphone Paging calls.

Queued Call Console (QCC)

When a QCC transfers an incoming trunk call to a monitored extension, the calling party parameters in the resulting events appear as if the trunk call came directly to that extension. This behavior lets a QCC transfer incoming calls to a customer service representative where an application can pop a screen using the original caller's information from the ***CSTADeliveredEvent***.

Park

When a park return call is answered at a monitored station, an application will receive a ***CSTAEstablishedEvent.***

In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, there is no special information in the ***CSTAEstablishedEvent*** to identify a call as a park return call.

Beginning with MERLIN MAGIX Release 2.0, the ***cause*** is EC_RECALL for park return calls.

PRI

An application monitoring an extension where a PRI call is answered receives a ***CSTAEstablishedEvent***.

Beginning with MERLIN MAGIX Release 2.0, an application monitoring an extension where the user makes a PRI call involving all digital lines receives a ***CSTAEstablishedEvent*** when the switch receives a message that the far-end has answered. Note that the ***calledDevice*** contains the dialed number, which may not match the answering device.

Reminder Service

An application monitoring a station where a Reminder Service call is answered will not receive ***CSTAEstablishedEvent***.

Service Observing

An application monitoring a station for a service observer receives a ***CSTA-EstablishedEvent*** when the observed call appears at the station of the service observer.

Beginning with MERLIN MAGIX Release 2.0, only an application monitoring the service observer receives a ***CSTAEstablishedEvent*** when the observer monitors calls at a targeted station. Service Observing can be activated for a targeted station at any time. If the targeted station is already active on a call when Service Observing is initiated no ***CSTAEstablishedEvent*** will be delivered to an application monitoring the Service Observer. In order for the ***CSTAEstablishedEvent*** to be delivered Service Observing must be active at the time the targeted station becomes active on the call The ***cause*** in the ***CSTA-EstablishedEvent*** is `EC_SILENT_MONITOR`.

Transfer Return

When a transfer return call is answered at a monitored station, an application receives a ***CSTAEstablishedEvent***. In a MERLIN LEGEND (Release 5.0 and later) or MERLIN MAGIX (Releases 1.0 and 1.5) environment, t here is no special information in the ***CSTAEstablishedEvent*** to identify the call as a transfer return call.

Beginning with MERLIN MAGIX Release 2.0, the ***cause*** is `EC_RECALL` for transfer return calls.

Voice Announce

An application receives a ***CSTAEstablishedEvent*** for incoming Voice Announce calls.

Voice Prompting

When a VMI port transfers an incoming trunk call to a monitored extension, the calling party parameters in the resulting events appear as if the trunk call came directly to that extension. This behavior lets a VMI port transfer incoming calls to a customer service representative where an application can pop a screen using the original call's information from the ***CSTADeliveredEvent***.

CSTAHeldEvent

The ***CSTAHeldEvent*** indicates that station ***holdingDevice*** placed the ***heldConnection*** on hold, on hold-for-conference, or on hold-for-transfer. Prior to MERLIN MAGIX Release 2.1, ***CSTAHeldEvent*** event did not distinguish between the various MERLIN LEGEND and MERLIN MAGIX switch hold types (hold, hold-for-transfer, hold-for-conference.). Beginning with MERLIN MAGIX Release 2.1, the ***CSTAHeldEvent*** distinguishes hold-for-transfer from hold and hold-for-conference.

NOTE:
The MERLIN LEGEND and MERLIN MAGIX switches do not supply this event when a call is placed on associative hold.

Event Parameters

Table 8-19. CSTAHeldEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_HELD`
monitorCrossRefID	event occurred on this monitor
heldConnection	held connection (contains deviceID and callID)
holdingDevice	device where connection held
localConnectionInfo	`CS_NONE`, not provided
cause	reason for Held event
privateData	`NULL`, none present in Held event

Event Scenario Diagram

Figure 8-6 illustrates one possible ***CSTAHeldEvent*** scenario.

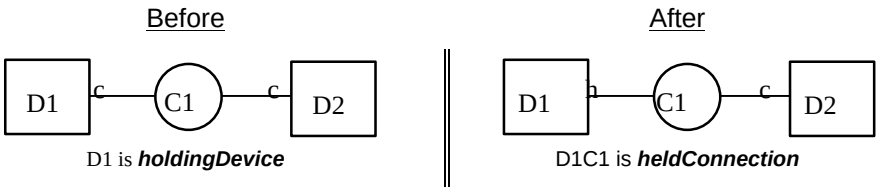


Figure 8-6. CSTAHeldEvent Scenario

Event Causes

Table 8-20. CSTAHeldEvent Causes Prior to MERLIN MAGIX 2.1

EC_NONE	Hold, hold for conference, hold for transfer

Table 8-21. CSTAHeldEvent Causes for MERLIN MAGIX Release 2.1 and Later

EC_NONE	Hold, hold for conference
EC_TRANSFER	Hold for transfer

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAHeldEvent_t   held;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAHeldEvent_t {
   ConnectionID_t          heldConnection;
   SubjectDeviceID_t       holdingDevice;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTAHeldEvent_t;
```

Important Feature Interactions

Conference

An application cannot use the ***cstaConferenceCall( )*** service to conference a call that has been placed on hold (the call must be held-for-conference or held-for-transfer.) Receipt of a ***CSTAHeldEvent*** is not sufficient for an application to infer that the held call can be conferenced. The user may have pressed the HOLD button (resulting in a hold) rather than the CONFERENCE button (resulting in hold-for-conference) or the TRANSFER button (resulting in hold-for-transfer).

Park

When a call is Parked an application monitoring the device where the call is Parked will receive a ***CSTAHeldEvent*** with a cause of EC_TRANSFER but the call can not be transferred.

Service Observing

An application monitoring a station for a service observer receives a ***CSTA-HeldEvent*** when an observed call is placed on hold.

Transfer

An application cannot use the ***cstaTransferCall( )*** service to transfer a call that has been placed on hold (must be held-for-transfer). Prior to MERLIN MAGIX Release 2.1, receipt of a ***CSTAHeldEvent*** is not sufficient to infer that the call can be transferred. The user may have pressed the HOLD button (resulting in a hold) rather than the TRANSFER button (resulting in hold-for-transfer). Beginning with MERLIN MAGIX Release 2.1, receipt of a ***CSTAHeldEvent*** is usually sufficient to determine if a call can transferred because ***cause*** is EC_TRANSFER for calls on hold-for-transfer.

CSTANetworkReachedEvent

The ***CSTANetworkReachedEvent*** indicates that a call is an outgoing trunk call and is seizing a trunk. Since trunk signaling does not provide as much information about Call Events as the MERLIN LEGEND or MERLIN MAGIX switch obtains for local devices, an application may not receive any additional events for the trunk far end.

connection is the trunk's connection to the call.

trunkUsed identifies the trunk. It contains the MERLIN LEGEND or MERLIN MAGIX switch Facility Identifier for that trunk.

NOTE:
Application design should always allow for a ***calledDevice*** parameter with a `deviceIDStatus` of `ID_NOT_KNOWN`.

Event Parameters

Table 8-22. CSTANetworkReachedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_NETWORK_REACHED`
monitorCrossRefID	Network Reached event occurred on this monitor
connection	connectionID for the trunk connection to the call (contains a deviceID that identifies a MERLIN LEGEND or MERLIN MAGIX CTI facility and a callID)
trunkUsed	identifies a MERLIN LEGEND or MERLIN MAGIX switch facility
calledDevice	destination (usually the dialed digits) If unknown, the `deviceIDStatus` component has a value of `ID_NOT_KNOWN`.
localConnectionInfo	`CS_NONE`, not provided
cause	reason for Network Reached event.
privateData	`NULL`, none present in Network Reached event

Event Scenario Diagram

Figure 8-7 illustrates one possible ***CSTANetworkReachedEvent*** scenario.

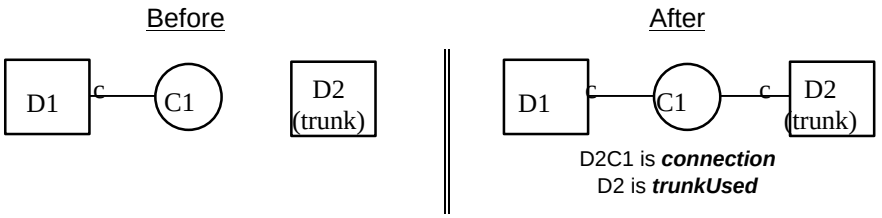


Figure 8-7. CSTANetworkReachedEvent Scenario

Event Causes

Table 8-23. CSTANetworkReachedEvent Causes

EC_NONE	The MERLIN LEGEND and MERLIN MAGIX switches provide this cause in all ***CSTANetworkReachedEvents***.

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTANetworkReachedEvent_t  networkReached;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTANetworkReachedEvent_t {
   ConnectionID_t          connection;
   SubjectDeviceID_t       trunkUsed;
   CalledDeviceID_t        calledDevice;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTANetworkReachedEvent_t;
```

Important Feature Interactions

ARS

The ***calledDevice*** parameter in the ***CSTANetworkReachedEvent*** includes absorbed digits. The added digits are not included in the ***calledDevice.***

Auto-Dial

An application monitoring a station from which a user uses the Auto-Dial feature to originate an outgoing trunk call receives a ***CSTANetworkReachedEvent***.

The ***calledDevice*** parameter does not include any Pool access code digits.

End-Of-Dial Character

If the call originator dialed a number which terminated with the End-Of-Dial character (#) or if an application used a CSTA service to originate the call and terminated the destination number with the End-Of-Dial character, then the MERLIN LEGEND or MERLIN MAGIX switch generates ***CSTANetworkReached-Event*** on trunk seizure. Applications that desire this event immediately on trunk seizure should include the End-Of-Dialing character in the destination number in ***cstaMakeCall( )*** service requests. For analog trunks, the seizure happens after the End-of-Dial character is dialed. On PRI trunks, trunk seizure is immediate.

The End-Of-Dial character, when used, does not appear in the ***calledDevice*** parameter in the resulting ***CSTANetworkReachedEvent***.

Marked System Speed Dial

An application monitoring a station from which a user uses the Marked System Speed Dial feature to originate an outgoing trunk call receives a ***CSTANetworkReachedEvent***.

The ***calledDevice*** parameter contains the system speed dial code and *not* the number dialed for the call.

Networking

An application monitoring a station from which a non-local Uniform Dial Plan (UDP) call is placed receives a ***CSTANetworkReachedEvent***. The ***calledDevice*** parameter in the ***CSTANetworkReachedEvent*** contains the extension number of the called device on the terminating MERLIN LEGEND or MERLIN MAGIX switch.

.An application monitoring a Calling Group containing a non-local member does not receive the ***CSTANetworkReachedEvent*** when the call leaves the switch***.***

Non-PRI Trunks

If the End-of-Dialing character was not present in the dialed number, then the MERLIN LEGEND or MERLIN MAGIX switch generates a ***CSTANetworkReachedEvent*** when the end-of-dialing time-out occurs.

Pool Access Code

The ***calledDevice*** parameter in a ***CSTANetworkReachedEvent*** includes absorbed digits.

PRI Trunks

If the End-of-Dialing character was not present in the dialed number, then the MERLIN LEGEND or MERLIN MAGIX switch generates a ***CSTANetwork-ReachedEvent*** when the Central Office signals end-of-dialing.

Redial

An application monitoring a station from which a user uses the Redial feature to originate a call involving a trunk receives a ***CSTANetworkReachedEvent***.

The ***calledDevice*** parameter does not include any Pool code digits.

Save Number Dial

An application monitoring a station from which a user uses the Save Number Dial feature to originate a call which leaves the switch on a trunk will receive a ***CSTANetworkReachedEvent***.

The ***calledDevice*** parameter does not include any Pool code digits.

Service Observing

An application monitoring the station of a service observer will not receive a ***CSTANetworkReachedEvent*** for calls originating from the station being observed.

CSTAQueuedEvent

The ***CSTAQueuedEvent*** indicates that a call has entered a Calling Group queue.

The MERLIN MAGIX switch provides this event beginning with Release 1.5. Private data support is added in MERLIN MAGIX Release 2.0.

The MERLIN MAGIX switch provides this event when a Calling Group call has been queued to a Calling Group. The Calling Group may be any type (e.g., Auto Login, Auto Logout, etc.).

This event is generated in the following scenarios:

- a call enters a Calling Group queue because no Calling Group members are available to receive the call
- a DGC call alerting at the station of a Calling Group member is returned to the queue because it was not answered
- a DGC call is redirected to the queue via the ***cstaDeflectCall( )*** service
- a Group coverage call has received Calling Group Coverage treatment

When a call has entered a Calling Group queue for any of these reasons, an application monitoring either the calling party or the Calling Group receives a ***CSTAQueuedEvent***.

A ***CSTAQueuedEvent*** is not generated when a call goes to overflow or to a support group.

It is possible to receive multiple ***cstaQueuedEvents*** for a single call.

Event Parameters

Table 8--24. CSTAQueuedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_QUEUED`
monitorCrossRefID	event occurred on this monitor
queuedConnection	Queued connection (contains deviceID and callID)
queue	device where connection is queued
callingDevice	the calling device may contain a number identifying the calling party number. See below for details
calledDevice	the called device may contain a number identifying the called party number. See below for details
lastRedirectionDevice	MERLIN MAGIX Release 1.5: not provided. The `deviceIDStatus` component always has a value of `ID_NOT_KNOWN`. MERLIN MAGIX Release 2.0 and later: the last redirection device for the call, when applicable. When not applicable, the `deviceIDStatus` component always has a value of `ID_NOT_KNOWN`
numberQueued	number of calls in the Calling Group queue
localConnectionInfo	`CS_NONE`, not provided
cause	reason for Queued event (See Table 8-24)
privateData	(Private Data version 2 and later) may contain call prompting digits, original call information, and/or the trunk identifier for the call

The ***callingDevice*** parameter contains the ANI/ICLID for an external party (when the trunk provides it) or the extension for a local party. CSTA permits values indicating “unknown” for certain ***CSTAQueuedEvent*** parameters in certain circumstances. When an incoming call arrives on a trunk that does not provide ANI/ICLID, the ***callingDevice*** has `deviceIDStatus` of `ID_NOT_KNOWN`.

IMPORTANT:

For a ***CSTAQueuedEvent*** event to provide the calling number for an incoming external call, the external call must arrive on either:

- PRI/BRI facilities provisioned to provide ANI.
- trunks that have ICLID-Delay applied by the switch.

When an incoming alerting arrives on a PRI/BRI trunk provisioned to provide DNIS, the ***calledDevice*** parameter contains the PRI Called Number. For other cases and prior to MERLIN MAGIX Release 2.1, the ***calledDevice*** parameter matches the **alertingDevice** parameter. Note that the parameter does not necessarily indicate the device called by the **callingDevice**.

Beginning with MERLIN MAGIX Release 2.1, the switch populates the TSAPI ***calledDevice*** parameter to identify the device being called. For DGC related calls the ***calledDevice*** parameter is populated as described below:

For incoming external calls, the ***calledDevice*** parameter is populated with one of the following:

- The *called number* from the ISDN setup message for calls over PRI facilities
- Calling Group Queue for Calling Group calls over non-PRI facilities and where the facilities do not terminate on DFT/DPT buttons
- `ID_NOT_KNOWN` (`deviceIDStatus`) for Calling Group calls over non-PRI facilities that terminate on DFT/DPT buttons

For intercom calls, the ***calledDevice*** parameter is populated with one of the following:

- The *coverage sender* for coverage calls including Calling Group coverage
- The Calling Group Queue for DGC calls

Event Scenario Diagram

Figure 8-8 illustrates one possible ***CSTAQueuedEvent*** scenario.

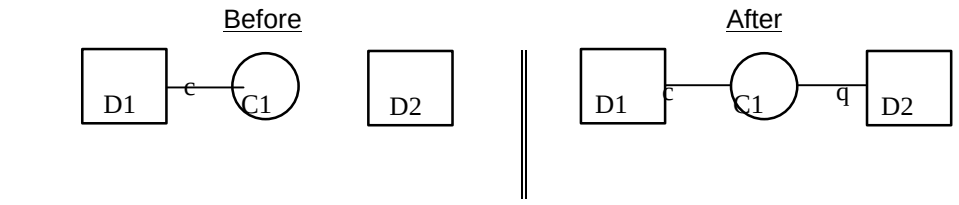


Figure 8-8. CSTAQueuedEvent Scenario

Event Causes

Table 8--25. CSTAQueuedEvent Causes

EC_CALL_FORWARD	Beginning in MERLIN MAGIX Release 2.1, this ***cause*** indicates call has been queued as a result of Calling Group coverage
EC_NONE	Prior to MERLIN MAGIX Release 2.1, this ***cause*** is provided in all ***CSTAQueuedEvents***. Beginning with MERLIN MAGIX Release 2.1, this ***cause*** indicates call has been queued for a reason other that Calling Group coverage

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAQueuedEvent_t    queued;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAQueuedEvent_t {
   ConnectionID_t          queuedConnection;
   SubjectDeviceID_t       queue;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   short                   numberQueued;
   RedirectionDeviceID_t   lastRedirectionDevice;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTAQueuedEvent_t;
```

Private Data Parameters

Table 8-26. CSTAQueuedEvent Private Data Parameters

userEnteredCode	Specifies the code/digits that may have been entered by the Collected Digits feature. If the ***userEnteredCode*** type is set to "**ML_UE_NONE**", no Collected Digits are provided with this event. If the ***userEnteredCode*** code is set to "**ML_CALL_PROMPTER**," Collected Digits are provided with this event. See the *MERLIN MAGIX Integrated System Feature Reference* (CTI Link Section) for information on how to set up the switch and application for collecting ***userEnteredCode*** through the Collected Digits feature.
originalCallInfo	Specifies the original call information. Note that information is not repeated in the ***originalCallInfo***, if it is already reported in the CSTA event parameters. For example, the ***callingDevice*** and ***calledDevice*** in the ***originalCallInfo*** will not be set if the ***callingDevice*** and the ***calledDevice*** in the CSTA service parameters are the original calling and called devices. The ***callingDevice*** and ***calledDevice*** in the ***originalCallInfo*** will be set only when the original devices are different from the most recent ***callingDevice*** and ***calledDevice***. **NOTE:** For the Queued Event corresponding to the ***newCall*** of a Consultation Call to a Calling Group queue, the ***originalCallInfo*** is taken from the ***activeCall*** specified in the Consultation Call request. Thus the application can pass the original call information between two calls. The ***calledDevice*** of the Consultation Call must reside on the same switch and must be monitored via the same Tserver. The original call information includes: ■ ***reason*** — the reason for the ***originalCallInfo***. The following reasons are supported. **ML_OR_NONE** — no ***originalCallInfo*** provided **ML_OR_CONSULTATION** — ***originalCallInfo*** provided ■ ***callingDevice*** — the original ***callingDevice*** received by the ***activeCall***. ■ ***calledDevice*** — the original ***calledDevice*** received by the ***activeCall***.

trunkUsed	Contains the trunk identifier (e.g. “T801”) when the call involves a trunk

Private Data Version 2 and 3 Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the MERLIN MAGIX private data library header files (mlpriv.h and mlpdefs.h) for a complete description.

```
typedef struct {
   MLEventType_t  eventType;     /* ML_QUEUED */
   union {
      /* Only the pertinent union element is shown */
      MLQueuedEvent_t   queuedEvent;
   } u;
} MLEvent_t;

typedef struct MLQueuedEvent_t {
   MLUserEnteredCode_t     userEnteredCode;
   MLOriginalCallInfo_t    originalCallInfo;
   DeviceID_t              trunkUsed;
} MLQueuedEvent_t;

/*
 * Note: ML_MAX_USER_CODE is defined in mlpriv.h to be the
 * maximum length of the collected digit string.
 */

typedef struct MLUserEnteredCode_t {
   MLUserEnteredCodeType_t type;
   char                    data[ML_MAX_USER_CODE];
} MLUserEnteredCode_t;

typedef enum MLUserEnteredCodeType_t {
   ML_UE_NONE = -1,        /* no collected digits */
   ML_CALL_PROMPTER = 5    /* collected digits */
} MLUserEnteredCodeType_t;

typedef struct MLOriginalCallInfo_t {
   MLReasonForCallInfo_t   reason;
   CallingDeviceID_t       callingDevice;
   CalledDeviceID_t        calledDevice;
   MLUserEnteredCode_t     userEnteredCode;
} MLOriginalCallInfo_t;

typedef enum MLReasonForCallInfo_t {
   ML_OR_NONE = 0,         /* no OCI present */
   ML_OR_CONSULTATION = 1  /* OCI present */
} MLReasonForCallInfo_t;
```

Important Feature Interactions

Coverage

An application monitoring a Calling Group Queue that is providing Group Coverage receives a ***CSTAQueuedEvent*** for all calls receiving coverage treatment. Prior to MERLIN MAGIX Release 2.1, if the call had not alerted at the sender (i.e., because Do Not Disturb was active), the ***calledDevice*** would be the Calling Group Queue.

Beginning with MERLIN MAGIX Release 2.1, the ***calledDevice*** is populated as described earlier in this section***.*** The ***lastRedirectionDevice*** contains the Coverage Sender.

Group Calling (DGC)

An application monitoring a DGC queue receives a ***CSTAQueuedEvent*** when the call enters the queue because there are no available members, because the call was alerting at a Calling Group member and returned to the queue, because the ***cstaDeflectCall( )*** service was used to redirect the call to the queue, or because the call is receiving DGC Group Coverage treatment.

CSTARetrievedEvent

The ***CSTARetrievedEvent*** indicates that station ***retrievingDevice*** retrieved the ***retrievedConnection*** from hold, hold-for-conference, or hold-for-transfer. The event does not provide any information to indicate from what type of hold the connection was retrieved.

NOTE:
The MERLIN LEGEND and MERLIN MAGIX switches do not supply this event when a call is retrieved from associative hold.

Event Parameters

Table 8-27. CSTARetrievedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_RETRIEVED`
monitorCrossRefID	Retrieved event occurred on this monitor
retrievedConnection	retrieved connection (contains a deviceID and a callID)
retrievingDevice	device where connection was retrieved
localConnectionInfo	`CS_NONE`, not provided
cause	reason for Retrieved event
privateData	`NULL`, none present in Retrieved event

Event Scenario Diagram

Figure 8-9 illustrates one possible ***CSTARetrievedEvent*** scenario.

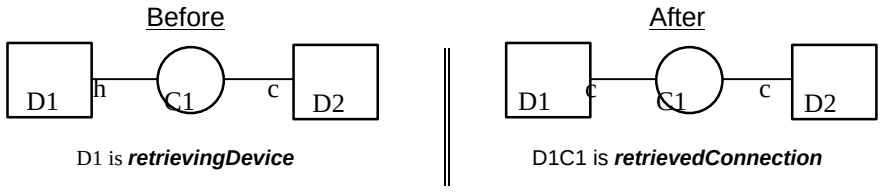


Figure 8-9. CSTARetrievedEvent Scenario

Event Causes

Table 8-28. CSTARetrievedEvent Causes

EC_NONE	The MERLIN LEGEND and MERLIN MAGIX switches provide this cause in all ***CSTARetrievedEvents***.

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTARetrievedEvent_t retrieved;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTARetrievedEvent_t {
   ConnectionID_t          retrievedConnection;
   SubjectDeviceID_t       retrievingDevice;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTARetrievedEvent_t;
```

Important Feature Interactions

Consultation

An application monitoring a station that has performed a consultation call receives a ***CSTARetrievedEvent*** when the held call is re-accessed.

Service Observing

An application monitoring a station for a service observer receives a ***CSTA-EstablishedEvent*** when an observed call is retrieved from the held state. The cause will be `EC_SILENT_MONITOR`.

Transfer

An application monitoring a station that has performed an unsupervised transfer receives a ***CSTARetrievedEvent*** when the transfer originator re-accesses the call. This happens even if the call is alerting at the transfer destination.

CSTAServiceInitiatedEvent

The ***CSTAServiceInitiatedEvent*** indicates that a device initiated a connection. The MERLIN LEGEND and MERLIN MAGIX switches provide this event when the initiating device receives dial tone.

NOTE:
Unlike other events, ***CSTAServiceInitiatedEvent*** does not have a parameter for the device where the event occurred. This is not a mistake. There is no such parameter for the device in TSAPI or CSTA. Since the MERLIN LEGEND and MERLIN MAGIX switches use only static device identifiers, an application may determine the device from the device component of the ***initiatedConnection*** parameter. Alternatively an application may use the ***monitorCrossRefID*** in the event to determine the device for which the event occurred.

Event Parameters

Table 8-29. CSTAServiceInitiatedEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_SERVICE_INITIATED`
monitorCrossRefID	Service Initiated event occurred on this monitor
initiatedConnection	initiated connection (contains a deviceID and a callID)
localConnectionInfo	`CS_NONE`, none provided
cause	reason for Service Initiated event
privateData	`NULL`, none present in Service Initiated event

Event Scenario Diagram

Figure 8-10 illustrates one possible ***CSTAServiceInitiatedEvent*** scenario.

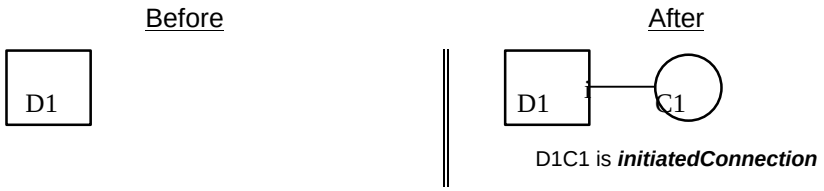


Figure 8-10. CSTAServiceInitiatedEvent Scenario

Event Causes

Table 8-30. CSTAServiceInitiatedEvent Causes

EC_NONE	The MERLIN LEGEND and MERLIN MAGIX switches provide this cause in all ***CSTAServiceInitiatedEvents***.

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAServiceInitiatedEvent_t   serviceInitiated;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAServiceInitiatedEvent_t {
   ConnectionID_t          initiatedConnection;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTAServiceInitiatedEvent_t;
```

Important Feature Interactions

Service Observing

An application monitoring a station that is a service observer does not receive a ***CSTAServiceInitiatedEvent*** for calls originating from the station being observed.

An application monitoring a station that is a service observer receives a ***CSTA-ServiceInitiatedEvent*** when the station goes off-hook on a System Access button to start observing.

CSTATransferredEvent

The ***CSTATransferredEvent*** indicates that station ***transferringDevice*** has transferred a call. Specifically, the ***transferringDevice*** station had the connection ***primaryOldCall*** on hold-for-transfer and the connection ***secondaryOldCall*** active and then transferred the ***primaryOldCall*** to ***secondaryOldCall***. In a typical transfer scenario, the ***transferringDevice*** placed the ***primaryOldCall*** on hold, then originated the ***secondaryOldCall*** to ***transferredDevice***, and then transferred the call.

An application may use the transfer service to transfer a consultation call. Consultation calls make information about the original call available to an application monitoring the extension receiving the consultation call as soon as the consultation call alerts. Refer to the ***cstaConsultationCall( )*** manual page for more information.

- the ***transferredDevic*e** is a device identifier for the MERLIN LEGEND or MERLIN MAGIX switch transfer destination[6]. This is the device to which the call is transferred (it is *not*, as the TSAPI name suggests, the transfer source. A careful reading of TSAPI shows that the TSAPI name is misleading.) When the transfer destination is a station, or a Calling Group Queue, ***transferredDevice*** contains the extension for that station or Calling Group Queue. When the transfer destination is a trunk connection the ***transferredDevic*e** contains the MERLIN LEGEND or MERLIN MAGIX switch Facility Identifier for the trunk.

transferConnections provides applications with information so that they may continue to track calls when the call identifiers change as the transfer merges calls together. When a trunk connection is a party to the call, the ***transferConnections*** may contain the MERLIN LEGEND or MERLIN MAGIX switch Facility Identifier for the trunk. The MERLIN LEGEND and MERLIN MAGIX switches always supply the trunk identifier, ANI, ICLID or dialed digits, never a pool identifier. Each list item contains:

- a device identifier for a party on the call,
- the connection identifier for the call at that device after the transfer occurred.

6 The party doing the transferring is the *transfer originator*. The party being transferred is the *transfer source*. The party receiving the transferred call is the *transfer destination*. Thus, the ***activeCall*** parameter in a ***cstaConsultationCall( )*** is a connection at the transfer originator for the call at the transfer source. The ***calledDevice*** parameter in a ***cstaConsultationCall*** specifies the transfer destination.

NOTE:
An application should always check ***transferConnections*** to track connection and call identifiers as transfers occur. For the MERLIN LEGEND and MERLIN MAGIX switches, a transfer of the ***primaryOldCall*** and the ***secondaryOldCall*** always results in the ***secondaryOldCall*** being the call identifier for the call. There is no guarantee that this will continue to be true in future releases. In addition, not all switches operate in this manner, so a switch-independent application must use the ***transferConnections*** to track connection identifiers and call identifiers.

Event Parameters

Table 8-31. CSTATransferredEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_TRANSFERRED`
monitorCrossRefID	Transfer event occurred on this monitor
primaryOldCall	connection that was held for transfer
secondaryOldCall	connection that was active for transfer
transferringDevice	transferring device (transfer originator)
transferredDevice	device to which the call was transferred (transfer destination).
transferredConnections	list of connections on the transferred call. Each connection contains a device identifier and a call identifier.
localConnectionInfo	`CS_NONE`, none provided
cause	reason for Transferred event
privateData	`NULL`, not used for this event

Event Scenario Diagram

Figure 8-11 illustrates one possible ***CSTATransferredEvent*** scenario.

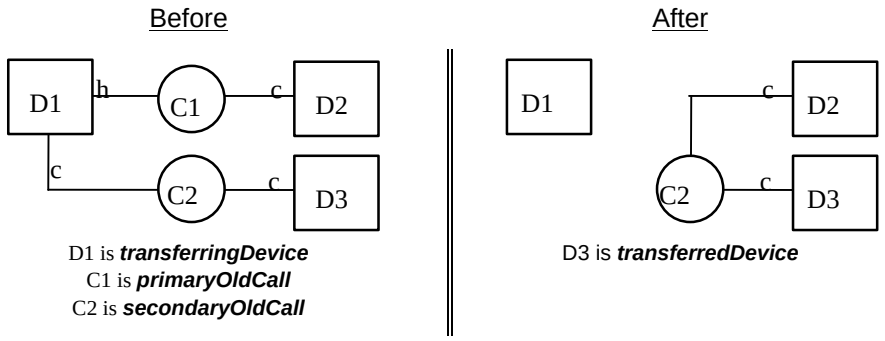


Figure 8-11. CSTATransferredEvent Scenario

Event Causes

Table 8-32. CSTATransferredEvent Causes

EC_NEW_CALL	The MERLIN LEGEND and MERLIN MAGIX switches provide EC_NEW_CALL on all ***CSTATransferredEvents***

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTATransferredEvent_t  transferred;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTATransferredEvent_t {
   ConnectionID_t          primaryOldCall;
   ConnectionID_t          secondaryOldCall;
   SubjectDeviceID_t       transferringDevice;
   SubjectDeviceID_t       transferredDevice;
   ConnectionList_t        transferredConnections;
   LocalConnectionState_t  localConnectionInfo;
   CSTAEventCause_t        cause;
} CSTATransferredEvent_t;
```

Important Feature Interactions

Coverage

For an unsupervised transfer, where the Transfer destination is a Coverage Sender, the ***transferredConnections*** in the ***CSTATransferredEvent*** contains the coverage sender, but will not contain any of the coverage receivers. If for some reason, the call does not alert at the coverage sender (e.g. Do Not disturb is active), then the Coverage Sender will also be absent from the ***transferredConnections.*** In either case, the transfer destination will appear as the ***transferredDevice*** in the ***CSTATransferredEvent.***

Group Calling (DGC)

When a user transfers a call to a Calling Group and the call goes directly to a DGC member, the ***transferredDevice*** parameter in the ***CSTATransferredEvent*** will contain the extension of the DGC member.

When a user transfers a call into a Calling Group and the call is queued, then the ***transferredDevice*** parameter will contain the DGC queue.

Networking

An application monitoring the transfer originator when the transfer destination is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network will receive a ***CSTATransferredEvent*** identifying the connections on the call.

An application monitoring the transfer destination when the transfer originator is on another MERLIN LEGEND or MERLIN MAGIX switch in the private network will receive a ***CSTADeliveredEvent*** and a ***CSTAEstablishedEvent*** for the consultation call, but these events will not contain any private data for the Original Call Information. The application will not receive a ***CSTATransferredEvent***.

Pool

When a user transfers a call to a Pool, the ***transferredDevice*** parameter in the ***CSTATransferredEvent*** will always contain an individual trunk identifier, not the Pool extension. Similarly, the ***transferConnections*** parameter contains an individual trunk identifier, not the Pool extension.

Direct Voice Mail

When an external call is transferred to a station's mailbox using Direct Voice Mail, the ***CSTATransferredEvent*** contains the extension number of the station in the list of transferred connections even though the station is not on the call. A ***CSTAConnectionClearedEvent*** is then generated to indicate that the station is not on the call.

Feature Events

9

Contents

Feature Events

9

Feature Events indicate a change in feature activation at a device.

Applications use Feature Events to track the activation or deactivation of the Do Not Disturb feature at an extension and to collect Account Code information. An application that needs to receive Feature Event Report for a device must:

- Open a stream using the Control Services (Chapter 3);
- Monitor that device using the Monitor Services (Chapter 6);
- Receive events using the Control Services (Chapter 3);

Table 9-1 shows the TSAPI Feature Events that the MERLIN MAGIX switch provide. Note that MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX Release 1.0 and 1.5 switches do not provide TSAPI Feature events. The MERLIN MAGIX (Release 2.0 and later) switch provides some but not all of the TSAPI Feature Events.

Table 9-1. MERLIN MAGIX CTI Support for TSAPI Feature Events

	TSAPI Feature Events for Monitored Stations - MERLIN MAGIX Release 2.0
	CSTACallInfoEventEvent
Ö	CSTADoNotDisturbEvent
	CSTAForwardingEvent
	CSTAMessageWaitingEvent

	TSAPI Feature Events for Monitored Stations - MERLIN MAGIX Release 2.1 and later
Ö	CSTACallInfoEventEvent
Ö	CSTADoNotDisturbEvent
	CSTAForwardingEvent
	CSTAMessageWaitingEvent

Event Page Format

The following pages in this chapter present the TSAPI Feature Report events that the MERLIN MAGIX switch provides to applications. Each TSAPI event description contains the following sections, as appropriate:

Event Name and Description

The event name appears first on the pages describing that event. Adescription of that event immediately follows the name.

Event Parameters

A table lists the event parameters and summarizes their use.

Event Syntax

This section contains C coding information for the event.

Important Feature Interactions

This section describes important interactions with the MERLIN MAGIX switch features that produce the event.

CSTACallInfoEvent

The ***CSTACallInfoEvent*** provides Account Code information entered by a user at a monitored station. The MERLIN MAGIX switch provides this event beginning with Release 2.1.

It is possible to receive multiple ***CSTACallInfoEvents*** for a single call, but only from a single extension. The MERLIN MAGIX switch blocks account codes being entered at more than one extension.

Event Parameters

Table 9-2. CSTACallInfoEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_CALL_INFO`
monitorCrossRefID	event occurred on this monitor
connection	connection (contains deviceID and callID)
device	Indicates from which extension account code information was entered
accountInfo	Specifies account code entered at device
authorisationCode	Not used
privateData	`NULL`

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files (acs.h, acsdefs.h, csta.h, and cstadefs.h) for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union
   {
      struct {
         cstaMonitorCrossRefID_t     monitorCrossRefID;
         union {
            CSTACallInfoEvent_t   callInformation;
         } u;
      } cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   ConnectionID_t     connection;
   SubjectDeviceID_t  device;
   AccountInfo_t      accountInfo;
   AuthCode_t         authorisationCode;
} CSTACallInfoEvent_t;

typedef char           AccountInfo_t[32];
typedef char           AuthCode_t[32];
```

Important Feature Interactions

Authorization

Authorization Codes are not provided by the event.

Intercom Calls

Account Codes can not be entered for intercom calls.

Transfer

An internal transfer destionation may enter an Account Code provided that one hasen't already been entered. Any internal party can enter an account code but once an Account Code has been entered only the extension that entered original Account can enter another Account Code.

CSTADoNotDisturbEvent

The ***CSTADoNotDisturbEvent*** indicates a change in status for the Do Not Disturb feature at an extension.

The MERLIN MAGIX switch provides this event beginning with MERLIN MAGIX Release 2.0.

Event Parameters

Table 9-3. CSTADoNotDisturbEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_DO_NOT_DISTURB`
monitorCrossRefID	event occurred on this monitor
device	station that has changed Do Not Disturb status
doNotDisturbOn	Specifies the state of Do Not Disturb (0 = deactivated, 1 = activated)
privateData	`NULL`, not used for this event

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTADoNotDisturbEvent_t doNotDisturb;
   } u;
} CSTAUnsolicitedEvent;
```

```
typedef struct CSTADoNotDisturbEvent_t {
   SubjectDeviceID_t       device;
   Boolean                 doNotDisturbOn;
} CSTADoNotDisturbEvent_t;
```

Important Feature Interactions

Extension Programming

If the Do Not Disturb feature is active at an extension and the Do Not Disturb button is deleted, the ***CSTADoNotDisturbEvent*** will be generated indicating that the feature has been deactivated.

Responding Mode

When a station transitions from non-responding to responding mode, its Do Not Disturb feature is deactivated regardless of its state prior to going non-responding. For MERLIN MAGIX Release 2.0, a ***CSTADoNotDisturbEvent*** is not generated when a station transitions from non-responding to responding mode, even though the status of the feature may have changed.

Beginning with MERLIN MAGIX Release 2.1, if a station goes non-responding while it has Do Not Disturb activated, a ***CSTADoNotDisturbEvent*** is generated to indicate that Do Not Disturb is deactivated. When the station returns to responding mode, the application and the station image are in sync.

Agent Status Events

10

Contents

Contents

Agent Status Events

10

Agent Status Events track changes in agent (Calling Group member) status occurring at a device. These changes may occur as a result of user activity at the device, call activity at the device, or the activity of a CTI application.

Applications use Agent Status Events to track agent login state and availability. Since agent status changes can occur at any time, these messages are asynchronous. An application that needs to receive Agent Status Events for a device must:

- Open a stream using the Control Services (Chapter 3);
- Monitor that device using the Monitor Services (Chapter 6);
- Receive events using the Control Services (Chapter 3).

Table 10-1 shows the TSAPI Agent Status Events that the MERLIN MAGIX switches provide. Note that MERLIN LEGEND (Release 5.0 and later) and MERLIN MAGIX Release 1.0 switches do not provide TSAPI Agent Status events. MERLIN MAGIX (Release 1.5 and 2.0) switches provide some but not all of the TSAPI Agent Status Events.

Table 10-1. MERLIN MAGIX CTI Support for TSAPI Agent Status Events

TSAPI Agent Status Events - MERLIN MAGIX Release 1.5
Ö CSTALoggedOnEvent
Ö CSTALoggedOffEvent
CSTANotReadyEvent
CSTAReadyEvent
Ö CSTAWorkNotReadyEvent
CSTAWorkReadyEvent

TSAPI Agent Status Events - MERLIN MAGIX Release 2.0
Ö CSTALoggedOnEvent
Ö CSTALoggedOffEvent
Ö CSTANotReadyEvent
Ö CSTAReadyEvent
Ö CSTAWorkNotReadyEvent
CSTAWorkReadyEvent

TSAPI Agent Status Events - MERLIN MAGIX Release 2.1 and later
Ö CSTALoggedOnEvent
Ö CSTALoggedOffEvent
Ö CSTANotReadyEvent
Ö CSTAReadyEvent
Ö CSTAWorkNotReadyEvent
Ö CSTAWorkReadyEvent

CAUTION:

When designing an application, be aware of the event parameters that the MERLIN MAGIX switch provides. The MERLIN MAGIX switch does not provide all of the optional TSAPI event parameters. The event manual pages list all of the TSAPI parameters and indicate those that the MERLIN MAGIX switch provides.

Event Page Format

The following pages in this chapter present the TSAPI agent events that the MERLIN MAGIX switch provides to applications. Each TSAPI event description contains the following sections, as appropriate:

Event Name and Description

The event name appears first on the pages describing that event. A description of that event immediately follows the name.

Event Parameters

A table lists the event parameters and summarizes their use.

Event Syntax

This section contains C coding information for the event.

Important Feature Interactions

This section describes important interactions with the MERLIN MAGIX switch features that produce the event.

CSTALoggedOffEvent

The ***CSTALoggedOffEvent*** indicates that station ***agentDevice*** has logged off. In MERLIN MAGIX terminology, the Extension Status of ***agentDevice*** has changed to Status 0 (Unavailable). This change may have occurred by any of the valid means available, including:

- the station user pressing a programmed Extension Status 2 button at ***agentDevice***
- the station user pressing a programmed Extension Status 1 button (to enter After Call Work state) at ***agentDevice***
- the station user dialing the Feature Code for Extension Status 0 at ***agentDevice***
- the DGC supervisor pressing either a programmed button or dialing a Feature Code to change ***agentDevice***'s status to Extension Status 0
- the successful completion of a ***cstaSetAgentState( )*** service request with ***agentMode*** `AM_LOG_OUT` on behalf of ***agentDevice***.

Beginning with MERLIN MAGIX Release 2.1, the there are additional valid means by the ***CSTALoggedOffEvent*** can occur:

- the station user pressing a programmed Logoff button and entering a ***agentGroup*** at ***agentDevice***
- the successful completion of a ***cstaSetAgentState( )*** service request with ***agentMode*** `AM_LOG_OUT` and ***agentGroup*** calling-group, on behalf of ***agentDevice***
- the station user dialing the Feature Code for Logoff and then entering an ***agentGroup*** at ***agentDevice***.

Because the MERLIN MAGIX switch supports members logging in and out independent of their Calling Group membership, this event is generated for an extension logging off even when the station is not a Calling Group member.

The MERLIN MAGIX switch provides this event beginning with Release 1.5.

Event Parameters

Table 10-2. CSTALoggedOffEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	CSTAUNSOLICITED
eventType	CSTA_LOGGED_OFF
monitorCrossRefID	event occurred on this monitor
agentDevice	station that has logged out
agentID	same as ***agentDevice***
agentGroup	the Calling Group that ***agentDevice*** has logged out off, or NULL
privateData	NULL, not used for this event

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTALoggedOffEvent_t loggedOff;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTALoggedOffEvent_t {
   SubjectDeviceID_t       agentDevice;
   AgentID_t               agentID;
   AgentGroup_t            agentGroup;
} CSTALoggedOffEvent_t;

typedef char            AgentID_t[32];
```

```
typedef DeviceID_t      AgentGroup_t;
```

Important Feature Interactions

Forwarding

A ***CSTALoggedOffEvent*** is generated when a Calling Group member activates forwarding, whether or not the member was available.

Group Calling (DGC)

A ***CSTALoggedOffEvent*** is generated regardless of how an agent logs out (either via a feature code, programmed button or successful ***cstaSetAgent-State( )*** request), even if the station was already logged out. If the station is not a member of a Calling Group, the ***agentGroup*** parameter will be `NULL`.

Beginning with MERLIN MAGIX Release 2.1, when the station is not a member of a Calling Group, and the agent performs a selective log out, the ***agentGroup*** parameter will contain the Calling Group that the agent logged out of.

CSTALoggedOnEvent

The ***CSTALoggedOnEvent*** indicates that station ***agentDevice*** has logged on. In MERLIN MAGIX terminology, the Extension Status of ***agentDevice*** has changed to Status 2 (Available). This change may have occurred by any of the valid means available, including:

- the station user pressing a programmed Extension Status 2 button at ***agentDevice***
- the station user dialing the Feature Code for Extension Status 2 at ***agentDevice***
- the DGC supervisor pressing either a programmed button or dialing a Feature Code to change ***agentDevice***'s status to Extension Status 2
- the successful completion of a ***cstaSetAgentState( )*** service request with ***agentMode*** `AM_LOG_IN` on behalf of ***agentDevice***.

Beginning with MERLIN MAGIX Release 2.1, the there are additional valid means by which the ***CSTALoggedOnEvent*** can occur:

- the station user pressing a programmed Logon button and entering an ***agentGroup*** at ***agentDevice***
- the successful completion of a ***cstaSetAgentState( )*** service request with ***agentMode*** `AM_LOG_IN` and ***agentGroup*** calling-group, on behalf of ***agentDevice***
- the station user dialing the Feature Code for Logon and then entering a Calling Group at ***agentDevice***.

Because the MERLIN MAGIX switch supports members logging in and out independent of their Calling Group membership, this event is generated for an extension logging in even when the station is not a Calling Group member.

The MERLIN MAGIX switch provides this event beginning with Release 1.5.

Event Parameters

Table 10-3. CSTALoggedOnEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_LOGGED_ON`
monitorCrossRefID	event occurred on this monitor
agentDevice	station that has logged in
agentID	same as ***agentDevice***
agentGroup	the Calling Group of which ***agentDevice*** is a member, or `NULL` if the ***agentDevice*** is not a Calling Group member.
password	`NULL`, not used for this event
privateData	`NULL`, not used for this event

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTALoggedOnEvent_t  loggedOn;
   } u;
} CSTAUnsolicitedEvent;
```

```
typedef struct CSTALoggedOnEvent_t {
   SubjectDeviceID_t       agentDevice;
   AgentID_t               agentID;
   AgentGroup_t            agentGroup;
   AgentPassword_t         password;
} CSTALoggedOnEvent_t;

typedef char            AgentID_t[32];

typedef DeviceID_t      AgentGroup_t;

typedef char            AgentPassword_t[32];
```

Important Feature Interactions

Group Calling (DGC)

The ***CSTALoggedOnEvent*** is generated regardless of how an agent logs in (either via a feature code, programmed button, or successful ***cstaSetAgent-State( )*** request), even if the station is already logged in. If the station is not a member of a Calling Group, the ***agentGroup*** parameter will be NULL.

Beginning with MERLIN MAGIX Release 2.1, if the station is not a member of a Calling Group, and the agent performs a selective login, the ***agentGroup*** parameter will contain the Calling Group that the agent logged into.

CSTANotReadyEvent

The ***CSTANotReadyEvent*** indicates that an extension has become unavailable to accept a Calling Group call. All monitored stations receive this event regardless of Calling Group membership. This event is generated when an extension is initially available to accept a call and then becomes *busy* through one of the following actions:

- the station goes off-hook
- the station activates the Do Not Disturb feature
- the station becomes non-responding (i.e., is unplugged)
- the station enters Program Mode
- the station enters Administration Mode
- the station enters Maintenance Mode
- the station enters Alarm Clock Mode
- the station enters Personal Directory Program Mode
- the station, port, or slot for the station is busied-out
- the station is forced-idle

The MERLIN MAGIX switch provides this event beginning with Release 2.0.

Event Parameters

Table 10-4. CSTANotReadyEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_NOT_READY`
monitorCrossRefID	event occurred on this monitor
agentDevice	station that is not ready
agentID	same as ***agentDevice***
privateData	`NULL`, not used for this event

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTANotReadyEvent_t  notReady;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTANotReadyEvent_t {
   SubjectDeviceID_t       agentDevice;
   AgentID_t               agentID;
} CSTANotReadyEvent_t;

typedef char            AgentID_t[32];
```

Important Feature Interactions

Administration/Maintenance Mode

An application monitoring the station that is the administration port will receive a ***CSTANotReadyEvent*** when the station enters Administration or Maintenance mode.

Alarm Clock Mode

An application monitoring an extension will receive a ***CSTANotReadyEvent*** when the station enters Alarm Clock Mode.

Busy-Out

An application monitoring an extension will receive a ***CSTANotReadyEvent*** when the station port or board for that station is busied-out.

Do Not Disturb

An application monitoring an extension will receive a ***CSTANotReadyEvent*** and a ***CSTADoNotDisturbEvent*** when the station activates Do Not Disturb.

Forced Idle

An application monitoring an extension will receive a ***CSTANotReadyEvent*** when the station is forced idled.

Group Calling

An application monitoring an extension will receive a ***CSTANotReadyEvent*** regardless of Calling Group membership.

Non-Responding Mode

An application monitoring a MLX, 4400-series or ETR station (not administered as Tip/Ring) will receive a ***CSTANotReadyEvent*** when the station is unplugged.

Off-Hook

An application monitoring an extension will receive a ***CSTANotReadyEvent*** when the station goes off-hook.

Personal Directory

An application monitoring an extension will receive a ***CSTANotReadyEvent*** when the station enters Personal Directory Program Mode, either at the station or via Administration.

Program Mode

An application monitoring an extension will receive a ***CSTANotReadyEvent*** when the station enters Program Mode or when the station is being programmed via Centralized Station Programming.

CSTAReadyEvent

The ***CSTAReadyEvent*** indicates that an extension is available to accept a Calling Group call. All monitored stations receive this event regardless of Calling Group membership. This event is generated when an extension is initially busy and then becomes not busy through one or more of the following actions:

- the station goes on-hook
- the station deactivates the Do Not Disturb feature
- the station that was non-responding (i.e., is unplugged) becomes responding
- the station leaves Program Mode
- the station leaves Administration Mode
- the station leaves Maintenance Mode
- the station leaves Alarm Clock Mode
- the station leaves Personal Directory Program Mode
- the station, port, or slot for the station is restored from a busy-out

The MERLIN MAGIX switch provides this event beginning with Release 2.0.

Event Parameters

Table 10-5. CSTAReadyEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_READY`
monitorCrossRefID	event occurred on this monitor
agentDevice	station that is ready
agentID	same as ***agentDevice***
privateData	`NULL`, not used for this event

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAReadyEvent_t  ready;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAReadyEvent_t {
   SubjectDeviceID_t       agentDevice;
   AgentID_t               agentID;
} CSTAReadyEvent_t;

typedef char            AgentID_t[32];
```

Important Feature Interactions

Administration/Maintenance Mode

An application monitoring the station that is the administration port may receive a ***CSTAReadyEvent*** when the station leaves Administration or Maintenance mode.

Alarm Clock Mode

An application monitoring an extension may receive a ***CSTAReadyEvent*** when the station leaves Alarm Clock Mode.

Busy-Out

An application monitoring an extension may receive a ***CSTAReadyEvent*** when the station port or board for that station is restored after a busy-out.

Do Not Disturb

An application monitoring an extension may receive a ***CSTAReadyEvent*** and a ***CSTADoNotDisturbEvent*** when the station deactivates Do Not Disturb.

Group Calling

An application monitoring an extension will receive a ***CSTAReadyEvent*** regardless of Calling Group membership.

Headset

An application monitoring an extension in headset mode will receive a ***CSTAReadyEvent*** when the far end hangs up the call.

Non-Responding Mode

An application monitoring a non-responding MLX, 4400-series or ETR station (not administered as a Tip/Ring) may receive a ***CSTAReadyEvent*** if the station that is non-responding is plugged in.

On-Hook

An application monitoring an extension may receive a ***CSTAReadyEvent*** when the station goes on-hook.

Personal Directory

An application monitoring an extension may receive a ***CSTAReadyEvent*** when the ***station*** leaves Personal Directory Programming (either at the station or via Administration).

Program Mode

An application monitoring an extension may receive a ***CSTAReadyEvent*** when the station leaves Program Mode or when Centralized Station Programming is exited for that station.

CSTAWorkNotReadyEvent

The MERLIN MAGIX switch provides this event beginning with Release 1.5.

The ***CSTAWorkNotReadyEvent*** indicates that station ***agentDevice*** has gone into After Call Work mode (Extension Status 1) or Auxiliary Work Time (beginning with MERLIN MAGIX Release 2.1. The change in agent status may have occurred by any of the valid means available, including:

- the station user pressing a programmed Extension Status 1 (After Call Work) button at ***agentDevice***
- the station user pressing a programmed Auxiliary Work Time button at ***agentDevice***
- the station user dialing the Feature Code for Extension Status 1 at ***agentDevice***
- the station user dialing the Feature Code for Auxiliary Work Time at ***agentDevice***
- the DGC supervisor pressing either a programmed button or dialing a Feature Code to change ***agentDevice***'s status to Extension Status 1
- the DGC supervisor pressing either a programmed button or dialing a Feature Code to change ***agentDevice***'s mode to Auxiliary Work Time
- the successful completion of a ***cstaSetAgentState( )*** service request with ***agentMode*** `AM_WORK_NOT_READY` on behalf of ***agentDevice***.

Beginning with MERLIN MAGIX Release 2.1, an agent station can be assigned membership in multiple Calling Groups. As part of this change the Auxiliary Work Time mode was introduced. Systems configured with multi-group membership for agent stations will generally use Auxilary Work Time buttons rather than After Call Work buttons.

Because the MERLIN MAGIX switch supports members entering After Call Work or Auxiliary Wort Time mode independent of their Calling Group membership, this event is generated for an extension entering After Call Work or Auxiliary Work Time mode, even if the station is not a Calling Group member.

Event Parameters

Table 10-6. CSTAWorkNotReadyEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_WORK_NOT_READY`
monitorCrossRefID	event occurred on this monitor
agentDevice	station that has gone into Auxiliary Work Time or After Call Work mode
agentID	same as ***agentDevice***
privateData	`NULL`, not used for this event

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAWorkNotReadyEvent_t workNotReady;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAWorkNotReadyEvent_t {
   SubjectDeviceID_t       agentDevice;
   AgentID_t               agentID;
} CSTAWorkNotReadyEvent_t;

typedef char            AgentID_t[32];
```

Important Feature Interactions

Group Calling (DGC)

An application monitoring an extension will receive a ***CSTAWorkNotReadyEvent*** regardless of how the extension enters Auxiliary Work Time or After Call Work mode, even if the station was already in Auxiliary Work Time or After Call Work mode, and regardless of Calling Group membership.

CSTAWorkReadyEvent

The MERLIN MAGIX switch provides this event beginning with Release 2.1.

The ***CSTAWorkReadyEvent*** indicates that station ***agentDevice*** has exited out of After Call Work mode (Extension Status 1) or Auxiliary Work Time (beginning with MERLIN MAGIX Release 2.1). This event indicates a transition from agent state, AG_WORK_NOT_READY to AG_WORK_READY. This transition may have occurred by any of the valid means available, including:

- the station user pressing a programmed After Call Work button at ***agentDevice***
- the station user pressing a programmed Auxiliary Work Time button at ***agentDevice***
- the station user dialing the Feature Code for After Call Work button at ***agentDevice***
- The station user dialing the Feature Code for Auxiliary Work Time at ***agentDevice***
- the DGC supervisor pressing either a programmed button or dialing a Feature Code to take ***agentDevice*** out of After Call Work mode
- the DGC supervisor pressing either a programmed button or dialing a Feature Code to take ***agentDevice*** out of Auxiliary Work Time mode
- the successful completion of a ***cstaSetAgentState( )*** service request with ***agentMode*** AM_WORK_READY on behalf of ***agentDevice***.

Beginning with MERLIN MAGIX Release 2.1, an agent station can be assigned membership in multiple Calling Groups. As part of this change the Auxiliary Work Time mode was introduced. Systems configured with multi-group membership for agent stations will generally use Auxilary Work Time buttons rather than After Call Work buttons.

Because the MERLIN MAGIX switch supports members entering the After Call Work or Auxiliary Work Time mode independent of their Calling Group membership, this event is generated for an extension exiting After Call Work or Auxiliary Work Time mode, even if the station is not a Calling Group member.

Event Parameters

Table 10-7. CSTAWorkReadyEvent Parameters

acsHandle	ACS stream on which event arrived
eventClass	`CSTAUNSOLICITED`
eventType	`CSTA_WORK_READY`
monitorCrossRefID	event occurred on this monitor
agentDevice	station exiting Agent Work Time of After Call Work mode
agentID	same as ***agentDevice***
privateData	`NULL`, not used for this event

Event Syntax

The syntax below shows only the relevant portions of structures and unions. Refer to the TSAPI header files for a complete description.

```
typedef struct {
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct {
   ACSEventHeader_t  eventHeader;
   union {
      CSTAUnsolicitedEvent cstaUnsolicited;
   } event;
} CSTAEvent_t;

typedef struct {
   CSTAMonitorCrossRefID_t    monitorCrossRefId;
   union {
      CSTAWorkReadyEvent_t workReady;
   } u;
} CSTAUnsolicitedEvent;

typedef struct CSTAWorkReadyEvent_t {
   SubjectDeviceID_t       agentDevice;
   AgentID_t               agentID;
} CSTAWorkReadyEvent_t;

typedef char            AgentID_t[32];
```

Important Feature Interactions

Group Calling (DGC)

An application monitoring an extension will receive a ***CSTAWorkReadyEvent*** regardless of how an extension exits Auxiliary Work Time or After Call Work mode, even if the station was not in Auxiliary Work Time or After Call Work mode, and regardless of Calling Group membership.

Escape Services

11

Contents

Contents

Contents

Contents

Escape Services

11

Applications use Escape Services to obtain access to switch services which are not defined by the CSTA standard.

Table 11-1 shows the Escape Services supported beginning with MERLIN MAGIX Release 2.0.

Table 11-1. MERLIN MAGIX CTI Support for Escape Services

Escape Services - MERLIN MAGIX Release 2.0
mlGetDGCGroupList( ) and mlGetDGCGroupListConfEvent
mlGetDGCGroupMemberList( ) and mlGetDGCGroupMemberListConfEvent
mlGetDGCGroupTrunkList( ) and mlGetDGCGroupTrunkListConfEvent
mlQueryDGCQueueStatus( ) and mlQueryDGCQueueStatusConfEvent
mlQueryDeviceName( ) and mlQueryDeviceNameConfEvent
mlQueryTrunkStatus( ) and mlQueryTrunkStatusEvent

Escape Services - MERLIN MAGIX Release 2.1 and later
mlGetDGCGroupList( ) and mlGetDGCGroupListConfEvent
mlGetDGCGroupMemberList( ) and mlGetDGCGroupMemberListConfEvent
mlGetDGCGroupTrunkList( ) and mlGetDGCGroupTrunkListConfEvent
mlQueryDGCQueueStatus( ) and mlQueryDGCQueueStatusConfEvent
mlQueryDeviceName( ) and mlQueryDeviceNameConfEvent
mlQueryTrunkStatus( ) and mlQueryTrunkStatusEvent
mlQueryDGCGroupDAUInfo( ) and mlQueryDGCGroupDAUInfoConfEvent
mlQueryDGCGroupParameters( ) and mlQueryDGCGroupParametersConfEvent

Requesting Escape Services and Receiving Confirmations

For each MERLIN MAGIX Escape Service, a private data function is provided in the private data library. The private data function initializes a private data buffer with the service type and service parameters. The application calls ***cstaEscapeService( )*** with a pointer to the initialized buffer to invoke the service request.

Each Escape Service request has an associated confirmation event. Some Escape Services also result in the application receiving private events. This book presents information about each service's confirmation event (and private event) under the heading for the service.

An application must receive the confirmation event (or private event) on the stream where it sends the Escape Service request. "Receiving Events" in Chapter 3 describes how applications receive confirmation events. "Extracting Private Data from Events" in Chapter 2 explains how an application extracts MERLIN MAGIX private data from the events

Confirmations have different meanings for various services. Refer to the manual page for each service when coding applications so as to use the service confirmations properly. In some cases, an application must wait for subsequent Private Events to receive the results of a query.

Escape Service Request Failures

If the service request fails for some reason, the application will receive a ***CSTAUniversalFailureConfEvent*** in place of the service confirmation. Each service description includes a list of the ***error*** values that the ***CSTAUniversalFailureConfEvent*** may carry for that service as well as the meanings of those values in the context of that service. Since the ***CSTAUniversalFailureConfEvent*** applies to other services, as well as Escape services, its description is found in the section pertaining to ***CSTAUniversalFailureConfEvent*** in Chapter 3.

Escape Service Page Format

The pages describing each TSAPI escape service contain the following sections, as appropriate:

Service Name and Description

The service name appears first. A description of that service immediately follows the name.

Service Request Parameters

A table lists the service request parameters and summarizes their use.

Private Service Request Parameters

A table lists the private service parameters and summarizes their use.

Return Values

A table lists the return values for the service request.

In all function returns, success values follow the TSAPI rules. If the requesting application generated the ***invokeID*** value, then a successful function call returns zero. If the TSAPI library generates the ***invokeID*** value, then a successful function call returns the value of the ***invokeID***. This is not explicitly re-stated for each service. “Sending TSAPI Requests and Receiving Confirmations” in Chapter 3 describes ***invokeID*** usage in more detail.

Confirmation Event

This section names the TSAPI confirmation event for the service and contains a table describing the confirmation event parameters.

Confirmation Event Private Data

This section names the MERLIN MAGIX private data confirmation event for the service and contains a table describing the private confirmation event parameters.

Private Event Parameters

This section names the MERLIN MAGIX private event for the service and contains a table describing the private event parameters for the service.

CSTA Universal Failure Confirmation Event Error Values

This section lists error values that the ***CSTAUniversalFailureConfEvent*** may return to an application when a service request fails. Items in all capitals are #defines from the TSAPI header files (acs.h, acsdefs.h, csta.h, and cstadefs.h).

Request Syntax

This section contains C coding information for the service request.

Confirmation Event Syntax

This section contains C coding information for the service's confirmation event.

Private Event Syntax

This section contains C coding information for the service's private event.

Important Feature Interactions

This section describes important interactions between the Escape Service and MERLIN MAGIX switch features.

mlGetDGCGroupList( )

The ***mlGetDGCGroupList( )*** escape service is introduced in MERLIN MAGIX Release 2.0 with private data version 2. This service allows an application to obtain a list of local Calling Groups. Once an application has the Calling Group IDs, it is able to monitor the Calling Groups. This enables an application to do statistical reporting or to manage Calling Group calls.

To allow applications to operate more efficiently, the list of Calling Groups generated by the ***mlGetDGCGroupList( )*** escape service only includes a Calling Group if at least one of the following conditions are met:

- a local member is assigned to the Calling Group
- a line is assigned to the Calling Group
- a pool is assigned to the Calling Group
- a Primary Delay Announcement Unit is assigned to the Calling Group
- a Secondary Delay Announcement Unit is assigned to the Calling Group
- an external alert is administered for the Calling Group
- an overflow group is administered for the Calling Group
- a support group is administered for the Calling Group

The list of Calling Groups generated by the ***mlGetDGCGroupList( )*** escape service will not include non-local Calling Groups.

Because the volume of data requested by this service may be large, the actual list of Calling Groups is not returned in the confirmation event. The confirmation event provides a unique private event cross-reference ID that associates subsequent ***CSTAPrivateEvent***s (containing the actual list of Calling Groups) with the original request. The private event cross reference ID is the only data returned in the confirmation event.

After returning the confirmation event, the service returns a sequence of ***CSTAPrivateEvent***s. Each ***CSTAPrivateEvent*** contains the private event cross reference ID, and a list. The list contains the number of elements in the message, and up to ten Calling Groups.

The service provides the private event cross reference ID in case an application has issued multiple ***mlGetDGCGroupList( )*** requests. The final ***CSTAPrivateEvent*** specifies that it contains zero Calling Groups and serves to inform the application that no more messages will be sent in response to this query.

Service Request Parameters

Table 11-2. cstaEscapeService( ) Parameters for mlGetDGCGroupList( )

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized by ***mlGetDGCGroupList( )***

Return Values

Table 11-3. cstaEscapeService( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch has accepted the request.

Table 11-4. CSTAEscapeServiceConfEvent Parameters for mlGetDGCGroupList( )

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_ESCAPE_SVC_CONF
invokeID	identifies service request within stream
privateData	private data buffer containing an ML_GETDGC_GROUP_LIST_CONF event

Confirmation Event Private Data

The ***CSTAEscapeServiceConfEvent*** will contain MERLIN MAGIX private data.

Table 11-5. mlGetDGCGroupList( ) Confirmation Event Private Data Parameters

eventType	`ML_GETDGC_GROUP_LIST_CONF`
privEventCrossRefID	a unique ID that associates subsequent ***CSTAPrivateEvent***s with this request

Private Event Parameters

Following the receipt of the ***CSTAEscapeServiceConfEvent***, the application will receive one or more ***CSTAPrivateEvent***s containing MERLIN MAGIX private data.

Table 11-6. mlGetDGCGroupList( ) CSTAPrivateEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTAEVENTREPORT`
eventType	`CSTA_PRIVATE`
privateData	private data buffer containing an `ML_GETDGC_GROUP_LIST_RESP` event

Table 11-7. mlGetDGCGroupList( ) CSTAPrivateEvent Private Data Parameters

eventType	`ML_GETDGC_GROUP_LIST_RESP`
privEventCrossRefID	a unique ID that associates this ***CSTAPrivateEvent*** with the service request
list	a list structure containing the following information: a ***count*** (0-10) of how many Calling Group IDs are in this response, and an array (***groupID[ ]***) containing up to 10 Calling Group IDs. A count of 0 indicates that this is the last ***CSTAPrivateEvent*** for the service request.

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***mlGetDGCGroupList( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` – An application will receive `GENERIC_UNSPECIFIED` when the request could not be satisfied for a reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` – The CTI link is disconnected or not in service.

`INVALID_FEATURE` – The application requested the escape service on a stream opened with private data version 1 stream, or on a stream opened without private data.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` – Processing the ***mlGetDGCGroupList( )*** request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` – The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` – A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlGetDGCGroupList (MLPrivateData_t     *privateData);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
              InvokeID_t            invokeID,         /* INPUT */
              PrivateData_t         *privateData);    /* INPUT */
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct  MLEvent_t {
   MLEventType_t  eventType;
   union
   {
      MLGetDGCGroupListConfEvent_t     getDGCGroupList;
   } u;
} MLEvent_t;

typedef struct MLGetDGCGroupListConfEvent_t
{
   MLPrivEventCrossRefID_t privEventCrossRefID;
} MLGetDGCGroupListConfEvent_t;
```

Private Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAEventReport   cstaEventReport;
   } event;
} CSTAEvent_t;

typedef struct
{
   union
   {
      CSTAPrivateEvent_t   privateEvent;
   } u;
} CSTAEventReport;

typedef struct CSTAPrivateEvent_t
{
   Nulltype       null;
} CSTAPrivateEvent_t;

   union
   {
      MLGetDGCGroupListResp_t          MLGetDGCGroupListResp;
   } u;
} MLEvent_t;

typedef struct  MLGetDGCGroupListResp_t
{
   MLPrivEventCrossRefID_t privEventCrossRefID;
   struct
   {
      short          count;
      DeviceID_t     groupID[10];
   } list;
} MLGetDGCGroupListResp_t;
```

Important Feature Interactions

Group Calling (DGC)

The list of Calling Groups generated by the ***mlGetDGCGroupList( )*** escape service will only include a Calling Group if at least one of the following conditions are met:

- a local member is assigned to the Calling Group
- a line is assigned to the Calling Group
- a pool is assigned to the Calling Group
- a Primary Delay Announcement Unit is assigned to the Calling Group
- a Secondary Delay Announcement Unit is assigned to the Calling Group
- an external alert is administered for the Calling Group
- an overflow group is administered for the Calling Group
- a support group is administered for the Calling Group

Networking

The list of Calling Groups generated by the ***mlGetDGCGroupList( )*** escape service will not include any Calling Group that contains a non-local member.

Renumbering

If Calling Groups are renumbered on the switch, a subsequent ***mlGetDGC-GroupList( )*** escape service request will return the new Calling Group numbers.

mlGetDGCGroupMemberList( )

The ***mlGetDGCGroupMemberList( )*** escape service is introduced in MERLIN MAGIX Release 2.0 with private data version 2. The service allows an application to obtain a list of Calling Group members for a specific local Calling Group.

Because the volume of data requested by this service may be large, the actual list of Calling Group members is not returned in the confirmation event. The confirmation event provides a unique private event cross reference ID that associates subsequent ***CSTAPrivateEvent***s (containing the actual list of Calling Group members) with the original request. The private event cross reference ID is the only data returned in the confirmation event.

After returning the confirmation event, the service returns a sequence of ***CSTAPrivateEvent***s. Each ***CSTAPrivateEvent*** contains the private event cross reference ID, and a list. The list contains the number of elements in the message, and up to 10 Calling Group members.

The service provides the private event cross reference ID in case an application has issued multiple ***mlGetDGCGroupMemberList( )*** requests. The final ***CSTAPrivateEvent*** specifies that it contains zero Calling Group members and serves to inform the application that no more messages will be sent in response to this query.

Service Request Parameters

Table 11-8. cstaEscapeService( ) Parameters for mlGetDGCGroupMember-List( )

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized by calling ***mlGetDGCGroupMemberList( )***

Private Service Request Parameters

Table 11-9. mlGetDGCGroupMemberList( ) Private Service Request Parameters

dgcID	Calling Group number of the group being queried

Return Values

Table 11-10. cstaEscapeService( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch has accepted the request.

Table 11-11. CSTAEscapeServiceConfEvent Parameters for mlGetDGCGroupMemberList( )

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_ESCAPE_SVC_CONF`
invokeID	identifies service request within stream
privateData	private data buffer containing an `ML_GETDGC_GROUP_MEMBER_LIST_CONF` event

Confirmation Event Private Data

The ***CSTAEscapeServiceConfEvent*** will contain MERLIN MAGIX private data.

Table 11-12. mlGetDGCGroupMemberList( ) Private Confirmation Event Private Data Parameters

eventType	`ML_GETDGC_GROUP_MEMBER_LIST_CONF`
privEventCrossRefID	a unique ID that associates subsequent ***CSTAPrivateEvent***s with this request

Private Event Parameters

Following the receipt of the ***CSTAEscapeServiceConfEvent***, the application will receive one or more ***CSTAPrivateEvent***s containing MERLIN MAGIX private data.

Table 11-13. mlGetDGCGroupMemberList( ) CSTAPrivateEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTAEVENTREPORT`
eventType	`CSTA_PRIVATE`
privateData	private data buffer containing an `ML_GETDGC_GROUP_MEMBER_LIST_RESP` event

Table 11-14. mlGetDGCGroupMemberList( ) CSTAPrivateEvent Private Data Parameters

eventType	`ML_GETDGC_GROUP_MEMBER_LIST_RESP`
privEventCrossRefID	a unique ID that associates this ***CSTAPrivateEvent*** with the service request
list	a list structure containing the following information: a ***count*** (0-10) of how many Calling Group members are in this response, and an array (***agentID[ ]***) containing up to 10 Calling Group members. A count of 0 indicates that this is the last ***CSTAPrivateEvent*** for the service request.

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***mlGetDGCGroupMemberList( )*** request, the ***CSTAUniversalFailure-ConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_UNSPECIFIED` when the request could not be satisfied for a reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` - ***dgcID*** is not a valid Calling Group number.

`INVALID_OBJECT_TYPE` - ***dgcID*** is not a local Calling Group number (i.e., the Calling Group contains a networked member).

`INVALID_FEATURE` - The application requested the escape service on a stream opened with private data version 1 stream, or on a stream opened without private data.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` - Processing the ***mlGetDGCGroupMemberList( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlGetDGCGroupMemberList (MLPrivateData_t     *privateData,
              DeviceID_t                     *dgcID);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
              InvokeID_t            invokeID,         /* INPUT */
              PrivateData_t         *privateData);    /* INPUT */
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct MLEvent_t
{
   MLEventType_t  eventType;
   union
   {
      MLGetDGCGroupMemberListConfEvent_t     getDGCGroupMemberList;
   } u;
} MLEvent_t;

typedef struct MLGetDGCGroupMemberListConfEvent_t
{
   MLPrivEventCrossRefID_t    privEventCrossRefID;
} MLGetDGCGroupMemberListConfEvent_t;
```

Private Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAEventReport   cstaEventReport;
   } event;
} CSTAEvent_t;

typedef struct
{
   union
   {
      CSTAPrivateEvent_t   privateEvent;
   } u;
} CSTAEventReport;

typedef struct CSTAPrivateEvent_t
{
   Nulltype       null;
} {
   MLEventType_t eventType;
   union
   {
      MLGetDGCGroupMemberListResp_t    MLGetDGCGroupMemberListResp;
   } u;
} MLEvent_t;

typedef struct MLGetDGCGroupMemberListResp_t
{
   MLPrivEventCrossRefID_t privEventCrossRefID;
   struct
   {
      short          count;
      DeviceID_t     agentID[10];
   } list;
} MLGetDGCGroupMemberListResp_t;
```

Important Feature Interactions

Group Calling (DGC)

The administered parameters for a local Calling Group have no affect on the success or failure of the ***mlGetDGCGroupMemberList( )*** escape service request. If there are no local members in the Calling Group, the service request will be successful, but will indicate that the Calling Group contains zero members.

Networking

If an application requests the ***mlGetDGCGroupMemberList( )*** escape service for a Calling Group containing a non-local member, the service request is denied.

Renumbering

If Calling Group members are renumbered on the switch, a subsequent ***mlGetDGCGroupMemberList( )*** escape service request will return the new extension numbers for the Calling Group members.

mlGetDGCGroupTrunkList( )

The ***mlGetDGCGroupTrunkList( )*** escape service is introduced in MERLIN MAGIX Release 2.0 with private data version 2. The service allows an application to obtain a list of the lines and trunks assigned to a specific local Calling Group.

Because the volume of data requested by this service may be large, the actual list of lines and trunks is not returned in the confirmation event. The confirmation event provides a unique private event cross reference ID that associates subsequent ***CSTAPrivateEvent***s (containing the actual list of lines and trunks assigned to the Calling Group) with the original request. The private event cross reference ID is the only data returned in the confirmation event.

After returning the confirmation event, the service returns a sequence of ***CSTAPrivateEvent***s. Each ***CSTAPrivateEvent*** contains the private event cross reference ID, and a list. The list contains the number of elements in the message, and up to 10 trunk identifiers.

The service provides the private event cross reference ID in case an application has issued multiple ***mlGetDGCGroupTrunkList( )*** requests. The final ***CSTAPrivateEvent*** specifies that it contains zero trunk identifiers and serves to inform the application that no more messages will be sent in response to this query.

Service Request Parameters

Table 11-15. cstaEscapeService( ) Parameters for mlGetDGCGroupTrunkList( )

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized by calling ***mlGetDGCGroupTrunkList( )***

Private Service Request Parameters

Table 11-16. mlGetDGCGroupTrunkList( ) Private Service Request Parameters

dgcID	Calling Group number of the group being queried

Return Values

Table 11-17. cstaEscapeService( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch has accepted the request.

Table 11-18. CSTAEscapeServiceConfEvent Parameters for mlGetDGCGroupTrunkList( )

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_ESCAPE_SVC_CONF`
invokeID	identifies service request within stream
privateData	private data buffer containing an `ML_GETDGC_GROUP_TRUNK_LIST_CONF` event

Confirmation Event Private Data

Table 11-19. mlGetDGCGroupTrunkList( ) Confirmation Event Private Data Parameters

eventType	`ML_GETDGC_GROUP_TRUNK_LIST_CONF`
privEventCrossRefID	a unique ID that associates subsequent ***CSTAPrivateEvent***s with this request

Private Event Parameters

Following the receipt of the ***CSTAEscapeServiceConfEvent***, the application will receive one or more ***CSTAPrivateEvent***s containing MERLIN MAGIX private data.

Table 11-20. mlGetDGCGroupTrunkList( ) CSTAPrivateEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTAEVENTREPORT`
eventType	`CSTA_PRIVATE`
privateData	private data buffer containing an `ML_GETDGC_GROUP_TRUNK_LIST_RESP` event

Private Event Private Data

Table 11-21. mlGetDGCGroupTrunkList( ) CSTAPrivateEvent Private Data Parameters

eventType	`ML_GETDGC_GROUP_TRUNK_LIST_RESP`
privEventCrossRefID	a unique ID that associates this ***CSTAPrivateEvent*** with the service request
list	a list structure containing the following information: a ***count*** (0-10) of how many trunk identifiers are in this response, and an array (***trunkID[ ]***) containing up to 10 trunk identifiers. A count of 0 indicates that this is the last ***CSTAPrivateEvent*** for the service request.

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***mlGetDGCGroupTrunkList( )*** request, the ***CSTAUniversalFailure-ConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_UNSPECIFIED` when the request could not be satisfied for a reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` - ***dgcID*** is not a valid Calling Group number.

`INVALID_OBJECT_TYPE` - ***dgcID*** is not a local Calling Group number (i.e., the Calling Group contains a networked member).

`INVALID_FEATURE` - The application requested the escape service on a stream opened with private data version 1 stream, or on a stream opened without private data.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` - Processing the ***mlGetDGCGroupTrunkList( )*** exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlGetDGCGroupTrunkList(MLPrivateData_t       *privateData,
              DeviceID_t                     *dgcID);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
              InvokeID_t            invokeID,         /* INPUT */
              PrivateData_t         *privateData)     /* INPUT */;
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct MLEvent_t
{
   MLEventType_t  eventType;
   union
   {
      MLGetDGCGroupTrunkListConfEvent_t   getDGCGroupTrunkList;
   } u;
} MLEvent_t;

typedef struct MLGetDGCGroupTrunkListConfEvent_t
{
   MLPrivEventCrossRefID_t privEventCrossRefID;
} MLGetDGCGroupTrunkListConfEvent_t;
```

Private Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAEventReport   cstaEventReport;
   } event;
} CSTAEvent_t;

typedef struct
{
   union
   {
      CSTAPrivateEvent_t   privateEvent;
   } u;
} CSTAEventReport;

typedef struct CSTAPrivateEvent_t
{
   Nulltype       null;
} CSTAPrivateEvent_t;

typedef struct  MLEvent_t
{
   MLEventType_t eventType;
   union
   {
      MLGetDGCGroupTrunkListResp_t     MLGetDGCGroupTrunkListResp;
   } u;
} MLEvent_t;

typedef struct MLGetDGCGroupTrunkListResp_t
{
   MLPrivEventCrossRefID_t privEventCrossRefID;
   struct
   {
      short          count;
      DeviceID_t     trunkID[10];
   } list;
} MLGetDGCGroupTrunkListResp_t;
```

Important Feature Interactions

Networking

If an application requests the ***mlGetDGCGroupTrunkList( )*** escape service for a Calling Group containing a non-local member, the service request will be denied with error `INVALID_OBJECT_TYPE`.

Pools

If a pool is administered to ring into the Calling Group, the list of trunk identifiers returned by the ***mlGetDGCGroupTrunkList( )*** escape service will return contain the individual lines assigned to the pool, not the pool code.

Renumbering

If lines assigned to the Calling Group are renumbered on the switch, a subsequent ***mlGetDGCGroupTrunkList( )*** escape service request will return the new trunk identifiers.

mlQueryDGCGroupDAUInfo( )

The ***mlQueryDGCGroupDAUInfo( )*** escape service is introduced in MERLIN MAGIX 2.1 with private data version 3. This service allows an application to obtain details on how a DGC Group is configured based on its programmable parameters pertaining to Delayed Announcement Units (DAUs). Other DGC Group information is provided by other services.

Service Request Parameters

Table 11-22. cstaEscapeService( ) Parameters for mlQueryDGCGroupDAUInfo( )

acsHandle	ACS stream on which service request being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized with ***mlQueryDGCGroupDAUInfo( )***

Private Service Request Parameters

Table 11-23. mlQueryDGCGroupDAUInfo( ) Private Service Request Parameters

dgcID	Calling Group number of the group being queried

Return Values

Table 11-24. cstaEscapeService( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch has accepted the request.

Table 11-25. CSTAEscapeServiceConfEvent Parameters for mlQueryDGCGroupDAUInfo( )

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_ESCAPE_SVC_CONF`
invokeID	identifies service request within stream
privateData	private data buffer containing an `ML_QUERYDGC_GROUPDAU_INFO_CONF` event

Confirmation Event Private Data

Table 11-26. mlQueryDGCGroupDAUInfo( ) Confirmation Event Private Data Parameters

eventType	`ML_QUERYDGC_GROUPDAU_INFO_CONF`
primaryDAUList	a list of structure containing the following information: a ***count*** (0-10) of how many primary DAUs are in the list, and an array (***primaryDAU[ ]***) containing up to 10 primary DAU extension numbers.
secondaryDAU	extension number of secondary DAU
primaryAnnList	a list of structure containing the following information: a ***count*** (0-10) of how many primary announcement numbers are in the list, and an array (***primaryAnn[ ]***) containing up to 10 announcement numbers.
secondaryAnn	secondary announcement number

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***mlQueryDGCGroupDAUInfo ( )*** request, the ***CSTAUniversalFailure-ConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_UNSPECIFIED` when the request could not be satisfied for a reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` - ***dgcID*** is not a valid Calling Group number.

`INVALID_OBJECT_TYPE` - ***dgcID*** is not a local Calling Group number (i.e., the Calling Group contains a networked member).

`INVALID_FEATURE` - The application requested the escape service on a stream opened with a private data version less than version 3, or on a stream opened without private data.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` - Processing the ***mlQueryDGCGroupDAUInfo ( )*** request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlQueryDGCGroupParameters (MLPrivateData_t      *privateData,
              DeviceID_t                        *dgcID);

typedef struct MLPrivateData_t
{
   char             vendor[32];
   unsigned short   length;
   char             data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
              InvokeID_t            invokeID,         /* INPUT */
              PrivateData_t         *privateData)     /* INPUT */;
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct MLEvent_t
{
   MLEventType_t  eventType;
   union
   {
      MLQueryDGCGroupDAUInfoConfEvent_t   queryDGCGroupDAUInfo;
   } u;
} MLEvent_t;
```

```
typedef struct MLQueryDGCGroupDAUInfoConfEvent_t
{
   struct
   {
      short       count;
      DeviceID_t  primaryDAU[10];
   } primaryDAUList;
   DeviceID_t     secondaryDAU;
   struct
   {
      short       count;
      DeviceID_t  primaryAnn[10];
   } primaryAnnList;
   short    secondaryAnn;
} MLQueryDGCGroupDAUInfoConfEvent_t;
```

Important Feature Interactions

Networking

If an application requests the ***mlQueryDGCGroupParameters( )*** escape service for a Calling Group containing a non-local member, the service request is denied.

mlQueryDGCGroupParameters( )

The ***mlQueryDGCGroupParameters( )*** escape service is introduced in MERLIN MAGIX Release 2.1 with private data version 3. The service allows an application to obtain the administered configuration parameters for a local DGC Group. It does not provide any information about administered members, trunks, or Delayed Announcement Units (DAUs); this information is available through other escape services.

The service is valid for any local Calling Group on the local system. The service will be denied for a group that has a non-local member.

Service Request Parameters

Table 11-27. cstaEscapeService( ) Parameters for mlQueryDGCGroupParameters ( )

acsHandle	ACS stream on which service request being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized with ***mlQueryDGCGroupParameters( )***

Private Service Request Parameters

Table 11-28. mlQueryDGCGroupParameters( ) Private Service Request Parameters

dgcID	Calling Group number of the group being queried

Return Values

Table 11-29. cstaEscapeService( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch has accepted the request.

Table 11-30. CSTAEscapeServiceConfEvent Parameters for mlQueryDGCGroupParameters( )

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_ESCAPE_SVC_CONF`
invokeID	identifies service request within stream
privateData	private data buffer containing an `ML_QUERYDGC_GROUP_PARAMETERS_CONF` event

Confirmation Event Private Data

Table 11-31. mlQueryDGCGroupParameters( ) Confirmation Event Private Data Parameters

eventType	ML_QUERYDGC_GROUP_PARAMETERS_CONF
groupType	group type: Auto Login – ML_GT_AUTO_IN Auto Logout – ML_GT_AUTO_OUT Integrated VMI – ML_GT_INTEG_VMI Generic VMI – ML_GT_GENERIC_VMI
huntType	hunt type: Circular – ML_HT_CIRCULAR Linear – ML_HT_LINEAR Most Idle – ML_HT_MOST_IDLE_AGENT
msgWaitingExt	extension number where messages are left for the DGC group
externalAlertExt	external alert station assigned to the group
supportGroup	DGC Group ID of the support group assigned to the specified group
overflowDest	DGC Group ID or QCC LDN of the overflow destination assigned for the specified group
priority	a number (1-32) indicating the priority level of calls being routed to DGC overflow groups
queueControlLimit	number of calls allowed in the DGC group's queue
alarmThreshold1	first of three Calls-in-Queue alarm thresholds at which alarms are displayed at supervisor stations monitoring DGC group
alarmThreshold2	second of three Calls-in-Queue alarm thresholds at which alarms are displayed at supervisor stations monitoring DGC group
alarmThreshold3	third of three Calls-in-Queue alarm thresholds at which alarms are displayed at supervisor stations monitoring DGC group
groupCoverage	flag (TRUE or FALSE) indicating whether the DGC group is a receiver for at least one coverage group

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to an ***mlQueryDGCGroupParameters( )*** request, the ***CSTAUniversalFailure-ConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_UNSPECIFIED` when the request could not be satisfied for a reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` - ***dgcID*** is not a valid Calling Group number.

`INVALID_OBJECT_TYPE` - ***dgcID*** is not a local Calling Group number (i.e., the Calling Group contains a networked member).

`INVALID_FEATURE` - The application requested the escape service on a stream opened with a private data version less than version 3, or on a stream opened without private data.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` - Processing the ***mlQueryDGCGroupParameters( )*** request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlQueryDGCGroupParameters (MLPrivateData_t      *privateData,
              DeviceID_t                        *dgcID);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
              InvokeID_t            invokeID,         /* INPUT */
              PrivateData_t         *privateData)     /* INPUT */;
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct MLEvent_t
{
   MLEventType_t  eventType;
   union
   {
      MLQueryDGCGroupParametersConfEvent_t
                     queryDGCGroupParameters;
   } u;
} MLEvent_t;

typedef enum MLDGCGroupType_t
{
   ML_GT_UNKNOWN = 0,
   ML_GT_AUTO_OUT = 1,
   ML_GT_AUTO_IN = 2,
   ML_GT_INTEG_VMI = 3,
   ML_GT_GENERIC_VMI = 4
} MLDGCGroupType_t;
```

```
typedef enum MLDGCHuntType_t
{
   ML_HT_UNKNOWN = 0,
   ML_HT_CIRCULAR = 1,
   ML_HT_LINEAR = 2,
   ML_HT_MOST_IDLE_AGENT = 3,
} MLDGCHuntType_t;

typedef struct MLQueryDGCGroupParametersConfEvent_t
{
   MLDGCGroupType_t     groupType;
   MLDGCHuntType_t      huntType;
   DeviceID_t           msgWaitingExt;
   DeviceID_t           externalAlertExt;
   DeviceID_t           supportGroup;
   DeviceID_t           overflowDest;
   short                priority;
   short                queueControlLimit;
   short                alarmThreshold1;
   short                alarmThreshold2;
   short                alarmThreshold3;
   Boolean     groupCoverage;
} MLQueryDGCGroupParametersConfEvent_t;
```

Important Feature Interactions

Networking

If an application requests the ***mlQueryDGCGroupParameters( )*** escape service for a Calling Group containing a non-local member, the service request is denied.

mlQueryDGCQueueStatus( )

The ***mlQueryDGCQueueStatus( )*** service is introduced in MERLIN MAGIX Release 2.0 with private data version 2. The service returns the number of calls in a DGC queue. The service is valid for any local Calling Group. This service is denied for a Calling Group that has non-local members in it.

Service Request Parameters

Table 11-32. cstaEscapeService( ) Parameters for mlQueryDGCQueueStatus( )

acsHandle	ACS stream on which service request being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized with ***mlQueryDGCQueueStatus( )***

Private Service Request Parameters

Table 11-33. mlQueryDGCQueueStatus( ) Private Service Request Parameters

dgcID	Calling Group number of the group being queried

Return Values

Table 11-34. cstaEscapeService( ) Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch has accepted the request.

Table 11-35. CSTAEscapeServiceConfEvent Parameters for mlQueryDGCQueueStatus( )

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_ESCAPE_SVC_CONF`
invokeID	identifies service request within stream
privateData	private data buffer containing an `ML_QUERYDGC_QUEUE_STATUS_CONF` event

Confirmation Event Private Data

Table 11-36. mlQueryDGCQueueStatus( ) Confirmation Event Private Data Parameters

eventType	`ML_QUERYDGC_QUEUE_STATUS_CONF`
callsInQueue	number of calls in the DGC queue

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to an ***mlQueryDGCQueueStatus( )*** request, the ***CSTAUniversalFailure-ConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_UNSPECIFIED` when the request could not be satisfied for a reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` - ***dgcID*** is not a valid Calling Group number.

`INVALID_OBJECT_TYPE` - ***dgcID*** is not a local Calling Group number (i.e., the Calling Group contains a networked member).

`INVALID_FEATURE` - The application requested the escape service on a stream opened with private data version 1 stream, or on a stream opened without private data.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` - Processing the ***mlQueryDGCQueueStatus( )*** request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlQueryDGCQueueStatus(MLPrivateData_t     *privateData,
              DeviceID_t                  *dgcID);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
              InvokeID_t            invokeID,         /* INPUT */
              PrivateData_t         *privateData)     /* INPUT */;
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct MLEvent_t
{
   MLEventType_t  eventType;
   union
   {
      MLQueryDGCQueueStatusConfEvent_t    queryDGCQueueStatus;
   } u;
} MLEvent_t;

typedef struct MLQueryDGCQueueStatusConfEvent_t
{
   short    callsInQueue;
} MLQueryDGCQueueStatusConfEvent_t;
```

Important Feature Interactions

Group Calling (DGC)

The ***mlQueryDGCQueueStatus( )*** service request returns the number of calls in the queue. If there are no calls in the Calling Group queue, the service request is successful, but indicates that there are zero calls in the queue.

Networking

If an application requests the ***mlQueryDGCQueueStatus( )*** escape service for a Calling Group containing a non-local member, the service request is denied.

mlQueryDeviceName( )

The ***mlQueryDeviceName( )*** service is introduced in MERLIN MAGIX Release 2.0 with private data version 2. The service returns the switch administered label for a Line, Trunk, Extension or Calling Group.

Service Request Parameters

Table 11-37. cstaEscapeService( ) Parameters for mlQueryDeviceName( )

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized by calling ***mlQueryDeviceName( )***

Private Service Request Parameters

Table 11-38. mlQueryDeviceName( ) Private Service Request Parameters

device	device (line, trunk, extension, or calling group) being queried

Return Values

Table 11-39. cstaEscapeService( ) Return Values for mlQueryDeviceName( )

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch was able to process the request.

Table 11-40. CSTAEscapeServiceConfEvent Parameters

acsHandle	handle for stream (from service request)
eventClass	`CSTACONFIRMATION`
eventType	`CSTA_ESCAPE_SVC_CONF`
invokeID	identifies service request within stream
privateData	private data buffer containing an `ML_QUERY_DEVICE_NAME_CONF` event

Confirmation Event Private Data

The ***CSTAEscapeServiceConfEvent*** will contain MERLIN MAGIX private data.

Table 11-41. mlQueryDeviceName( ) Confirmation Event Private Data Parameters

eventType	`ML_QUERY_DEVICE_NAME_CONF`

deviceType	device type:
	`ML_DT_TRUNK` - Line/Trunk
	`ML_DT_STATION - Extension`
	`ML_DT_DGC_QUEUE - DGC Queue`

device	device (from service request)
name[ ]	a null-terminated string containing the administered label for ***device***

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***mlQueryDeviceName( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

GENERIC_UNSPECIFIED – An application will receive GENERIC_UNSPECIFIED when the request could not be satisfied for a reason other than the more specific reasons given below.

RESOURCE_OUT_OF_SERVICE – The CTI link is disconnected or not in service.

INVALD_CSTA_DEVICE_IDENTIFIER – ***device*** is not a valid trunk identifier, extension number or DGC Group identifier on the MERLIN MAGIX system.

INVALID_FEATURE – The application requested the escape service on a stream opened with private data version 1 stream, or on a stream opened without private data.

OUTSTANDING_REQUEST_LIMIT_EXCEEDED – Processing the ***mlQueryDeviceName( )*** request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

REQUEST_TIMEOUT_REJECTION – The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

RESOURCE_LIMITATION_REJECTION – A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlQueryDeviceName(MLPrivateData_t      *privateData,
                  DeviceID_t           *deviceID);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
                   InvokeID_t       invokeID,         /* INPUT */
                   PrivateData_t    *privateData)     /* INPUT */;
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct MLEvent_t
{
   MLEventType_t  eventType;
   union
   {
      MLQueryDeviceNameConfEvent_t     queryDeviceName;
   } u;
} MLEvent_t;

typedef enum MLDeviceType_t
{
   ML_DT_DGC_QUEUE = 1,
   ML_DT_STATION = 5,
   ML_DT_TRUNK = 6
} MLDeviceType_t;

typedef struct MLQueryDeviceNameConfEvent_t
{
   MLDeviceType_t    deviceType;
   DeviceID_t        device;
   char              name[16];
} MLQueryDeviceNameConfEvent_t;
```

Important Feature Interactions

Busy-Out

The board or port for ***device*** may be busied out without affecting the result of the ***mlQueryDeviceName( )*** service.

Demand Test

When the board for ***device*** is going through a demand test, the result of the ***mlQueryDeviceName( )*** service is not affected.

Direct Facility Termination (DFT)

The ***mlQueryDeviceName( )*** service is allowed for trunks that appear on DFT's . There is no effect on DFT's when the ***mlQueryDeviceName( )*** service is requested.

Group Calling

The ***mlQueryDeviceName( )*** service is available for any Calling Group in the system.

Labels

The ***mlQueryDeviceName( )*** service returns the administered label for ***device***. When no label is administered, the ***mlQueryDeviceName( )*** service returns the empty string ("").

Lines

The ***mlQueryDeviceName( )*** service is allowed for any line. If the line has a call on it, the success or failure of the ***mlQueryDeviceName( )*** service is not affected.

Maintenance Busy Mode

If ***device*** is a station or trunk, it may be in Maintenance Busy Mode without affecting the result of the ***mlQueryDeviceName( )*** service.

Normal/Responding Mode

If ***device*** is a station, the station does not have to be in Normal Responding Mode for the ***mlQueryDeviceName( )*** service to be successful.

Outgoing Calls

Outgoing calls are not affected by the ***mlQueryDeviceName( )*** service.

Page Zones

If ***device*** is a Page Zone, the service is denied.

Park Zones

If ***device*** is a Park Zone, the service is denied.

Pools

If ***device*** is a trunk in a pool, the request will still be granted. If the ***device*** is a pool, the service is denied.

Provisioning

When a 800BRI board is undergoing a provisioning test, the label for a trunk on the board can still be requested and received via the ***mlQueryDeviceName( )*** service.

Slot Reset/Busy-out

device may be on a board that is busied-out through a reset or busy-out operation without affecting the result of the ***mlQueryDeviceName( )*** service.

Station Modes

If ***device*** is a station, the station may be in any of the following modes without affecting the result of the ***mlQueryDeviceName( )*** service.

- Administration
- Maintenance
- Alarm Clock
- Directory/Directory Programming
- Feature
- Inspect
- Menu
- Program
- Test

Trunk Test

If ***device*** is a trunk, the trunk may be tested without affecting the result of the ***mlQueryDeviceName( )*** service.

UDP/Networking

Only local extensions, trunks and Calling Groups are valid for the ***mlQueryDeviceName( )*** service.

mlQueryTrunkStatus( )

The ***mlQueryTrunkStatus( )*** escape service is introduced in MERLIN MAGIX Release 2.0 with private data version 2. The service returns the status of a line or trunk. The service is supported for all trunk types, and is allowed whether or not the trunk is assigned to a pool.

Table 11-42 lists the possible values for the trunk status.

Table 11-42. mlQueryTrunkStatus( ) Trunk Status Values

`ML_TS_BUSY`	The trunk is in use by at least one user.
`ML_TS_MAINT_BUSY`	The trunk is in Maintenance Busy Mode.
`ML_TS_IDLE`	The trunk is not in use and is not in Maintenance Busy Mode.

Service Request Parameters

Table 11-43. cstaEscapeService( ) Parameters for mlQueryTrunkStatus( )

acsHandle	ACS stream on which service request is being made
invokeID	identifies this service request within the stream
privateData	private data buffer initialized with ***mlQueryTrunkStatus( )***

Private Service Request Parameters

Table 11-44. mlQueryTrunkStatus( ) Private Service Request Parameters

trunkID	trunk identifier of the line or trunk being queried

Return Values

Table 11-45. cstaEscapeService() Return Values

zero or positive value	Success
ACSERR_BADHDL	***acsHandle*** is not a valid stream identifier
ACSERR_STREAM_FAILED	***acsHandle*** is not valid. The stream may have been closed or aborted

Confirmation Event - CSTAEscapeServiceConfEvent

The ***CSTAEscapeServiceConfEvent*** indicates that the switch has accepted the request.

Table 11-46. CSTAEscapeServiceConfEvent Parameters for mlQueryTrunkStatus( )

acsHandle	handle for stream (from service request)
eventClass	CSTACONFIRMATION
eventType	CSTA_ESCAPE_SVC_CONF
invokeID	identifies service request within stream
privateData	private data buffer containing an ML_QUERY_TRUNK_STATUS_CONF event

Confirmation Event Private Data

Table 11-47. mlQueryTrunkStatus( ) Confirmation Event Private Data Parameters

eventType	ML_QUERY_TRUNK_STATUS_CONF
trunkStatus	status of the line or trunk

CSTA Universal Failure Confirmation Event Errors

When an application receives a ***CSTAUniversalFailureConfEvent*** in response to a ***mlQueryTrunkStatus( )*** request, the ***CSTAUniversalFailureConfEvent*** will contain one of the following values in the ***error*** parameter:

`GENERIC_UNSPECIFIED` - An application will receive `GENERIC_UNSPECIFIED` when the request could not be satisfied for a reason other than the more specific reasons given below.

`RESOURCE_OUT_OF_SERVICE` - The CTI link is disconnected or not in service.

`INVALID_CSTA_DEVICE_IDENTIFIER` - ***trunkID*** is not a valid trunk identifier.

`INVALID_FEATURE` - The application requested the escape service on a stream opened with private data version 1 stream, or on a stream opened without private data.

`OUTSTANDING_REQUEST_LIMIT_EXCEEDED` - Processing the ***mlQueryTrunkStatus( )*** request exceeds the maximum number of outstanding requests permitted at either the driver or the switch.

`REQUEST_TIMEOUT_REJECTION` - The MERLIN MAGIX PBX driver sent the request to the switch, but did not receive a response within the allotted time. This is usually an indication that there is a problem with the CTI link.

`RESOURCE_LIMITATION_REJECTION` - A Telephony Server or MERLIN MAGIX PBX driver resource limitation prevented the system from processing the request.

Request Syntax

```
mlQueryTrunkStatus(MLPrivateData_t     *privateData,
                   DeviceID_t          *trunkID);

typedef struct MLPrivateData_t
{
   char              vendor[32];
   unsigned short    length;
   char              data[ML_MAX_PRIVATE_DATA];
} MLPrivateData_t;

cstaEscapeService (ACSHandle_t      acsHandle,        /* INPUT */
                   InvokeID_t       invokeID,         /* INPUT */
                   PrivateData_t    *privateData)     /* INPUT */;
```

Confirmation Event Syntax

```
typedef struct
{
   ACSHandle_t    acsHandle;
   EventClass_t   eventClass;
   EventType_t    eventType;
} ACSEventHeader_t;

typedef struct
{
   ACSEventHeader_t  eventHeader;
   union
   {
      CSTAConfirmationEvent   cstaConfirmation;
   } event;
} CSTAEvent_t;

typedef struct
{
   InvokeID_t  invokeID;
   union
   {
      CSTAEscapeServiceConfEvent_t  escapeService;
   } u;
} CSTAConfirmationEvent;

typedef struct CSTAEscapeServiceConfEvent_t
{
   Nulltype    null;
} CSTAEscapeServiceConfEvent_t;

typedef struct MLEvent_t
{
   MLEventType_t  eventType;
   union
   {
      MLQueryTrunkStatusConfEvent_t    queryTrunkStatus;
   } u;
} MLEvent_t;

typedef struct MLQueryTrunkStatusConfEvent_t
{
   short    trunkStatus;
} MLQueryTrunkStatusConfEvent_t;

typedef enum MLTrunkStatus_t
{
   ML_TS_MAINT_BUSY = 1,
   ML_TS_BUSY = 2,
   ML_TS_IDLE = 3
} MLTrunkStatus_t;
```

Important Feature Interactions

Auto Maintenance

When the line or trunk being queried has been taken out of service because of this feature, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_MAINT_BUSY`.

Busy-Out

When the line or trunk being queried is on a board that has been busied-out, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_MAINT_BUSY`.

Demand Test

When the line or trunk being queried is on a board that is going through a demand test, the ***mlQueryTrunkStatus( )*** service returns a status `ML_TS_MAINT_BUSY` [1].

Direct Facility Termination (DFT)

The ***mlQueryTrunkStatus( )*** may be used for a line assigned to a DFT.

When the LED next to the DFT is off, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_IDLE`.

E911

When a line is administered as a E911 line, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_IDLE.`

Group Calling (DGC)

The line or trunk being queried may be administered to ring into a Calling Group, either individually or as part of a pool.

Hold

When an outside call is on Hold (any type), the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_BUSY` for the line or trunk associated with the call.

Incoming Calls

When an incoming call is ringing, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_BUSY` for the line or trunk associated with the call.

Lines/Trunks

The ***mlQueryTrunkStatus( )*** service supports all trunk types.

---

[1] A board that is going through a demand test will be busied-out.

Loudspeaker Page

When a line is administered as Loudspeaker Page, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_IDLE` even if the Loudspeaker Page line is in use by a call.

Music-On-Hold

When a line is administered as Music-On-Hold, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_IDLE` even if the Music-On-Hold line is in use by a call.

Networking

The ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_BUSY` for the line associated with a Networked Call (whether incoming or outgoing).

This ***mlQueryTrunkStatus( )*** service is only available for lines and trunks on the local switch.

Outgoing Calls

When an outgoing call is made to the CO (or over the network), the ***mlQueryTrunkStatus( )*** service will return a status of `ML_TS_BUSY` for the line or trunk associated with the call.

Phantom Board

When a line is administered on a phantom board, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_IDLE.`

Pools

The ***trunkID*** may be a line that is in a pool.

If the ***trunkID*** is a pool ID, the ***mlQueryTrunkStatus( )*** service is denied.

Provisioning

When a 800BRI board is undergoing a provisioning test, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_MAINT_BUSY` for lines and trunks on that board.

Ringing Options

The ringing options associated with a DFT have no effect on the result of the ***mlQueryTrunkStatus( )*** service.

Slot Reset/Busy-out

When a board is busied-out through a reset or busy-out operation, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_MAINT_BUSY` for any of the lines or trunks on the board.

Slot Restore

When a board is restored through a Restore operation, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_IDLE` for any of the lines or trunks on the board.

T1 and PRI lines

T1 and PRI lines may be used for voice or data on certain boards (e.g., 100R and 100DCD). When they are used for data, from the switch's point of view, the lines are unequipped and look idle even when there is data going across the line. For any data line that fits this description, the ***mlQueryTrunkStatus( )*** service returns a status of `ML_TS_IDLE` unless the board is in any of the maintenance states listed in this section.

Event Flows

12

Contents

Contents

Contents

Contents

Event Flows

12

This chapter describes various MERLIN LEGEND and MERLIN MAGIX CTI event flows. The flows are organized into the following subsections:

- Service Invocation Event Flows — These flows illustrate the events that flow in response to various service invocations. The service invocation event flows usually occur within a broader context, and they are important building blocks.
- Basic Call Event Flows — These flows illustrate basic extension calling scenarios to and from internal and external destinations.
- Incoming Trunk Event Flows — These flows illustrate incoming trunk calls arriving at a monitored extension. These flows include flows for incoming trunk calls that arrive through Voice Prompting Unit, DGC Group, QCC, and unmonitored DLC[1].
- Consultation Scenarios — These flows illustrate the use of the consultation service (including private data) to extend a call to another user. The receiving user's application may use information about the original caller to pop a screen. Private data lets an application monitoring the receiving extension pop a screen using the original caller's information as soon as the consultation call begins to alert.
- Conference Scenarios — These flows illustrate the use of the conference service.
- Transfer Scenarios — These flows illustrate the use of the transfer service.
- Feature Invocation Scenarios — These flows illustrate the interaction of MERLIN LEGEND and MERLIN MAGIX switch features with TSAPI services and events.
- Shared System Access Scenarios — These flows illustrate the interaction of Shared System Access and similar button types with TSAPI services and events. The beginning of this section describes the MERLIN LEGEND and MERLIN MAGIX switch rules for dealing with such facilities in the context of the TSAPI model.

1 A monitored DLC behaves like any other monitored extension. See Chapter 8 for a discussion of DLC interactions.

Note that headset operation is not involved in these scenarios.

Throughout this chapter, diagrams show the devices, connections, and calls before, during, and after event scenarios. In the diagrams, squares are devices and are labeled D1, D2, etc. (as well as having illustrative extension numbers) Circles are calls and are labeled C1, C2, etc. Lines are connections and their label identifies the device and the call (for example D1C2 would be the connection of device D1 to call C2). Table 12-1 shows the symbols used to label connections with their connection state.

Table 12-1. Symbols Used in Call Control Service Scenario Figures

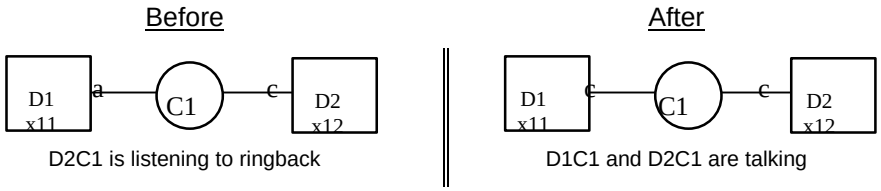
Symbol	Connection State
i	Initiated (the extension is hearing dial tone, is in the process of dialing, or has completed dialing but the call has not yet originated)
a	Alerting (often audible ringing, but not necessarily)
c	Connected
h	Held
ht, hc	Held for Transfer, Held for Conference - These are used when necessary to distinguish these states from Held.
q	Queued
*	Any non-null state
assoc	Always shown with a dotted line, “assoc” indicates that a call appears at the device in a MERLIN LEGEND or MERLIN MAGIX switch associative state.
bridged	Shown with a dotted line, “bridged” indicates that the device has used an SSA button to bridge onto a call.

Service Invocation Event Flows

The event flows in this section show service invocations, service confirmations and call events that flow as a result of service invocations. These flows will typically occur in a stream where other service requests occur and where other call event reporting will occur.

cstaAnswerCall( )

Extension 12 has placed a call to Extension 11 that is now alerting.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 12 has called Extension 11, where the call is now alerting.	***cstaAnswerCall( )*** ***alertingCall*** = D1C1	
	CSTAAnswerCallConfEvent	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 12 has called Extension 11, where the call is now alerting.	***cstaAnswerCall( )*** ***alertingCall*** = D1C1	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL

cstaClearConnection( )

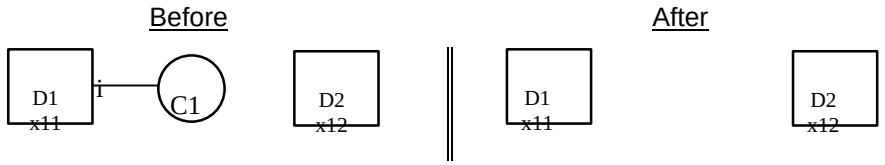
The ***cstaClearConnection( )*** scenarios below show event flows that result in different situations:

- clearing a connection for an initiated call (the call is not connected at the far end);
- clearing a connection for a two-party call;
- clearing the connection for a conference call at the conference originator;
- clearing an extension other than the conference originator from a conference call (at least one remaining party is connected to the conference call);
- clearing an extension other than the conference originator from a conference call and, finding that all remaining parties have the call held, clearing the call.

cstaClearConnection( ) Drops Initiated Call

Extension 11 is placing a call to Extension 12. An application requests ***cstaClearConnection( )*** before that call is delivered to Extension 12. This includes the cases where:

- Extension 11 is hearing dial tone.
- Extension 11 is in the middle of manual dialing.
- Extension 11 is hearing busy tone (the call is not delivered to Extension 12).



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

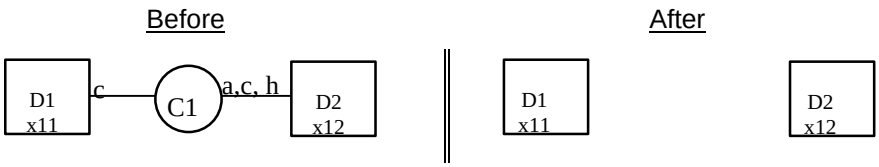
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 is in the midst of dialing a call to Extension 12 or is hearing dial tone.	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 is in the midst of dialing a call to Extension 12 or is hearing dial tone.	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	
If Extension 11 was off-hook on the speakerphone (and not the handset), Extension 11 is now on-hook and idle	***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	

cstaClearConnection( ) Drops Extension from Two-Party Call

Extension 11 is connected to a call that has been delivered to Extension 12. The call may be connected or held at Extension 12. An application requests cstaClearConnection( ) for Extension 11's connection to that call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

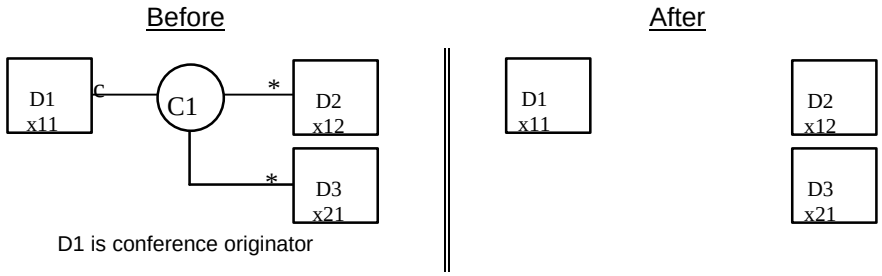
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 is connected to a call that is alerting, connected, or held at Extension 12.	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
Event confirms that the connection has cleared from Extension 11.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
Since this is a two-party call, the connection at Extension 12 also clears.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 is connected to a call that is alerting, connected, or held at Extension 12.	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
Event confirms that the connection has cleared from Extension 11.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
Since this is a two-party call, the connection at Extension 12 also clears.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
If Extension 11 was off-hook on the speakerphone (and not the handset), Extension 11 is now on-hook and idle	***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	

cstaClearConnection( ) Drops Conference Originator from Conference Call

Extension 11 is the conference originator for conference call C1. When the conference originator drops from a conference call, the switch tears down that conference call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Application drops Conference originator from conference call. ***cstaClearConnection( )*** ***call*** = D1C1		
CSTAClearConnectionConfEvent		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
When the conference originator drops from a conference call, the switch tears down the call, so events show other parties being dropped as well.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause***= EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
When the conference originator drops from a conference call, the switch tears down the call, so events show other parties being dropped as well.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

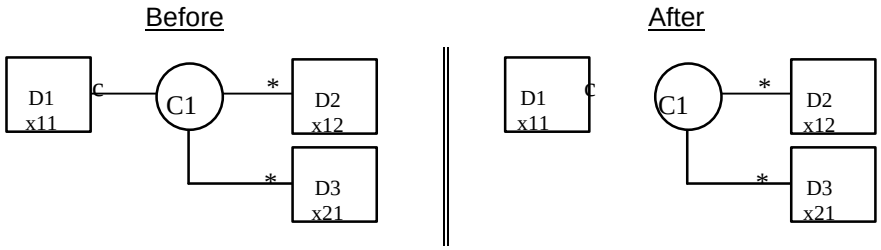
NOTE:
The ordering of the events showing the call clearing from Extensions 12 and 21 depends on the order in which the MERLIN LEGEND or MERLIN MAGIX switch clears the connections. In this example, the switch cleared the connection at Extension 12 first. Thus, the monitors on Extensions 12 and 21 see the connection at Extension 12 clear. Then, the connection at Extension 21 clears with the monitor on Extension 21 receiving that event.

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Application drops Conference originator from conference call. ***cstaClearConnection( )*** ***call*** = D1C1		
CSTAClearConnectionConfEvent		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
When the conference originator drops from a conference call, the switch tears down the call, so events show other parties being dropped as well.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause***= EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
When the conference originator drops from a conference call, the switch tears down the call, so events show other parties being dropped as well.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
If Extension 11 was off-hook on the speakerphone (and not the handset), Extension 11 is now on-hook and idle ***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		

cstaClearConnection( ) Drops Extension (Not Conference Originator) from Conference Call

Extension 11 is not the conference originator, and either Extension 12 or Extension 21 (or both) must be connected to the call. If both Extension 12 and Extension 21 have the call on hold, then the call is torn down (the following scenario shows this event flow).



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

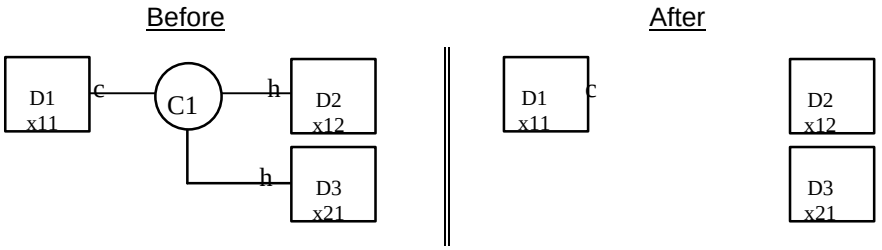
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12 or 21
Application clears conference call connection from Extension 11 (not conference originator).	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
Event confirms that the connection has cleared from the extension set.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12 or 21
Application clears conference call connection from Extension 11 (not conference originator).	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
Event confirms that the connection has cleared from the extension set.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
If Extension 11 was off-hook on the speakerphone (and not the handset), Extension 11 is now on-hook and idle	***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	

cstaClearConnection( ) Drops Extension (Not Conference Originator) from Conference Call and Finding All Parties Held, Clears Call

Extension 11 is not the conference originator, and both Extension 12 and Extension 21 have held the conference call. When an application clears the conference call connection at Extension 11, the resulting call has all parties on hold, so it is torn down.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Application clears conference call connection from Extension 11 (not conference originator) ***cstaClearConnection( )*** ***call*** = D1C1		
CSTAClearConnectionConfEvent		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
When all parties remaining on a call have the call on hold, the switch tears down the call.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
When all parties remaining on a call have the call on hold, the switch tears down the call.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

NOTE:
The ordering of the events showing the call clearing from Extensions 12 and 21 depends on the order in which the MERLIN LEGEND or MERLIN MAGIX switch clears the connections. In this example, the switch cleared the connection at Extension 12 first.

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Application clears conference call connection from Extension 11 (not conference originator) ***cstaClearConnection( )*** ***call*** = D1C1		
CSTAClearConnectionConfEvent		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11		
When all parties remaining on a call have the call on hold, the switch tears down the call.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
When all parties remaining on a call have the call on hold, the switch tears down the call.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
If Extension 11 was off-hook on the speakerphone (and not the handset), Extension 11 is now on-hook and idle	***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	

cstaConferenceCall( )

An application typically uses ***cstaConsultationCall( )*** prior to requesting ***cstaConferenceCall( )***. In addition, there are certain combinations of manual operations that are acceptable prerequisites. Refer to the ***cstaConferenceCall( )*** manual page in Chapter 4 for information on the manual operations.

The ***cstaConferenceCall( )*** scenarios below show event flows that result in different situations:

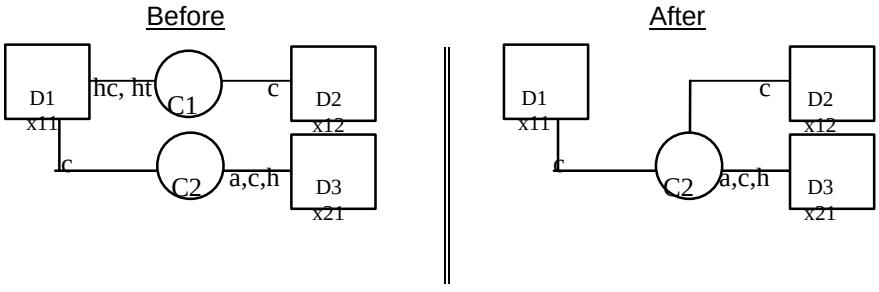
- creating a typical three-party conference call;
- conferencing a held conference call with a two-party call.

The scenario diagrams and flows show a resulting conference call. In a MERLIN LEGEND or MERLIN MAGIX switch environment, the call ID of the resulting call will always be the same as one of the call IDs for one of the calls that was merged into the conference call. However, other switches may allocate a new identifier for the conference call, so switch independent applications should never depend on this MERLIN LEGEND/MERLIN MAGIX switch behavior.

cstaConferenceCall( ) Creates Typical Three-Party Conference

Call activity at Extension 11 (or application activity on behalf of Extension 11) has the connections at Extension 11 in the required states for an application to make a successful ***cstaConferenceCall( )*** request. A variety of scenarios may have brought the connections to this state, including:

- Establishing a call between Extension 11 and Extension 12 (either application or manual action) and the application issuing ***cstaConsultationCall( )*** and making a consultation call to Extension 21.
- Establishing a call between Extension 11 and Extension 12 (either application or manual action) and the user at Extension 11 pressing the CONFERENCE button, and making a call to Extension 21.



Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12, 21
Application conferences consultation call C2 with held call C1 at Extension 11.	***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
	CSTAConferenceCallConfEvent ***newCall*** = D1C2	
	CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C2 12 D2C2 21 D3C2	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C2 12 D2C2 21 D3C2

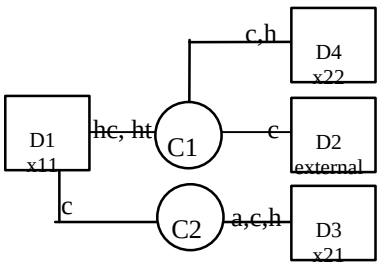
cstaConferenceCall( ) Conferences Held Conference Call with Another Call

Extension 11 has a conference call C1 on hold-for-transfer or hold-for-conference and a connection to call C2, which may be alerting, connected or held at Extension 21. The conference operation will join these two calls.

The notation <*> indicates that the device identifier contains:

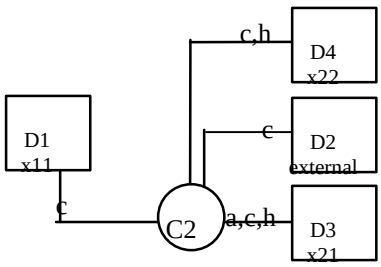
- ANI/ICLID if the connection was an incoming call that arrived on a trunk providing ANI or ICLID
- Dialed Digits if the connection was an outgoing connection
- A trunk device identifier if the connection was an incoming call that arrived on a trunk that does not provide ANI or ICLID

Before



There may also be an additional connection D5C1 in the same initial states as D4C1.

After



If D5C1 was present, there will now be D5C3 in the same final states as D4C2.

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21, 22
Application conferences held conference call C1 with consultation call C2 at Extension 11.	***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
	CSTAConferenceCallConfEvent ***newCall*** = D1C2	
	CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C2 <*> D2C2 21 D3C2 22 D4C2	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C2 <*> D2C2 21 D3C2 22 D4C2

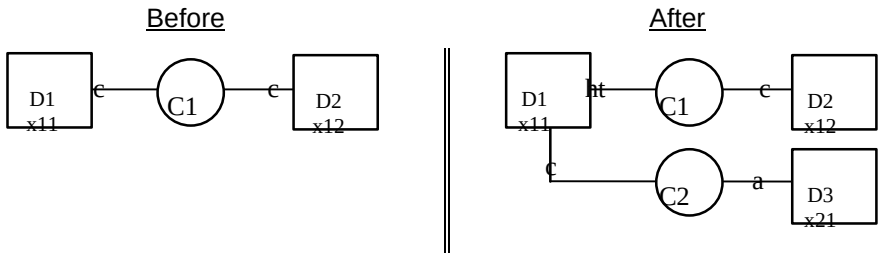
cstaConsultationCall( )

The ***cstaConsultationCall( )*** scenarios below show event flows that result in different situations:

- making a typical consultation call;
- making a consultation call when the connection placed on hold at the consulting station drops during the consultation operation;
- a consultation call attempt results in all parties on the held call being on hold (the held call gets torn down) and the consultation service fails.

cstaConsultationCall( ) Makes Typical Consultation Call

Extension 11 had called Extension 12 and is connected to Extension 12 and an application makes a consultation call from Extension 11 to Extension 21. This places the connection D1C1 on hold and initiates the connection D1C2.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
cstaConsultationCall( ) ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

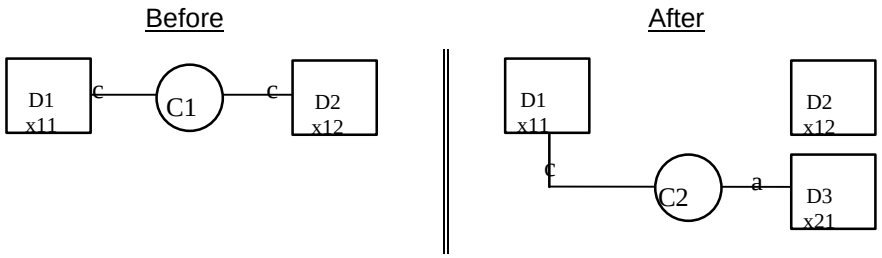
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
cstaConsultationCall( )		
activeCall = D1C1		
calledDevice = 21		
CSTAHeldEvent	***CSTAHeldEvent***	
heldConnection = D1C1	***heldConnection*** = D1C1	
holdingDevice = 11	***holdingDevice*** = 11	
CSTAConsultationCallConfEvent		
newCall = D1C2		
CSTAServiceInitiatedEvent		
initiatedConnection = D1C2		
CSTADeliveredEvent		***CSTADeliveredEvent***
connection = D3C2		***connection*** = D3C2
alertingDevice = 21		***alertingDevice*** = 21
callingDevice = 11		***callingDevice*** = 11
calledDevice = 21		***calledDevice*** = 21
cause = EC_NEW_CALL		***cause*** = EC_NEW_CALL
PrivateData		***PrivateData***
originalCallInfo		***originalCallInfo***
callingDevice = 12		***callingDevice*** = 12
calledDevice = 11		***calledDevice*** = 11

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
cstaConsultationCall( )		
activeCall = D1C1		
calledDevice = 21		
CSTAHeldEvent	***CSTAHeldEvent***	
heldConnection = D1C1	***heldConnection*** = D1C1	
holdingDevice = 11	***holdingDevice*** = 11	
cause = EC_TRANSFER	***cause*** = EC_TRANSFER	
CSTAConsultationCallConfEvent		
newCall = D1C2		
CSTAServiceInitiatedEvent		
initiatedConnection = D1C2		
CSTADeliveredEvent		***CSTADeliveredEvent***
connection = D3C2		***connection*** = D3C2
alertingDevice = 21		***alertingDevice*** = 21
callingDevice = 11		***callingDevice*** = 11
calledDevice = 21		***calledDevice*** = 21
cause = EC_NEW_CALL		***cause*** = EC_NEW_CALL
PrivateData		***PrivateData***
originalCallInfo		***originalCallInfo***
callingDevice = 12		***callingDevice*** = 12
calledDevice = 11		***calledDevice*** = 11

cstaConsultationCall( ) When Party is Placed on Hold and Then Drops During Consultation

Extension 11 and Extension 12 are connected on call C1. An application makes a consultation call from Extension 11 to Extension 21. During the consultation operation, the connection D1C1 is held, the connection D1C2 is initiated, and then Extension 12 drops from call C1 before the consultation call alerts at Extension 21.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

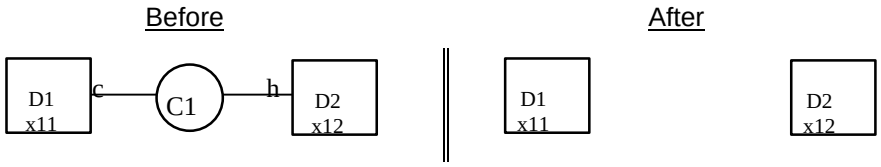
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 is connected to Extension 12 and consults to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21
Device D2 hangs up, causing call C1 to clear.		
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED		
The consultation call C2 is still up.		

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 is connected to Extension 12 and consults to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER (R2.1 only)	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER (R2.1 only)	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
Device D2 hangs up, causing call C1 to clear.		
	CSTAReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED		
The consultation call C2 is still up.		

cstaConsultationCall( ) When Consultation Causes All Parties to be on Hold

Extensions 11 and 12 are on a call. Extension 12 has the call on hold. Extension 11 attempts a consultation, leaving all parties on that call on hold, so the call is cleared and the consultation fails.



Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 is connected to Extension 12 and consults to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
The consultation operation fails because placing the connection on hold results in its being torn down (placed all parties on hold).	***CSTAUniversalFailureConfEvent*** ***error*** = GENERIC_UNSPECIFIED	
Placing D1C1 on hold causes call C1 to be cleared, since all parties are on hold.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
		CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

cstaDeflectCall( )

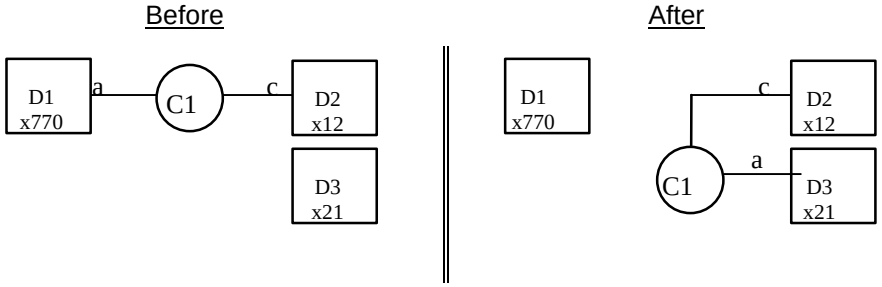
The ***cstaDeflectCall( )*** scenarios below show event flows that result in different situations:

- successfully redirecting a queued call to a station;
- successfully redirecting a Calling Group call that is alerting at a station to a Calling Group queue;
- successfully redirecting a Calling Group call alerting at one station to another station.

The ***cstaDeflectCall( )*** service is available beginning with MERLIN MAGIX Release 2.0.

cstaDeflectCall( ) for Call in Queue to Station – MERLIN MAGIX Release 2.0 and Later

Extension 12 is connected to a call that is queued for Calling Group 770. An application requests the ***cstaDeflectCall( )*** service to redirect the call to Extension 21 (it is a member of Calling Group 771 that is logged in and idle).



MERLIN MAGIX R2.0

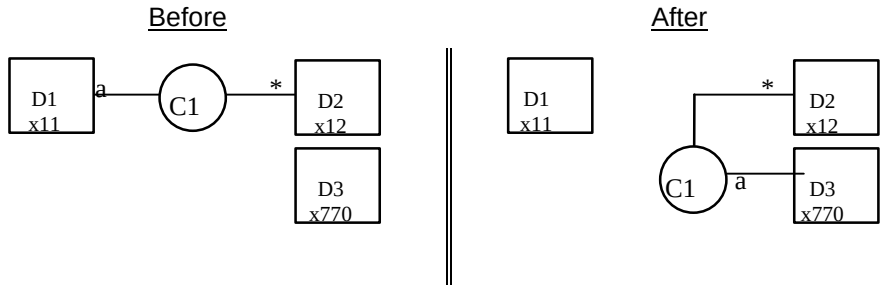
Stream Monitoring Calling Group 770	Stream Monitoring Extension 12	Stream Monitoring Extension 21
cstaDeflectCall( ) ***deflectCall*** = D1C1 ***calledDevice*** = 21		
CSTADeflectCallConfEvent		
Event confirms that the call has been redirected.		
CSTADivertedEvent ***connection*** = D1C1 ***divertingDevice*** =Q770 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D1C1 ***divertingDevice*** =Q770 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D1C1 ***divertingDevice*** =Q770 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
	CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_REDIRECTED ***lastRedirectionDevice***= Q771 ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_REDIRECTED ***lastRedirectionDevice***= Q771 ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770

MERLIN MAGIX R2.1 and later

Stream Monitoring Calling Group 770	Stream Monitoring Extension 12	Stream Monitoring Extension 21
cstaDeflectCall( ) ***deflectCall*** = D1C1 ***calledDevice*** = 21		
CSTADeflectCallConfEvent		
Event confirms that the call has been redirected.		
CSTADivertedEvent ***connection*** = D1C1 ***divertingDevice*** =Q770 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D1C1 ***divertingDevice*** =Q770 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D1C1 ***divertingDevice*** =Q770 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
	CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = Q770 ***cause*** = EC_REDIRECTED ***lastRedirectionDevice*** = ID_NOT_KNOWN	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = Q770 ***cause*** = EC_REDIRECTED ***lastRedirectionDevice*** = ID_NOT_KNOWN

cstaDeflectCall( ) for Station to Calling Group Queue – MERLIN MAGIX Release 2.0 and Later

A call for Calling Group queue 771 is alerting at Extension 11. The ***cstaDeflectCall( )*** service is used to redirect the call to Calling Group queue 770.



MERLIN MAGIX R2.0

Stream Monitoring Calling Group 770	Stream Monitoring Extension 11	Stream Monitoring Extension 12
cstaDeflectCall( ) ***deflectCall*** = D1C1 ***calledDevice*** = 770		
CSTADeflectCallConfEvent		
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED
Event confirms that the call has been redirected. ***CSTAQueuedEvent*** ***connection*** = D3C1 ***queue*** = Q770 ***callingDevice*** =12 ***calledDevice*** = Q770 ***numberQueued*** = 1 ***lastRedirectionDevice*** = ID_NOT_KNOWN ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		***CSTAQueuedEvent*** ***connection*** = D3C1 ***queue*** = Q770 ***callingDevice*** =12 ***calledDevice*** = Q770 ***numberQueued*** = 1 ***lastRedirectionDevice*** = ID_NOT_KNOWN ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11

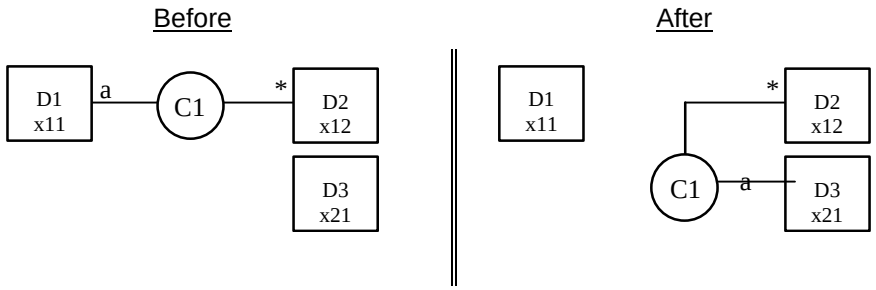
MERLIN MAGIX R2.1 and later

Stream Monitoring Calling Group 770	Stream Monitoring Extension 11	Stream Monitoring Extension 12
cstaDeflectCall( ) ***deflectCall*** = D1C1 ***calledDevice*** = 770		
CSTADeflectCallConfEvent		
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED
Event confirms that the call has been redirected.		
CSTAQueuedEvent ***connection*** = D3C1 ***queue*** = Q770 ***callingDevice*** =12 ***calledDevice*** = Q771 ***numberQueued*** = 1 ***lastRedirectionDevice*** = ID_NOT_KNOWN		***CSTAQueuedEvent*** ***connection*** = D3C1 ***queue*** = Q770 ***callingDevice*** =12 ***calledDevice*** = Q771 ***numberQueued*** = 1 ***lastRedirectionDevice*** = ID_NOT_KNOWN

cstaDeflectCall( ) for Station to Station – MERLIN MAGIX Release 2.0 and Later

A call for Calling Group queue

771 is alerting at Extension 11. The ***cstaDeflectCall( )*** service is used to redirect the call to Extension 21 that is idle and available, but is not a member of any Calling Group.



MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
cstaDeflectCall( ) ***deflectCall*** = D1C1 ***calledDevice*** = 21		
CSTADeflectCallConfEvent		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED
	CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***lastRedirectionDevice*** = ID_NOT_KNOWN ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***lastRedirectionDevice*** = ID_NOT_KNOWN ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
cstaDeflectCall( ) ***deflectCall*** = D1C1 ***calledDevice*** = 21		
CSTADeflectCallConfEvent		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_NOT_ANSWERED
	CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = Q771 ***cause*** = EC_NEW_CALL ***lastRedirectionDevice*** = ID_NOT_KNOWN	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = Q771 ***cause*** = EC_NEW_CALL ***lastRedirectionDevice*** = ID_NOT_KNOWN

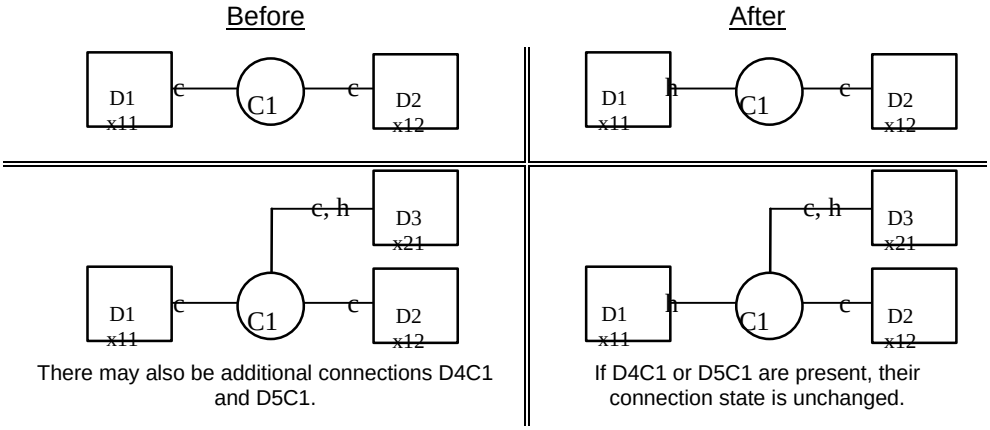
cstaHoldCall( )

The ***cstaHoldCall( )*** scenarios below show event flows that result in different situations:

- successfully placing a two-party or conference call on hold;
- attempting to place a call on hold in a situation that results in all parties on the call being on hold (the call is cleared).

cstaHoldCall( ) Places Call on Hold

The first pair of diagrams below shows a hold scenario in the context of a typical two-party call. The second pair of diagrams shows a conference call. In the case of the conference call, the additional connections (or the internal connections) may also be trunk connections (subject, of course, to the MERLIN LEGEND or MERLIN MAGIX switch limits on the number of internal and external parties that may be connected on a conference call).



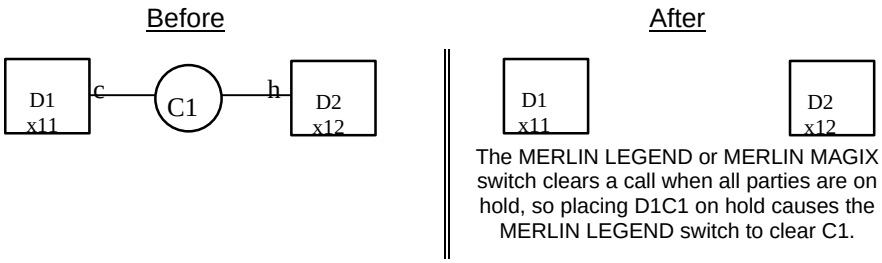
There may also be additional connections D4C1 and D5C1.

If D4C1 or D5C1 are present, their connection state is unchanged.

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12 (or Extension 21)
Extension 11 is connected to Extension 12 (and other extensions if C1 is conference call) and places connection on hold.	***cstaHoldCall( )*** ***activeCall*** = D1C1 ***reservation*** = NO	
	CSTAHoldCallConfEvent	
	CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11

cstaHoldCall( ) Causes Call Clearing When All Parties On Hold

Extension 11 is connected on call C1 with Extension 12. Extension 12 has placed C1 on hold. When an application requests that D1C1 be held, this results in all parties having call C1 held and the call is cleared, even though the ***cstaHoldCall( )*** request was successful.



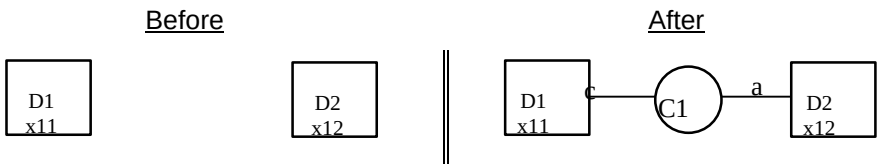
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 is connected to Extension 12 and places connection on hold.	***cstaHoldCall( )*** ***activeCall*** = D1C1 ***reservation*** = TRUE	
Hold request is successful.	***CSTAHoldCallConfEvent***	
All parties on hold on the call result in the call being torn down.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
		CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

NOTE:
The exact flow of the ***CSTAConnectionClearedEvents*** will depend on the order in which the switch tears down the connections comprising call C1.

cstaMakeCall( )

cstaMakeCall to Local Extension

An application requests that a call be made from Extension 11 to Extension 12.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

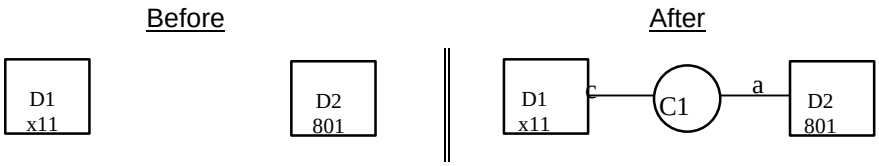
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 makes a call to Extension 12.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 12	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
	CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 makes a call to Extension 12.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 12	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
	CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL

cstaMakeCall to External Number

An application requests that a call be made from Extension 11 to an external number, 555-1234. The call leaves the switch on trunk 801. Trunk 801 is a PRI facility.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11
Extension 11 makes a call to D2. D2 is an external number.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 95551234#
	CSTAMakeCallConfEvent ***newCall*** = D1C1
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1
	CSTANetworkReachedEvent ***connection*** = D2C1 ***trunkUsed*** = T801 ***calledDevice*** = 5551234

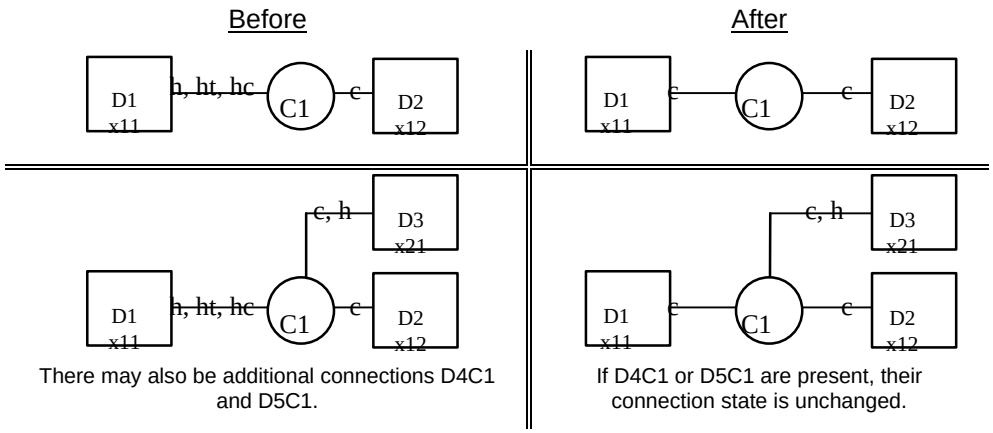
MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11
Extension 11 makes a call to D2. D2 is an external number.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 95551234#
	CSTAMakeCallConfEvent ***newCall*** = D1C1
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1
	CSTANetworkReachedEvent ***connection*** = D2C1 ***trunkUsed*** = T801 ***calledDevice*** = 5551234
Because trunk 801 is a PRI facility, the MERLIN MAGIX switch provides the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** when the call alerts and is subsequently answered at the far end.	
Far end gives an indication that call is alerting. Note that the calledDevice will be the called number, which may or may not match the alerting device.	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 5551234 ***callingDevice*** = 11 ***calledDevice*** = 5551234 ***cause*** = NONE ***Private Data*** ***trunkUsed*** = T801
Far end gives an indication that call is answered. Note that the calledDevice will be the called number, which may or may not match the answering device.	***CSTAEstablishedEvent*** ***connection*** = D2C1 ***answeringDevice*** = 5551234 ***callingDevice*** = 11 ***calledDevice*** = 5551234 ***cause*** = NONE ***Private Data*** ***trunkUsed*** = T801
Extension 11 hangs up.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE

cstaRetrieveCall( )

The first pair of diagrams below shows a retrieve scenario in the context of a typical two-party call. The second pair of diagrams shows a conference call. In the case of the conference call, the additional connections (or the internal connections) may also be trunk connections (subject, of course, to the MERLIN LEGEND and MERLIN MAGIX switch limits on the number of internal and external parties that may be connected on a conference call).

NOTE:
An application may retrieve a call on normal hold, hold-for-transfer, or hold-for-conference. An attempt to retrieve a call on associative hold is denied.

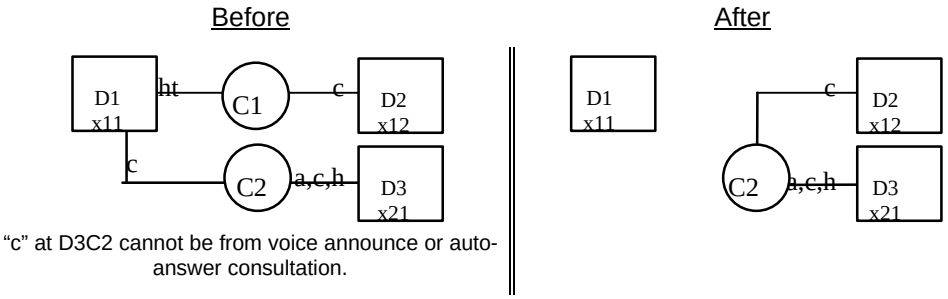


ctivity	Stream Monitoring Extension 11	Stream Monitoring Extension 12 or 21
Extension 11 is connected to Extension 12 (and other extensions if C1 is conference call) and retrieves held connection.	***cstaRetrieveCall( )*** ***heldCall*** = D1C1	
	CSTARetrieveCallConfEvent	
	CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

cstaTransferCall( )

An application typically uses ***cstaConsultationCall( )*** prior to requesting ***cstaTransferCall( )***. In addition, there are certain combinations of manual operations that are acceptable prerequisites. Refer to the ***cstaTransferCall( )*** manual page in Chapter 4 for information on the manual operations.

Typical cstaTransferCall( )



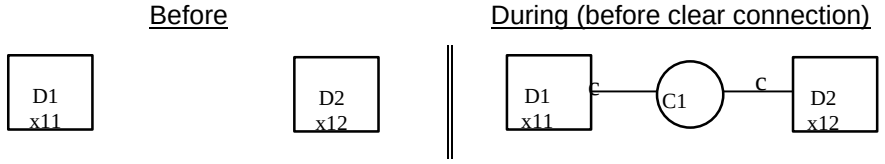
"c" at D3C2 cannot be from voice announce or auto-answer consultation.

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12, 21
	cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2	
	CSTATransferCallConfEvent ***newCall*** = D1C2	
	CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device after 12 D2C2 21 D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device after 12 D2C2 21 D3C2

Basic Extension Calling Event Flows

User Manually Calls Local Extension

A user at Extension 11 makes a call to Extension 12.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

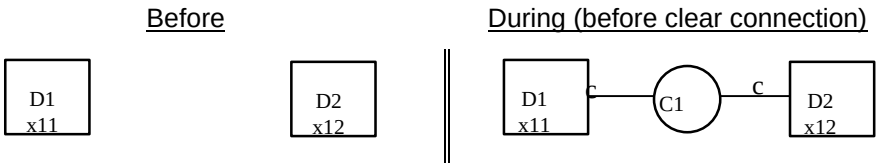
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 goes off-hook.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1	
User at Extension 11 has completed dialing Extension 12 and switch originated call.	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
		cstaAnswerCall( ) ***alertingCall*** = D2C1
		CSTAAnswerCallConfEvent
	CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
Conversation Occurs - The “During” illustration in the figure above applies at this point.		
Application drops Extension 11.	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED
		CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 goes off-hook.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
User at Extension 11 has completed dialing Extension 12 and the switch originated the call.	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
		cstaAnswerCall( ) ***alertingCall*** = D2C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12
	CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
Conversation Occurs - The “During” illustration in the figure above applies at this point.		
Application drops Extension 11.	***cstaClearConnection( )*** ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = = EC_CALL_CANCELLED
		CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
	CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	

cstaMakeCall( ) to Local Extension

An application monitoring Extension 11 uses ***cstaMakeCall( )*** to make a call to Extension 12.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

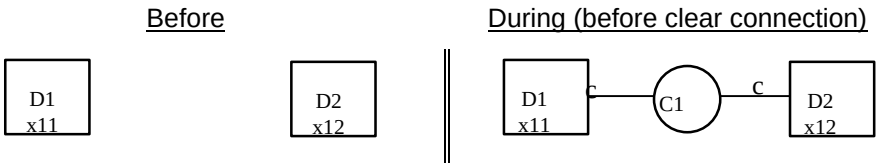
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Application with stream monitoring Extension 11 makes call to Extension 12.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 12	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
	CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
Application answers call at Extension 12.		***cstaAnswerCall( )*** ***alertingCall*** = D2C1
		CSTAAnswerCallConfEvent
	CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
Conversation Occurs - The “During” illustration in the figure above applies at this point.		
		cstaClearConnection( ) ***call*** = D2C1
		CSTAClearConnectionConfEvent
Extension 12 drops first.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Application with stream monitoring Extension 11 makes call to Extension 12.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 12	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
	CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
Application answers call at Extension 12.		***cstaAnswerCall( )*** ***alertingCall*** = D2C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12
	CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
Conversation Occurs - The “During” illustration in the figure above applies at this point.		
		cstaClearConnection( ) ***call*** = D2C1
		CSTAClearConnectionConfEvent
Extension 12 drops first.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	
		CSTAReadyEvent ***agentDevice*** = 12 ***agentID*** = 12

cstaMakeCall( ) Completes Partial Dialing

User at Extension 11 begins dialing Extension 12 manually and then completes dialing using ***cstaMakeCall( )***.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

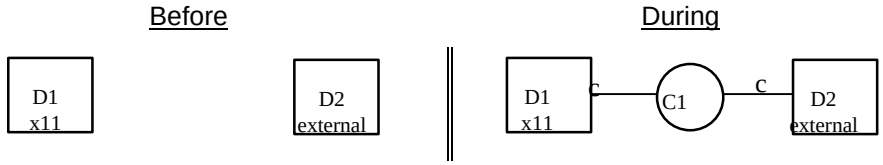
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 goes off-hook.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1	
User at Extension 11 dials digit "1".		
Remaining digit in service request.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 2	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
User at Extension 12 manually answers.	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
Conversation Occurs - The "During" illustration in the figure above applies at this point.		
Application clears connection D2C1.		***cstaClearConnection( )*** ***call*** = D2C1
		CSTAClearConnectionConfEvent
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
This causes switch to clear the remaining connection.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 goes off-hook.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
User at Extension 11 dials digit "1".		
Remaining digit in service request.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 2	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
User at Extension 12 manually answers.		***CSTANotReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12
	CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
Conversation Occurs - The "During" illustration in the figure above applies at this point.		
Application clears connection D2C1.		***cstaClearConnection( )*** ***call*** = D2C1
		CSTAClearConnectionConfEvent
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
This causes switch to clear the remaining connection.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	
		CSTAReadyEvent ***agentDevice*** = 12 ***agentID*** = 12

cstaMakeCall( ) to External Number

An application monitoring Extension 11 uses ***cstaMakeCall( )*** to make call to an external number.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

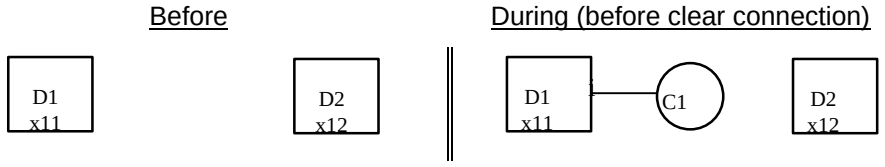
Activity	Stream Monitoring Extension 11
Application with stream monitoring Extension 11 makes call to external number 5551234. Note inclusion of ARS code (9). If a Pool Access Code were used instead, it would be included.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 95551234#
	CSTAMakeCallConfEvent ***newCall*** = D1C1
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1
The ***calledDevice*** in the Network Reached event does not contain the ARS digits or Pool Access Codes.	***CSTANetworkReachedEvent*** ***connection*** = D1C1 ***trunkUsed*** = T801 ***calledDevice*** = 5551234
Conversation Occurs - The “During” illustration in the figure above applies at this point.	
External party drops, causing switch to drop call.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11
Application with stream monitoring Extension 11 makes call to external number 5551234. Note inclusion of ARS code (9). If a Pool Access Code were used instead, it would be included.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 95551234#
	CSTAMakeCallConfEvent ***newCall*** = D1C1
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1
The ***calledDevice*** in the Network Reached event does not contain the ARS digits or Pool Access Codes.	***CSTANetworkReachedEvent*** ***connection*** = D1C1 ***trunkUsed*** = T801 ***calledDevice*** = 5551234
Conversation Occurs - The “During” illustration in the figure above applies at this point.	
External party drops, causing switch to drop call.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED

cstaMakeCall( ) to Invalid or Busy Number

An application monitoring Extension 11 makes a call to Extension 12. Extension 12 does not have an available SA and there is no alternate call treatment, so the caller hears busy tone. In the case of a call to an invalid number, the caller would hear reorder.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

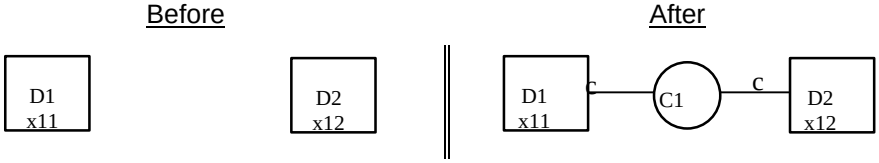
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Application with stream monitoring Extension 11 makes call to Extension 12.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 12	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
Extension 12 does not have an SA and there is no alternate treatment, so calling user hears audible busy tone (call to invalid number would hear reorder). Note that there is no delivered event. The “During” illustration in the figure above applies at this point.		
	cstaClearConnection( ) ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Application with stream monitoring Extension 11 makes call to Extension 12.	***cstaMakeCall( )*** ***callingDevice*** = 11 ***calledDevice*** = 12	
	CSTAMakeCallConfEvent ***newCall*** = D1C1	
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
Extension 12 does not have an SA and there is no alternate treatment, so calling user hears audible busy tone (call to invalid number would hear reorder). Note that there is no delivered event. The “During” illustration in the figure above applies at this point.		
	cstaClearConnection( ) ***call*** = D1C1	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	

Internal Call to DGC Group Arrives at Extension

A call from Extension 12 enters DGC Group 770, and then arrives at an SA button on Extension 11.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 12 makes call to DGC Group 770.		***cstaMakeCall( )*** ***callingDevice*** = 12 ***calledDevice*** = 770
		CSTAMakeCallConfEvent ***newCall*** = D2C1
		CSTAServiceInitiatedEvent ***initiatedConnection*** = D2C1
User at Extension 11 signs in to DGC Group and becomes an available member.		
Call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE
User at Extension 11 manually answers.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 12 makes call to DGC Group 770.		***cstaMakeCall( )*** ***callingDevice*** = 12 ***calledDevice*** = 770
		CSTAMakeCallConfEvent ***newCall*** = D2C1
		CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12
		CSTAServiceInitiatedEvent ***initiatedConnection*** = D2C1
The call is queued because no agents are available.		***CSTAQueuedEvent*** ***connection*** = D3C1 ***queue*** = Q770 ***callingDevice*** =12 ***calledDevice*** = Q770 ***numberQueued*** = 1
User at Extension 11 signs in to DGC Group and becomes an available member.	***CSTALoggedOnEvent*** ***agentDevice*** = 11 ***agentID*** = 11 ***agentGroup*** = Q770	
The call is redirected from the Calling Group queue to Extension 11.	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED
Call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770
User at Extension 11 manually answers.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***PrivateData*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770

MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 12 makes call to DGC Group 770.		***cstaMakeCall( )*** ***callingDevice*** = 12 ***calledDevice*** = 770
		CSTAMakeCallConfEvent ***newCall*** = D2C1
		CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12
		CSTAServiceInitiatedEvent ***initiatedConnection*** = D2C1
The call is queued because no agents are available.		***CSTAQueuedEvent*** ***connection*** = D3C1 ***queue*** = Q770 ***callingDevice*** =12 ***calledDevice*** = Q770 ***numberQueued*** = 1
User at Extension 11 signs in to DGC Group and becomes an available member.	***CSTALoggedOnEvent*** ***agentDevice*** = 11 ***agentID*** = 11 ***agentGroup*** = Q770	
The call is redirected from the Calling Group queue to Extension 11.	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED
Call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = Q771 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = Q771 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED
User at Extension 11 manually answers.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = Q771 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = Q771 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED

Incoming Trunk-to-Extension Calling

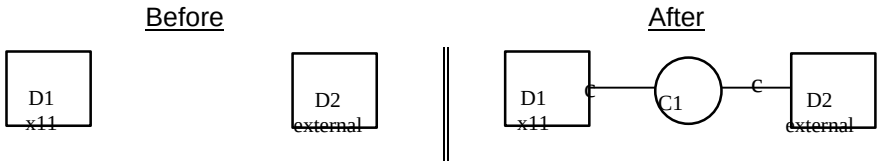
Trunk Call Arrives at Extension

Incoming trunk call arrives at SA button on Extension 11. Note that beginning in MERLIN MAGIX 2.0, the call may arrive on a SA or DFT/DPT button.

The notation <DNIS/EXT> indicates that the parameter contains the DNIS if the call arrived on a facility that provides DNIS (PRI Called Number); otherwise, the parameter contains the extension number.

The notation <ANI/ICLID/UNK> indicates that the parameter contains the ANI if the call arrived on PRI or BRI, ICLID if the call arrived on a facility that provides ICLID, and it has a deviceIDStatus of ID_NOT_KNOWN for all other conditions.

The notation <trunk number> indicates that the parameter contains the trunk dial plan number that is associated with the call. The parameter will be filled in with "Txxxx". The "xxxx" indicates the dial plan id of the trunk.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11
Trunk call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE
User at Extension 11 manually answers.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE
User at Extension 11 manually hangs up.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11
Trunk call arrives at SA, DFT, or DPT button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk number>
User at Extension 11 manually answers.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk number>
User at Extension 11 manually hangs up.	***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE

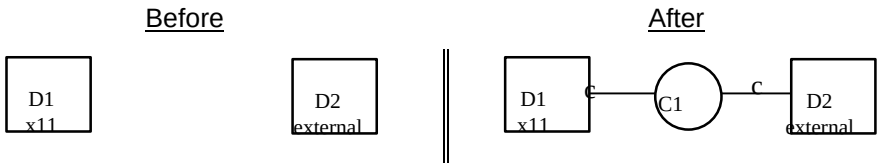
MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring Extension 11
Trunk call arrives at SA, DFT, or DPT button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/UNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk number>
User at Extension 11 manually answers.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/UNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk number>
User at Extension 11 manually hangs up.	***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE

Trunk Call Arrives Through DGC Group

Incoming trunk call arrives at SA button on Extension 11 through Calling Group 770.

The notation <ANI/ICLID/UNK> indicates that the parameter contains the ANI if the call arrived on PRI or BRI, ICLID (with ICLID delay enabled) if the call arrived on a facility that provides ICLID, and it contains “unknown” for all other conditions.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0

Activity	Stream Monitoring Extension 11
Call arrives for Calling Group 770 and no group member is available. Call may hear announcement, if administered.	
User at Extension 11 signs in to DGC Group and becomes an available member.	
Trunk call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***cause*** = EC_NONE
Application answers alerting call at Extension 11.	***cstaAnswerCall( )*** ***alertingCall*** = D1C1
	CSTAAnswerCallConfEvent
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Activity	Stream Monitoring DGC Group 770	Stream Monitoring Extension 11
	CSTAQueuedEvent ***queuedConnection*** = D3C1 ***queue*** = Q770 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = Q770 ***cause*** = EC_NONE ***numberQueued*** = 1 ***Private Data*** ***trunkUsed*** = <trunk number>	
Call arrives for Calling Group 770 and no group member is available. Call may hear announcement, if administered		
User at Extension 11 signs in to DGC Group and becomes an available member.		***CSTALoggedOnEvent*** ***agentDevice*** = 11 ***agentID*** = 11 ***agentGroup*** = Q770
The call is redirected from the Calling Group queue to Extension 11.	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED
Trunk call arrives at SA button on Extension 11.		***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770 ***trunkUsed*** = <trunk number>
Application answers alerting call at Extension 11.		***cstaAnswerCall( )*** ***alertingCall*** = D1C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11
User at Extension 11 manually answers.		***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = Q770 ***trunkUsed*** = <trunk number>

MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring DGC Group 770	Stream Monitoring Extension 11
	CSTAQueuedEvent ***queuedConnection*** = D3C1 ***queue*** = Q770 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = Q770 ***cause*** = EC_NONE ***numberQueued*** = 1 ***Private Data*** ***trunkUsed*** = <trunk number>	
Call arrives for Calling Group 770 and no group member is available. Call may hear announcement, if administered		
User at Extension 11 signs in to DGC Group and becomes an available member.		***CSTALoggedOnEvent*** ***agentDevice*** = 11 ***agentID*** = 11 ***agentGroup*** = Q770
The call is redirected from the Calling Group queue to Extension 11.	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = Q770 ***newDestination*** = 11 ***cause*** = EC_REDIRECTED
Trunk call arrives at SA button on Extension 11.		***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = Q770 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED
Application answers alerting call at Extension 11.		***cstaAnswerCall( )*** ***alertingCall*** = D1C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11
User at Extension 11 manually answers.		***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = Q770 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED

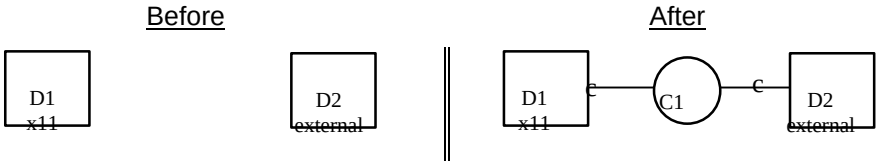
Trunk Call to DGC Group Overflows to DGC Group Then Arrives at Extension

An incoming trunk call arrives at DGC1 (Group Q711), overflows to DGC2 (Group Q770), and then arrives at an SA button on Extension 11 (Extension 11 is a member of DGC2).

An application monitoring Extension 11 will receive the same information about the caller as in the previous event flow (where the call arrives after passing through one DGC Group).

The notation <ANI/ICLID/UNK> indicates that the parameter contains the ANI if the call arrived on PRI or BRI, ICLID (when the ICLID delay is enabled) if the call arrived on a facility that provides ICLID, and it contains “unknown” for all other conditions.

The notation <trunk number> indicates that the parameter contains the trunk dial plan number that is associated with the call. This will be indicated by “Txxxx” where xxxx indicates the dial plan number.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0

Activity	Stream Monitoring Extension 11
Call arrives for DGC1 and no group member is available. Caller may hear announcement, if administered. Then call overflows into DGC2.	
User at Extension 11 signs in to DGC Group 2 and becomes an available member.	
Trunk call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***cause*** = EC_NONE
User at Extension 11 manually answers.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11
Call arrives for DGC1 and no group member is available. Caller may hear announcement, if administered. Then call overflows into DGC2.	
User at Extension 11 signs in to DGC Group and becomes an available member.	***CSTALoggedOnEvent*** ***agentDevice*** = 11 ***agentID*** = 11 ***agentGroup*** = Q770
The call is redirected from the Calling Group queue to DGC1	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = <DGC1> ***newDestination*** = 11 ***cause*** = EC_OVERFLOW
Trunk call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***lastRedirectionDevice*** = <DGC2> ***cause*** = EC_REDIRECTED ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = <DGC1> ***trunkUsed*** = <trunk number>
Application answers alerting call at Extension 11.	***cstaAnswerCall( )*** ***alertingCall*** = D1C1
	CSTAAnswerCallConfEvent
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 11 ***lastRedirectionDevice*** = <DGC2> ***cause*** = EC_REDIRECTED ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = <DGC1> ***trunkUsed*** = <trunk number>

MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring Extension 11
Call arrives for DGC1 and no group member is available. Caller may hear announcement, if administered. Then call overflows into DGC2.	
User at Extension 11 signs in to DGC Group and becomes an available member.	***CSTALoggedOnEvent*** ***agentDevice*** = 11 ***agentID*** = 11 ***agentGroup*** = Q770
The call is redirected from the Calling Group queue to DGC1	***CSTADivertedEvent*** ***connection*** = D3C1 ***divertingDevice*** = <DGC1> ***newDestination*** = 11 ***cause*** = EC_OVERFLOW
Trunk call arrives at SA button on Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = Q771 ***lastRedirectionDevice*** = <DGC2> ***cause*** = EC_REDIRECTED
Application answers alerting call at Extension 11.	***cstaAnswerCall( )*** ***alertingCall*** = D1C1
	CSTAAnswerCallConfEvent
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = Q771 ***lastRedirectionDevice*** = <DGC2> ***cause*** = EC_REDIRECTED

Trunk Call Arrives Through Voice Prompting Unit, QCC, Voice Mail, or Unmonitored DLC

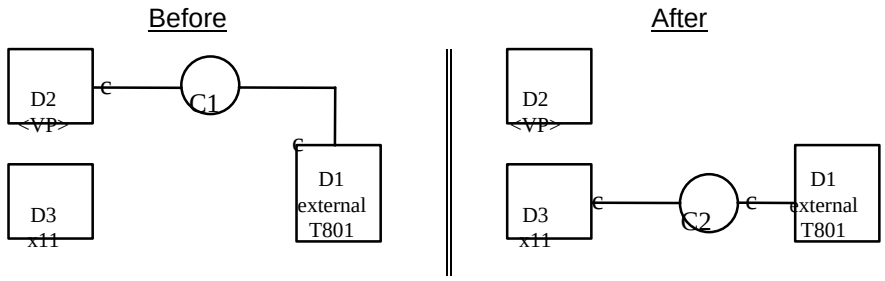
When an incoming trunk call passes through a Voice Prompting Unit QCC, Voice Mail, or an unmonitored DLC (here, D2) to a receiving extension (here, D3, x11), the calling party parameters in the events that flow for a monitor on that extension appear as if the trunk call came directly to that extension. This allows an application to pop a screen using the calling party information as soon as the call alerts (rather than having the voice prompting extension appear there and having to wait for the transfer event). Of course, when the call passes through a Voice Prompting Unit, digits may be collected that also appear in events.

The notation <DNIS/EXT> indicates that the parameter contains the DNIS if the call arrived on a facility that provides DNIS (PRI Called Number); otherwise, the parameter contains the extension number of the alerting or answering device.

The notation <VP> indicates the voice prompting, QCC, Voice Mail, or unmonitored DLC extension.

The notation <ANI/ICLID/UNK> indicates that the parameter contains the ANI if the call arrived on PRI or BRI, ICLID if the call arrived on a facility that provides ICLID, and it contains “unknown” for all other conditions.

The notation <trunk number> indicates that the parameter contains the trunk dial plan number that is associated with the call. The parameter will be filled in with “Txxxx”. The “xxxx” indicates the dial plan id of the trunk.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11
Trunk call arrives and terminates at Voice Prompting Unit.	
Voice Prompting Unit makes call to Extension 11 before transferring incoming trunk call. Delivered event contains collected digit information.	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE ***Private Data*** ***userEnteredCode*** = <collected digits>
User at Extension 11 manually answers.	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE ***Private Data*** ***userEnteredCode*** = <collected digits>

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension VP	Stream Monitoring Extension 11
Voice Prompting Unit makes call to Extension 11 before transferring incoming trunk call. Delivered event contains collected digit information.	***CSTAHeldEvent*** ***heldConnection*** = D2C1 ***holdingDevice*** = VP	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D2C2	
	CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = < EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = < EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>
	CSTATransferredEvent ***primaryOldCall*** = D2C1 ***secondaryOldCall*** = D2C2 ***transferringDevice*** = VP ***transferredDevice*** = 11 ***cause*** = EC_VOICE_UNIT_INITIATOR ***transferredConnections*** device — after <ANI/ICLID/UNK> — D2C2 11 — D3C2	

Consultation Event Flows

Consultation calls have the special property of making original caller information available in private data. Original Caller Information makes it possible for an application monitoring the extension receiving the consultation call to pop a screen using the original caller's information as soon as the consultation call alerts. In manual transfer and conference scenarios, the point at which the same application has access to the original caller's information (and the event containing that information) varies according to the type of transfer (supervised or unsupervised) and other factors. Events provide Original Caller Information only when an application uses ***cstaConsultationCall( )*** to make the consultation call; events do not contain the information when manual operations make the consultation call (as later sections show).

The notation <ANI/ICLID/UNK> indicates that the parameter contains the ANI if the call arrived on BRI or PRI, ICLID if the call arrived on a facility that provides ICLID, and it contains "unknown" for all other conditions.

The notation <ANI/ICLID/TRK> indicates that the parameter contains the ANI if the call arrived on BRI or PRI, ICLID if the call arrived on a facility that provides ICLID, and it contains the trunk number for all other conditions.

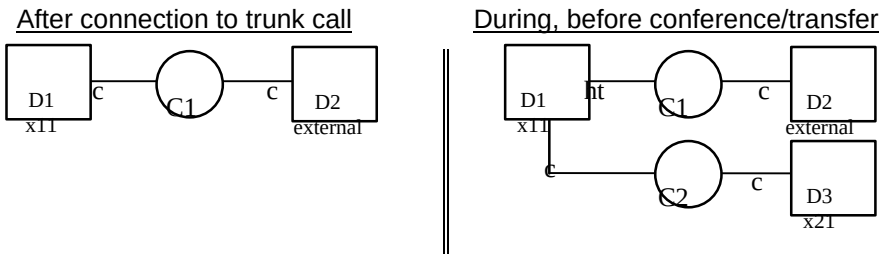
An application may transfer or conference the original call with the consultation call. The event flows in this section show a transfer. A conference would be similar, with conference services, confirmations, and events taking the place of the transfer operation and events.

Supervised Consultation of Incoming Trunk Call

This consultation scenario is quite powerful in a customer service environment. An application monitoring the extension receiving the consultation may pop a screen using information about:

- the extension sending the consultation call (when it alerts or is answered); or,
- the original caller (when the consultation call alerts, is answered, or is conferenced/transferred with the original caller).

<Collected digits> indicates the possible presence of collected digits.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Incoming trunk call delivered to Extension 11, possibly passing through call prompter.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE ***Private Data*** ***userEnteredCode*** = <collected digits>	
The outside caller is connected to Extension 11.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE ***Private Data*** ***userEnteredCode*** = <collected digits>	
Application monitoring Extension 11 makes consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
Call with outside party is put on hold to make consultation call.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
Consultation call from Extension 11 to Extension 21 is initiated.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	
Consultation call alerts at Extension 21 with original caller information in private data.	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>
Extension 21 answers consultation call.	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>
Extension 11 and Extension 21 are connected. The “During” illustration in the figure above applies at this point.		

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Application now transfers the original caller to Extension 21.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
	CSTATransferCallConfEvent ***newCall*** = D1C3	
The T801 device appearing as the ***transferredConnections*** parameter identifies the trunk connecting the external party.	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/UNK> / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/UNK> / D2C2 21 / D3C2

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Incoming trunk call delivered to Extension 11, possibly passing through call prompter.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
The outside caller is connected to Extension 11.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
Application monitoring Extension 11 makes consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
Call with outside party is put on hold to make consultation call.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
Consultation call from Extension 11 to Extension 21 is initiated.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	

MERLIN MAGIX R2.0, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Consultation call alerts at Extension 21 with original caller information in private data.	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>
Extension 21 answers consultation call.		***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>
Extension 11 and Extension 21 are connected. The “During” illustration in the figure above applies at this point.		
Application now transfers the original caller to Extension 21.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
	CSTATransferCallConfEvent ***newCall*** = D1C3	
The T801 device appearing as the ***transferredConnections*** parameter identifies the trunk connecting the external party.	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***cause*** = EC_NEW_CALL ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***cause*** = EC_NEW_CALL ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2

MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Incoming trunk call delivered to Extension 11, possibly passing through call prompter.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
The outside caller is connected to Extension 11.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
Application monitoring Extension 11 makes consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
Call with outside party is put on hold to make consultation call.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
Consultation call from Extension 11 to Extension 21 is initiated.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	

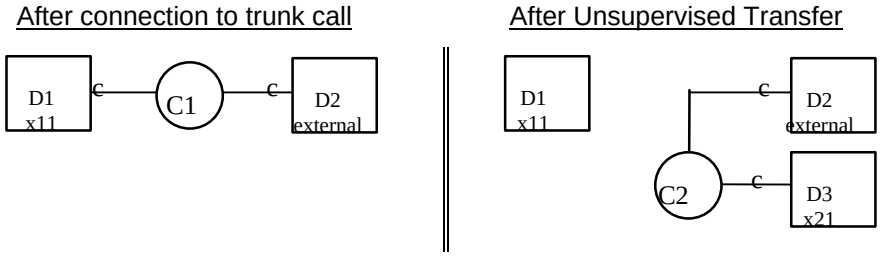
MERLIN MAGIX R2.1, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Consultation call alerts at Extension 21 with original caller information in private data.	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Extension 21 answers consultation call.		***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Extension 11 and Extension 21 are connected. The “During” illustration in the figure above applies at this point.		
Application now transfers the original caller to Extension 21.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
	CSTATransferCallConfEvent ***newCall*** = D1C3	
The T801 device appearing as the ***transferredConnections*** parameter identifies the trunk connecting the external party.	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***cause*** = EC_NEW_CALL ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***cause*** = EC_NEW_CALL ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2

Unsupervised Consultation of Incoming Trunk Call

This consultation scenario is similar to the previous one, but the transfer is unsupervised, rather than supervised. The consulting party does not have the opportunity to speak with the consulted party.

<Collected digits> indicates the possible presence of collected digits.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Incoming trunk call delivered to Extension 11, possibly passing through call prompter.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE ***Private Data*** ***userEnteredCode*** = <collected digits>	
The outside caller is connected to Extension 11.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE ***Private Data*** ***userEnteredCode*** = <collected digits>	
Application monitoring Extension 11 makes consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
Call with outside party is put on hold to make consultation call.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
Consultation call from Extension 11 to Extension 21 is initiated.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Consultation call alerts at Extension 21 with original caller information in private data.	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>
Application now does unsupervised transfer to Extension 21.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
	CSTATransferCallConfEvent ***newCall*** = D1C3	
The T801 device appearing as the ***transferredConnections*** parameter identifies the trunk connecting the external party.	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/UNK> / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/UNK> / D2C2 21 / D3C2
Extension 21 manually answers the call.	No event here since this call no longer has a connection at this device.	User at Extension 21 manually answers. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE ***Private Data*** ***userEnteredCode*** = <collected digits>

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Incoming trunk call delivered to Extension 11, possibly passing through call prompter.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
The outside caller is connected to Extension 11.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
Application monitoring Extension 11 makes consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
Call with outside party is put on hold to make consultation call.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
Consultation call from Extension 11 to Extension 21 is initiated.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	
Consultation call alerts at Extension 21 with original caller information in private data.	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>
Application now does unsupervised transfer to Extension 21.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	

MERLIN MAGIX R2.0, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
	CSTATransferCallConfEvent ***newCall*** = D1C2	
The T801 device appearing as the ***transferredConnections*** parameter identifies the trunk connecting the external party.	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** ***cause*** = EC_NEW_CALL device — after <ANI/ICLID/UNK> — D2C2 21 — D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** ***cause*** = EC_NEW_CALL device — after <ANI/ICLID/UNK> — D2C2 21 — D3C2
Extension 21 answers the call.		***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	No event here since this call no longer has a connection at this device.	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>

MERLIN MAGIX R2.1 and later

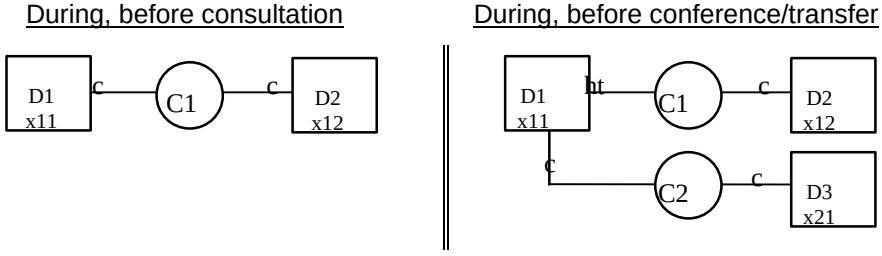
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
Incoming trunk call delivered to Extension 11, possibly passing through call prompter.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
The outside caller is connected to Extension 11.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***userEnteredCode*** = <collected digits> ***trunkUsed*** = <trunk number>	
Application monitoring Extension 11 makes consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
Call with outside party is put on hold to make consultation call.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
Consultation call from Extension 11 to Extension 21 is initiated.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	
Consultation call alerts at Extension 21 with original caller information in private data.	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***userEnteredCode*** = <collected digits>
Application now does unsupervised transfer to Extension 21.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	

MERLIN MAGIX R2.1, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 21
	CSTATransferCallConfEvent ***newCall*** = D1C2	
The T801 device appearing as the ***transferredConnections*** parameter identifies the trunk connecting the external party.	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** ***cause*** = EC_NEW_CALL device / after <ANI/ICLID/UNK> / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** ***cause*** = EC_NEW_CALL device / after <ANI/ICLID/UNK> / D2C2 21 / D3C2
Extension 21 answers the call.		***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	No event here since this call no longer has a connection at this device.	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_TRANSFER

Supervised Consultation of Internal Call

Extension 12 manually calls Extension 11. An application monitoring Extension 11 makes a consultation call to Extension 21 and then does a supervised transfer.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call from Extension 12 alerts at Extension 11. ***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	
Extension 11 manually answers. ***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	
Application monitoring Extension 11 initiates consultation call to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
		Consultation call alerts at Extension 21. ***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
		User at Extension 21 manually answers. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
Extensions 11 and 21 are connected. Application makes supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C3		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C3, D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C3, D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C3, D3C3

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call from Extension 12 alerts at Extension 11. ***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Extension 11 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Application monitoring Extension 11 initiates consultation call to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		Consultation call alerts at Extension 21. ***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
		User at Extension 21 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21

MERLIN MAGIX R2.0, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
Extensions 11 and 21 are connected. Application makes supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C2		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2

MERLIN MAGIX R2.1 and later

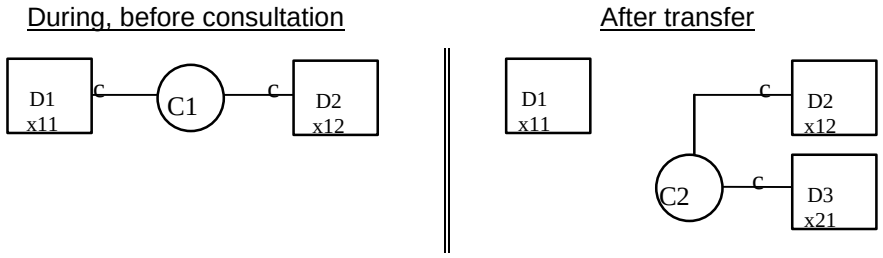
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call from Extension 12 alerts at Extension 11. ***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Extension 11 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Application monitoring Extension 11 initiates consultation call to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		Consultation call alerts at Extension 21. ***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
		User at Extension 21 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21

MERLIN MAGIX R2.1 and later, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
Extensions 11 and 21 are connected. Application makes supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C3		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2

Unsupervised Consultation of Internal Call

Extension 12 manually calls Extension 11. An application monitoring Extension 11 makes a consultation call to Extension 21 and then does an unsupervised transfer.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call from Extension 12 alerts at Extension 11. ***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	
Extension 11 manually answers. ***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	
Application monitoring Extension 11 initiates consultation call to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
		Consultation call alerts at Extension 21.
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Application makes unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C3		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3
No event here since this call no longer has a connection at this device.	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	User at Extension 21 manually answers. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call from Extension 12 alerts at Extension 11. ***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Extension 11 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Application monitoring Extension 11 initiates consultation call to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		Consultation call alerts at Extension 21. ***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
Application makes unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C2		

MERLIN MAGIX R2.0, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2
		User at Extension 21 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
No event here since this call no longer has a connection at this device.	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11

MERLIN MAGIX R2.1 and later

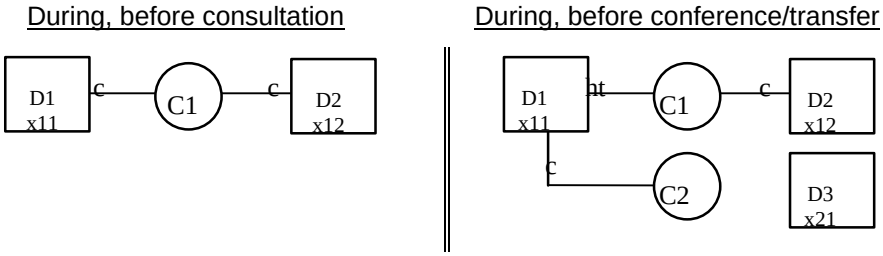
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call from Extension 12 alerts at Extension 11. ***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Extension 11 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	
Application monitoring Extension 11 initiates consultation call to Extension 21. ***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAConsultationCallConfEvent ***newCall*** = D1C2		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11		Consultation call alerts at Extension 21. ***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
Application makes unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C2		

MERLIN MAGIX R2.1 and later, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2
		User at Extension 21 manually answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
No event here since this call no longer has a connection at this device.	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11

Consultation with Consulted Device Busy (No SA)

Extension 12 manually calls Extension 11. Extension 11 attempts to consult with Extension 21 which does not have an available SA button (or any alternate call treatment administered). The consulting party at Extension 11 hears busy tone.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 12 manually calls Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE
User at Extension 11 answers.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NONE
Application monitoring Extension 11 requests consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
	CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
Consultation call cannot be delivered to Extension 21 and user at Extension 11 hears busy. The “During, before conference/transfer” diagram illustrates this point in the event flow.		
Application hangs up the attempted consultation.	***cstaClearConnection( )*** ***call*** = D1C2	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11	

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Application retrieves the original active call.	***cstaRetrieveCall( )*** ***heldCall*** = D1C1	
	CSTARetrieveCallConfEvent	
	CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11
User directs application to retry consultation to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
	CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
	CSTAConsultationCallConfEvent ***newCall*** = D1C3	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C3	
Again, no SA at Extension 21, so user at Extension 11 hears busy.		
Extension manually hangs up the attempted consultation. Extension 11 reconnects to the original caller.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C3 ***releasingDevice*** = 11 ***cstaRetrieveCall( )*** ***heldCall*** = D1C1	
	CSTARetrieveCallConfEvent	
	CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 12 manually calls Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL
User at Extension 11 answers.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL
Application monitoring Extension 11 requests consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
	CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
Consultation call cannot be delivered to Extension 21 and user at Extension 11 hears busy. The “During, before conference/transfer” diagram illustrates this point in the event flow.		
Application hangs up the attempted consultation.	***cstaClearConnection( )*** ***call*** = D1C2	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11	
Application retrieves the original active call.	***cstaRetrieveCall( )*** ***heldCall*** = D1C1	
	CSTARetrieveCallConfEvent	
	CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11
User directs application to retry consultation to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
	CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
	CSTAConsultationCallConfEvent ***newCall*** = D1C3	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C3	
Again, no SA at Extension 21, so user at Extension 11 hears busy.		

MERLIN MAGIX R2.0, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension manually hangs up the attempted consultation. Extension 11 reconnects to the original caller.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C3 ***releasingDevice*** = 11 ***cstaRetrieveCall( )*** ***heldCall*** = D1C1	
	CSTARetrieveCallConfEvent	
	CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 12 manually calls Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL
User at Extension 11 answers.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL
Application monitoring Extension 11 requests consultation call to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
	CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER
	CSTAConsultationCallConfEvent ***newCall*** = D1C2	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
Consultation call cannot be delivered to Extension 21 and user at Extension 11 hears busy. The “During, before conference/transfer” diagram illustrates this point in the event flow.		
Application hangs up the attempted consultation.	***cstaClearConnection( )*** ***call*** = D1C2	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11	
Application retrieves the original active call.	***cstaRetrieveCall( )*** ***heldCall*** = D1C1	
	CSTARetrieveCallConfEvent	
	CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11
User directs application to retry consultation to Extension 21.	***cstaConsultationCall( )*** ***activeCall*** = D1C1 ***calledDevice*** = 21	
	CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER
	CSTAConsultationCallConfEvent ***newCall*** = D1C3	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C3	
Again, no SA at Extension 21, so user at Extension 11 hears busy.		

MERLIN MAGIX R2.1 and later, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension manually hangs up the attempted consultation. Extension 11 reconnects to the original caller.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C3 ***releasingDevice*** = 11 ***cstaRetrieveCall( )*** ***heldCall*** = D1C1	
	CSTARetrieveCallConfEvent	
	CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

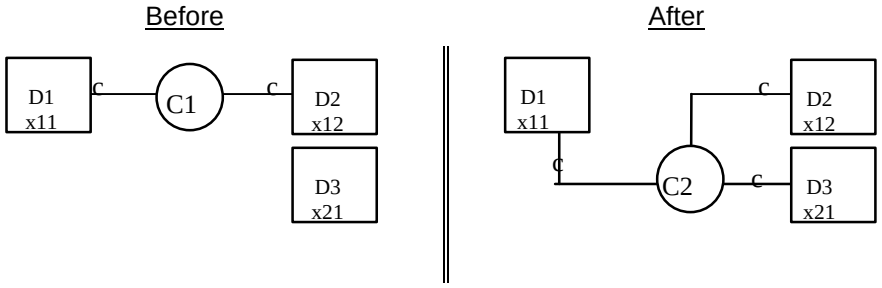
Conference Event Flows

The event flows in this section show the events that applications receive in various conference scenarios. The Consultation Event Flows cover the conference scenarios where an application uses ***cstaConsultationCall( )*** in preparation for using ***cstaConferenceCall( )***. The event flows in this section apply when a user has manually placed a call on hold for conference, then used ***cstaMakeCall( )*** (or manually made a call) and then uses ***cstaConferenceCall( )*** to conference the calls.

Not all event flows in this section contain Original Call Information (OCI) in private data. OCI is only provided:

- in the private data of the ***CSTADeliveredEvent*** and ***CSTAEstablished-Event*** when ***cstaConferenceCall( )*** is used in conjunction with ***csta-ConsultationCall( )***;
- in the private data of a CSTAEstablishedEvent following an unsupervised conference operation

Unsupervised Conference of Local Extension to Local Extension



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses CONFERENCE button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests unsupervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTAConferenceCallConfEvent ***newCall*** = D1C3		
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C3 12 D2C3 21 D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C3 12 D2C3 21 D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C3 12 D2C3 21 D3C3
Extensions 11 and 12 are hearing ringback.		Extension 21 is alerting.

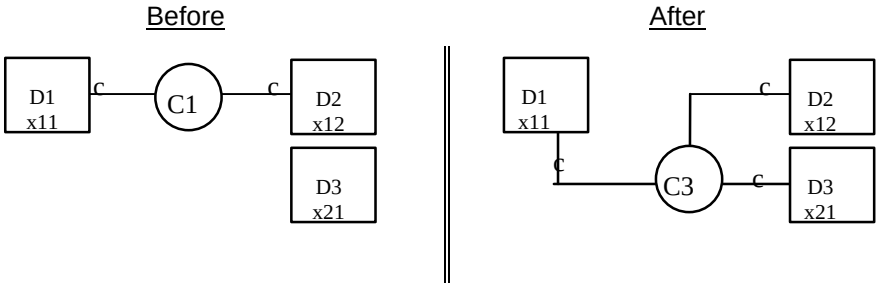
MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
		Extension 21 answers call.
CSTAEstablishedEvent	***CSTAEstablishedEvent***	***CSTAEstablishedEvent***
establishedConnection = D3C3	***establishedConnection*** = D3C3	***establishedConnection*** = D3C3
answeringDevice = 21	***answeringDevice*** = 21	***answeringDevice*** = 21
callingDevice = 11	***callingDevice*** = 11	***callingDevice*** = 11
calledDevice = 21	***calledDevice*** = 21	***calledDevice*** = 21
cause = EC_NONE	***cause*** = EC_NONE	***cause*** = EC_NONE
Private Data	***Private Data***	***Private Data***
originalCallInfo	***originalCallInfo***	***originalCallInfo***
callingDevice = 12	***callingDevice*** = 12	***callingDevice*** = 12
calledDevice = 11	***calledDevice*** = 11	***calledDevice*** = 11

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses CONFERENCE button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTAConferenceCallConfEvent ***newCall*** = D1C2		
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C2 12 D2C2 21 D3C2	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C2 12 D2C2 21 D3C2	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C2 12 D2C2 21 D3C2
Extensions 11 and 12 are hearing ringback.		Extension 21 is alerting.
		Extension 21 answers call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11

Supervised Conference of Local Extension to Local Extension



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses CONFERENCE button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
		Extension 21 answers
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests supervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTAConferenceCallConfEvent ***newCall*** = D1C3		
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C3 12 D2C3 21 D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C3 12 D2C3 21 D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device after 11 D1C3 12 D2C3 21 D3C3

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses CONFERENCE button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
		Extension 21 answers ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests supervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTAConferenceCallConfEvent ***newCall*** = D1C3		
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C3 12 / D2C3 21 / D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C3 12 / D2C3 21 / D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C3 12 / D2C3 21 / D3C3

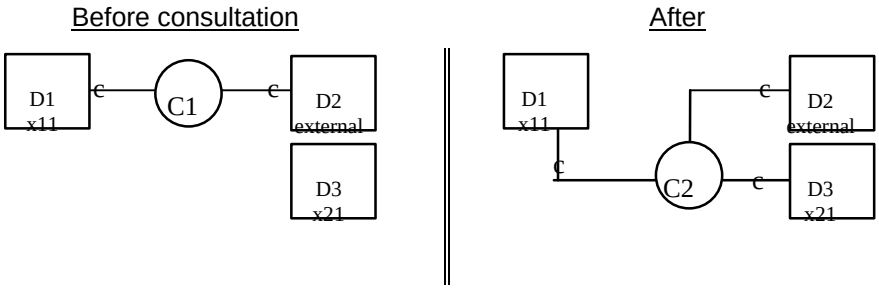
Unsupervised Conference of Incoming Trunk Call

Extension 11 is connected to a trunk call on an SA button.

The notation <ANI/ICLID/UNK> indicates that the parameter contains the ANI if the call arrived on BRI or PRI, ICLID if the call arrived on a facility that provides ICLID, and it contains “unknown” for all other conditions.

The notation <ANI/ICLID/TRK> indicates that the parameter contains the ANI if the call arrived on BRI or PRI, ICLID if the call arrived on a facility that provides ICLID, and it contains the trunk number for all other conditions.

The notation <DNIS/TRUNK> indicates that this parameter contains PRI DNIS, if any, otherwise it contains the trunk facility.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call.	
User at Extension 11 presses CONFERENCE button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests unsupervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTAConferenceCallConfEvent ***newCall*** = D1C3	

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 21
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device — after 11 — D1C3 <ANI/ICLID/TRK> — D2C3 21 — D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device — after 11 — D1C3 <ANI/ICLID/TRK> — D2C3 21 — D3C3
Extension 11 and trunk party are hearing ringback.	
	Extension 21 answers.
CSTAEstablishedEvent ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT>	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT>

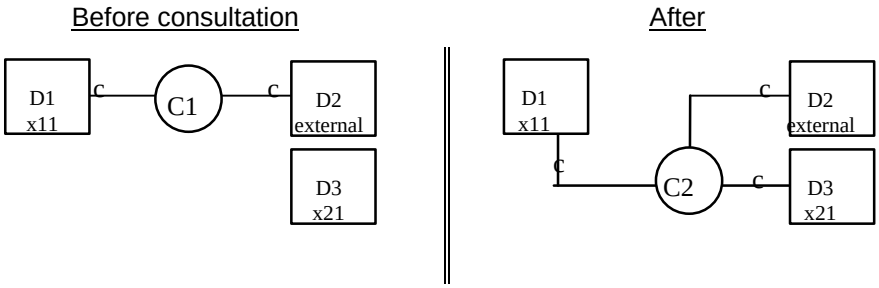
MERLIN MAGIX R2.0

Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
Extension 11 is connected to an external party on a call. User at Extension 11 presses CONFERENCE button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTAConferenceCallConfEvent ***newCall*** = D1C2	
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C2 <ANI/ICLID/TRK> / D2C2 21 / D3C2	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C2 <ANI/ICLID/TRK> / D2C2 21 / D3C2
Extension 11 and trunk party are hearing ringback.	
	Extension 21 answers.
	CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT>	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT>

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
Extension 11 is connected to an external party on a call. User at Extension 11 presses CONFERENCE button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTAConferenceCallConfEvent ***newCall*** = D1C2	
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C2 <ANI/ICLID/TRK> / D2C2 21 / D3C2	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C2 <ANI/ICLID/TRK> / D2C2 21 / D3C2
Extension 11 and trunk party are hearing ringback.	
	Extension 21 answers.
	CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL

Supervised Conference of Incoming Trunk Call



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
Extension 11 is connected to an external party on a call.	
User at Extension 11 presses CONFERENCE button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
	Extension 21 answers.
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests supervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTAConferenceCallConfEvent ***newCall*** = D1C3	
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device — after 11 — D1C3 <ANI/ICLID/TRK> — D2C3 21 — D3C3	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device — after 11 — D1C3 <ANI/ICLID/TRK> — D2C3 21 — D3C3

MERLIN MAGIX R2.0 and later

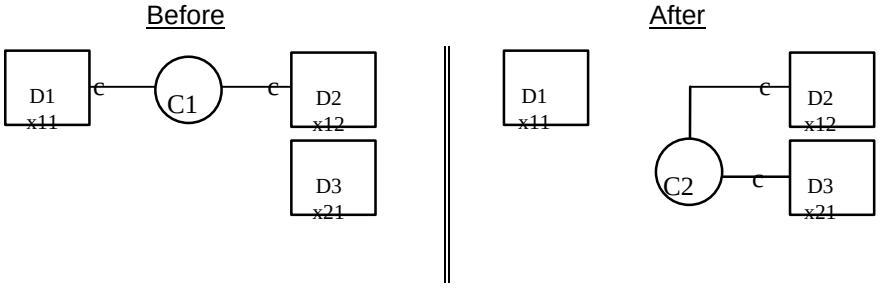
Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call.	
User at Extension 11 presses CONFERENCE button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
	Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests supervised conference. ***cstaConferenceCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTAConferenceCallConfEvent ***newCall*** = D1C2	
CSTAConferencedEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C2 <ANI/ICLID/TRK> / D2C2 21 / D3C2	***CSTAConferencedEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***confController*** = 11 ***addedParty*** = 21 ***conferenceConnections*** device / after 11 / D1C2 <ANI/ICLID/TRK> / D2C2 21 / D3C2

Transfer Event Flows

The event flows in this section show the events that applications receive in various transfer scenarios. The Consultation Event Flows cover the conference scenarios where an application uses ***cstaConsultationCall( )*** in preparation for using ***cstaTransferCall( )***. The event flows in this section apply when a user has manually placed a call on hold for transfer, then used ***cstaMakeCall( )*** (or manually made a call) and then uses ***cstaTransferCall( )*** to transfer the calls.

The event flows in this section do not contain the Original Caller Information in private data because that is passed only when ***cstaTransferCall( )*** is used in conjunction with ***cstaConsultationCall( )***.

Unsupervised Transfer of Local Extension to Local Extension



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses TRANSFER button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C3		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3
		Extension 21 answers call.
	CSTAEstablishedEvent ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE

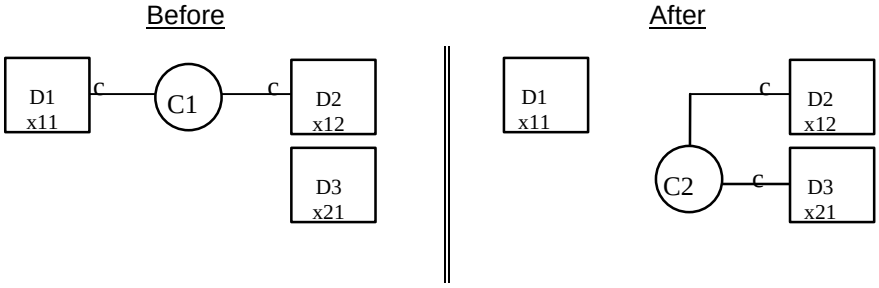
MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses TRANSFER button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C3		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 — after: D2C2, D3C2
		Extension 21 answers call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses TRANSFER button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C3		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2
		Extension 21 answers call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER

Supervised Transfer of Local Extension to Local Extension



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses TRANSFER button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
		Extension 21 answers.
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C2		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses TRANSFER button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
		Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C2		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on a call.		
User at Extension 11 presses TRANSFER button.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
		Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2		
CSTATransferCallConfEvent ***newCall*** = D1C2		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2

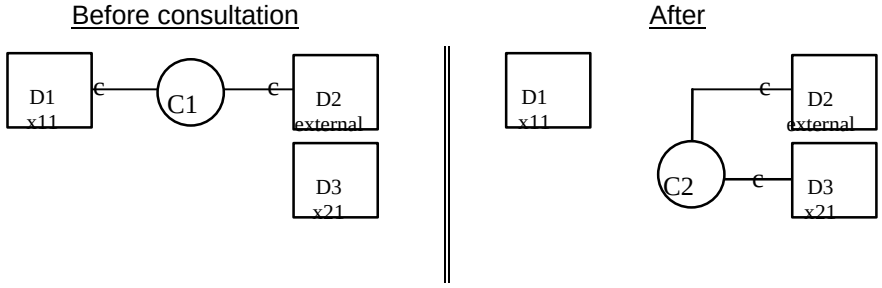
Unsupervised Transfer of Incoming Trunk Call

Extension 11 is connected to a trunk call on an SA button.

The notation <ANI/ICLID/UNK> indicates that this parameter contains ANI or ICLID when known; it contains “unknown” for all other conditions.

The notation <ANI/ICLID/TRK> indicates that the parameter contains the ANI if the call arrived on BRI or PRI, ICLID if the call arrived on a facility that provides ICLID, and it contains the trunk number for all other conditions.

The notation <DNIS/EXT> indicates that this parameter contains PRI DNIS, if any, otherwise it contains the extension number.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C3	

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 21
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: <ANI/ICLID/TRK>, after: D2C3 device: 21, after: D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: <ANI/ICLID/TRK>, after: D2C3 device: 21, after: D3C3
	Extension 21 answers. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2
	Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_TRANSFER

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2
	Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_TRANSFER

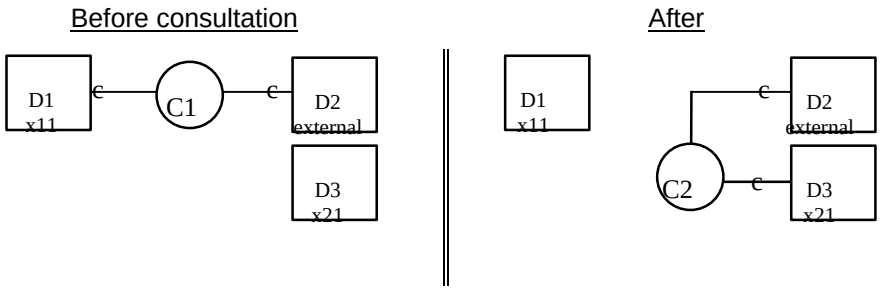
Unsupervised Transfer of Outgoing Trunk Call

Extension 11 is connected to a trunk call on a button. Extension 11 made the call.

The notation <ANI/ICLID/UNK> indicates that this parameter contains ANI or ICLID when known; it contains "unknown" for all other conditions.

The notation <ANI/ICLID/TRK> indicates that the parameter contains the ANI if the call arrived on BRI or PRI, ICLID if the call arrived on a facility that provides ICLID, and it contains the trunk number for all other conditions.

The notation <DNIS/EXT> indicates that this parameter contains PRI DNIS, if any, otherwise it contains the extension number.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** <ANI/ICLID/UNK>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** <ANI/ICLID/UNK>
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 21
CSTATransferCallConfEvent ***newCall*** = D1C3	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C3 21 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C3 21 / D3C3
	Extension 21 answers. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C3 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 21 ***cause*** = EC_NONE

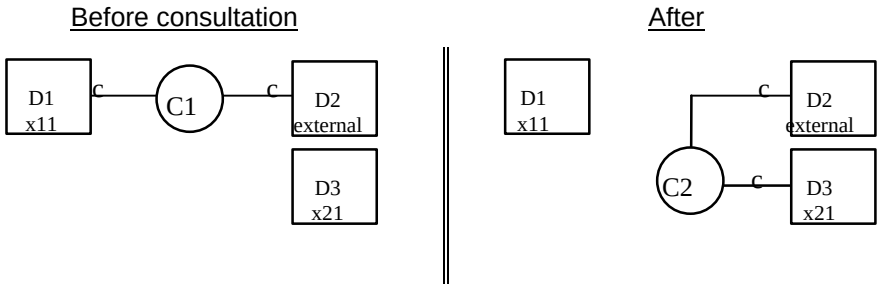
MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** <ANI/ICLID/UNK>	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** <ANI/ICLID/UNK>
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2
	Extension 21 answers.
	CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
	CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 21 ***cause*** = EC_TRANSFER

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests unsupervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after <ANI/ICLID/TRK> / D2C2 21 / D3C2 Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21 ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = 21 ***cause*** = EC_TRANSFER

Supervised Transfer of Incoming Trunk Call



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
	Extension 21 answers.
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Application requests supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C3	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C3 21 — D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C3 21 — D3C3

MERLIN MAGIX R2.0

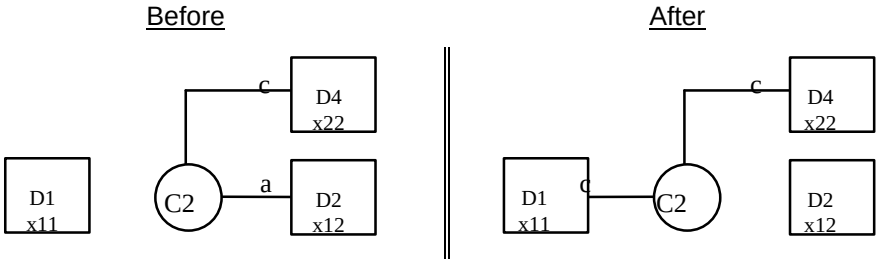
Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
	Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 is connected to an external party on a call. User at Extension 11 presses TRANSFER button.	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 dials Extension 21 and then the call alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
	Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
Application requests supervised transfer. ***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C2 21 — D3C2

Transfer Return with Answer

Extension 11 has transferred a call from Extension 22 to Extension 12 (where it is alerting). The Transfer return timer expires, and the call returns to Extension 11 and alerts. An application uses ***cstaAnswerCall( )*** to answer the returned call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
User at Extension 11 has transferred a call from Extension 22 to Extension 12, where it is alerting. The Transfer return timer now causes that call to re-alert at Extension 11.		
CSTADeliveredEvent ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE
Application uses answer to reconnect to returning call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C2		
CSTAAnswerCallConfEvent		
Established event indicates successful completion. ***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE		***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE
CSTAConnectionClearedEvent ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	Connecting to the transfer return at the transfer originator clears the connection at the transfer destination. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.0

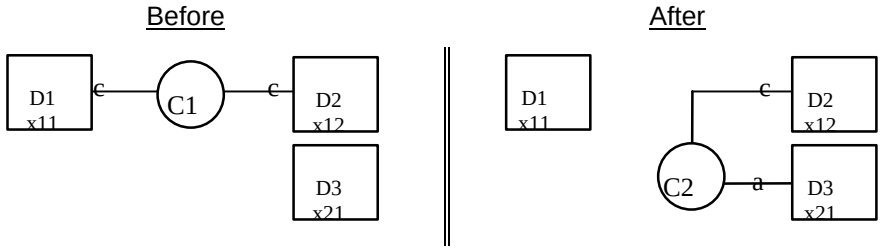
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
User at Extension 11 has transferred a call from Extension 22 to Extension 12, where it is alerting. The Transfer return timer now causes that call to re-alert at Extension 11.		
CSTADeliveredEvent ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL		***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL
Application uses answer to reconnect to returning call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C2		
CSTAAnswerCallConfEvent		
Established event indicates successful completion. ***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL		***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_RECALL
CSTAConnectionClearedEvent ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	Connecting to the transfer return at the transfer originator clears the connection at the transfer destination. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
User at Extension 11 has transferred a call from Extension 22 to Extension 12, where it is alerting. The Transfer return timer now causes that call to re-alert at Extension 11.		
CSTADeliveredEvent ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL
Application uses answer to reconnect to returning call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C2		
CSTAAnswerCallConfEvent		
Established event indicates successful completion. ***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 12 ***cause*** = EC_RECALL	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 12 ***cause*** = EC_RECALL
CSTAConnectionClearedEvent ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	Connecting to the transfer return at the transfer originator clears the connection at the transfer destination. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

Call is Answered with Voice Announce on Speaker; cstaTransferCall( ) Follows

Extension 11 and Extension 12 are connected on call C1. The user at Extension 11 presses transfer, then post selects a Voice Announce button, then calls Extension 21. Since Extension 11 made the call on a Voice Announce button, Extension 21 answers the Voice Announce call on speaker. The application then transfers call C1 to Extension 21. After the transfer, the consultation call will again alert at the transfer destination (using a new call identifier).



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
User at Extension 11 presses TRANSFER, then post selects a Voice Announce button. User at extension 11 then dials extension 21.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
		The Voice Announce Feature on Speaker immediately answers the incoming call.
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2		
		The transferred call alerts at Extension 21.
CSTADeliveredEvent ***connection*** = D3C3 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D3C3 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D3C3 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTATransferCallConfEvent ***newCall*** = D3C3		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / before / after 12 / D2C1 / D2C3 21 / D3C2 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
User at Extension 11 presses TRANSFER, then post selects a Voice Announce button. User at extension 11 then dials extension 21.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
		The Voice Announce Feature on Speaker immediately answers the incoming call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2		
		The transferred call alerts at Extension 21.
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER
CSTATransferCallConfEvent ***newCall*** = D3C2		

MERLIN MAGIX R2.0, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 before: D2C1, D3C2 after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 after: D2C2, D3C2
		CSTAReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
		Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D1C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER

MERLIN MAGIX R2.1 and later

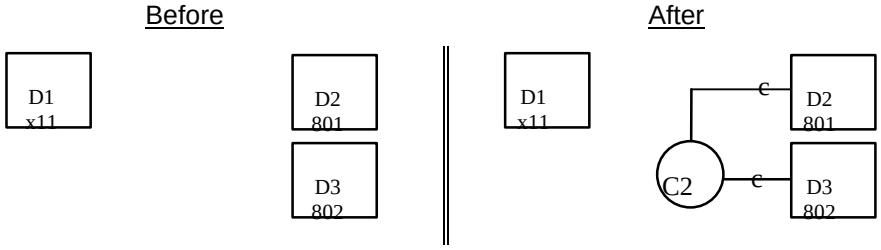
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
User at Extension 11 presses TRANSFER, then post selects a Voice Announce button. User at extension 11 then dials extension 21.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
		The Voice Announce Feature on Speaker immediately answers the incoming call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2		
		The transferred call alerts at Extension 21.
CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER
CSTATransferCallConfEvent ***newCall*** = D3C2		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / before / after 12 / D2C1 / D2C2 21 / D3C2 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / before / after 12 / D2C1 / D2C2 21 / D3C2 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / before / after 12 / D2C1 / D2C2 21 / D3C2 / D3C2
		CSTAReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
		Extension 21 answers. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D1C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C2 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_TRANSFER

Trunk-to-Trunk Transfer

Extension 11 receives an incoming trunk call on a DFT. Extension 11 then (manually) puts it on hold for transfer, and (manually) transfers the call to an external party.

The notation <ANI/ICLID/UNK> indicates that this parameter contains ANI if the call arrived on BRI or PRI, ICLID if the call arrived on a facility that provides ICLID, and it contains "unknown" for all other conditions.

The notation <DNIS/EXT> indicates that this parameter contains DNIS if the call arrived on a facility that provides DNIS. Otherwise, the parameter contains the extension number.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 &
MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11
Extension 11 receives an incoming trunk call on a DFT. If the call arrives on an SA, an application monitoring Extension 11 would receive a delivered event.	
User at Extension 11 answers call.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NONE
User at Extension 11 presses TRANSFER.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2
User at Extension 11 dials an external number.	***CSTANetworkReachedEvent*** ***connection*** = D2C2 ***trunkUsed*** = T801 ***calledDevice*** = 95551234
Application does trunk-to-trunk transfer.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2
The device in the ***newCall*** parameter is T802.	***CSTATransferCallConfEvent*** ***newCall*** = D3C3

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 &
MERLIN MAGIX R1.0 and 1.5, continued

Activity	Stream Monitoring Extension 11
	CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = T802 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C3 5551234 — D3C3

MERLIN MAGIX R2.0

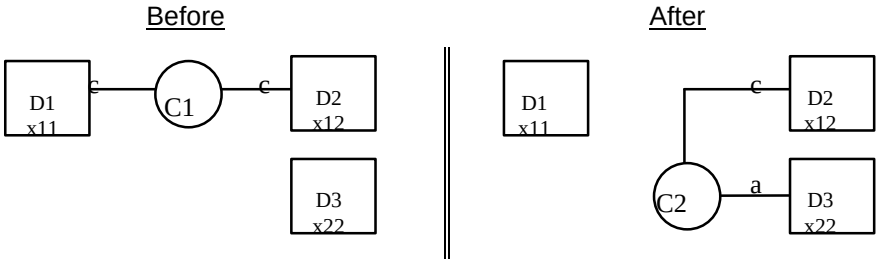
Activity	Stream Monitoring Extension 11
Extension 11 receives an incoming trunk call on an SA, DFT, or DPT button. An application monitoring Extension 11 would receive a delivered event.	***CSTADeliveredEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL
User at Extension 11 answers call.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL
User at Extension 11 presses TRANSFER.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2
User at Extension 11 dials an external number.	***CSTANetworkReachedEvent*** ***connection*** = D2C2 ***trunkUsed*** = T801 ***calledDevice*** = 95551234
Application does trunk-to-trunk transfer.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2
The device in the ***newCall*** parameter is T802.	***CSTATransferCallConfEvent*** ***newCall*** = D3C3
	CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = T802 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C3 5551234 — D3C3

MERLIN MAGIX R2.1and later

Activity	Stream Monitoring Extension 11
Extension 11 receives an incoming trunk call on an SA, DFT, or DPT button. An application monitoring Extension 11 would receive a delivered event.	***CSTADeliveredEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_NEW_CALL
User at Extension 11 answers call.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_NEW_CALL
User at Extension 11 presses TRANSFER.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2
User at Extension 11 dials an external number.	***CSTANetworkReachedEvent*** ***connection*** = D2C2 ***trunkUsed*** = T801 ***calledDevice*** = 95551234
Application does trunk-to-trunk transfer.	***cstaTransferCall( )*** ***heldCall*** = D1C1 ***activeCall*** = D1C2
The device in the ***newCall*** parameter is T802.	***CSTATransferCallConfEvent*** ***newCall*** = D3C3
	CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = T802 ***transferredConnections*** device — after <ANI/ICLID/TRK> — D2C3 5551234 — D3C3

Transfer into DGC Group with No Members Available; Member Becomes Available

Extension 11 is connected to Extension 12 on call C1. Extension 11 puts call C1 on hold-for-transfer, manually makes a call to a DGC Group (with no members available), and then manually transfers the call to the DGC Group. Extension 22, a member of the DGC Group, then becomes available and the call rings at Extension 22. Note that D4 is DGC queue 770.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
User at Extension 11 manually places call C1 on hold and makes a call to an unstaffed DGC Group.		
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
User at Extension 11 manually completes the transfer to the DGC Group.		
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = ***transferredConnections*** device: 12 after: D2C3		
Extension 22 becomes an available member of DGC Group.		
	CSTADeliveredEvent ***connection*** = D3C3 ***alertingDevice*** = 22 ***callingDevice*** = 12 ***calledDevice*** = 22 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D3C3 ***alertingDevice*** = 22 ***callingDevice*** = 12 ***calledDevice*** = 22 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22	Stream Monitoring DGC Group 770
User at Extension 11 manually presses the TRANSFER button, placing call C1 on hold, and makes a call to unstaffed DGC Group 770.			
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2			
CSTAQueuedEvent ***queuedConnection*** = D4C2 ***queue*** = Q770 ***callingDevice*** = 11 ***calledDevice*** = Q770 ***cause*** = EC_NONE ***numberQueued*** = 1			***CSTAQueuedEvent*** ***queuedConnection*** = D4C2 ***queue*** = Q770 ***callingDevice*** = 12 ***calledDevice*** = Q770 ***cause*** = EC_NONE ***numberQueued*** = 1
User at Extension 11 manually presses the TRANSFER button to complete the transfer.			
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = Q770 ***transferredConnections*** device / after 12 / D2C2 Q770 / D4C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = Q770 ***transferredConnections*** device / after 12 / D2C2 Q770 / D4C2		
Extension 22 becomes an available member of DGC Group 770.			
		CSTALoggedOnEvent ***agentDevice*** = 22 ***agentID*** = 22 ***agentGroup*** = Q770	
	CSTADivertedEvent ***connection*** = D4C2 ***divertingDevice*** = Q770 ***newDestination*** = 22 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C2 ***divertingDevice*** = Q770 ***newDestination*** = 22 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C2 ***divertingDevice*** = Q770 ***newDestination*** = 22 ***cause*** = EC_REDIRECTED
	CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 22 ***callingDevice*** = 12 ***calledDevice*** = 22 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 22 ***callingDevice*** = 12 ***calledDevice*** = 22 ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	

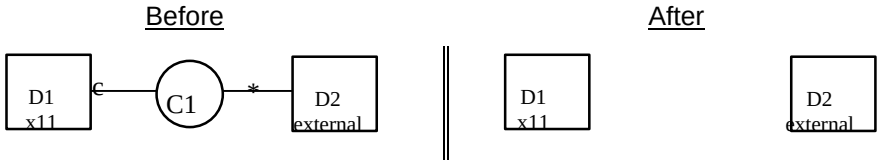
MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22	Stream Monitoring DGC Group 770
User at Extension 11 manually presses the TRANSFER button, placing call C1 on hold, and makes a call to unstaffed DGC Group 770.			
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** =EC_TRANSFER		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2			
CSTAQueuedEvent ***queuedConnection*** = D4C2 ***queue*** = Q770 ***callingDevice*** = 11 ***calledDevice*** = Q770 ***cause*** = EC_NONE ***numberQueued*** = 1			***CSTAQueuedEvent*** ***queuedConnection*** = D4C2 ***queue*** = Q770 ***callingDevice*** = 12 ***calledDevice*** = Q770 ***cause*** = EC_NONE ***numberQueued*** = 1
User at Extension 11 manually presses the TRANSFER button to complete the transfer.			
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = Q770 ***transferredConnections*** device / after 12 / D2C2 Q770 / D4C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = Q770 ***transferredConnections*** device / after 12 / D2C2 Q770 / D4C2		
Extension 22 becomes an available member of DGC Group 770.			
		CSTALoggedOnEvent ***agentDevice*** = 22 ***agentID*** = 22 ***agentGroup*** = Q770	
	CSTADivertedEvent ***connection*** = D4C2 ***divertingDevice*** = Q770 ***newDestination*** = 22 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C2 ***divertingDevice*** = Q770 ***newDestination*** = 22 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C2 ***divertingDevice*** = Q770 ***newDestination*** = 22 ***cause*** = EC_REDIRECTED
	CSTADeliveredEvent ***connection*** = D3C2 ***alertingDevice*** = 22 ***callingDevice*** = 11 ***calledDevice*** = <Q770> ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***connection*** = D3C2 ***alertingDevice*** = 22 ***callingDevice*** = 11 ***calledDevice*** = <Q770> ***lastRedirectionDevice*** = Q770 ***cause*** = EC_REDIRECTED	

Feature Invocation Event Flows

Account Code Entry/Forced Account Code Entry (ACE/FACE)

Extension 11 is on an external call. During the call, Extension 11 enters an account code. Extension 11 hangs up.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11
Extension 11 is active on a call at an SA, DFT, or DPT button. During the call, Extension 11 enters an account code.	
Extension 11 hangs up.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11
Extension 11 is active on a call at an SA, DFT, or DPT button. During the call, Extension 11 enters an account code.	
Extension 11 hangs up.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE ***Private Data*** ***accountCode*** = <account code>
	CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11

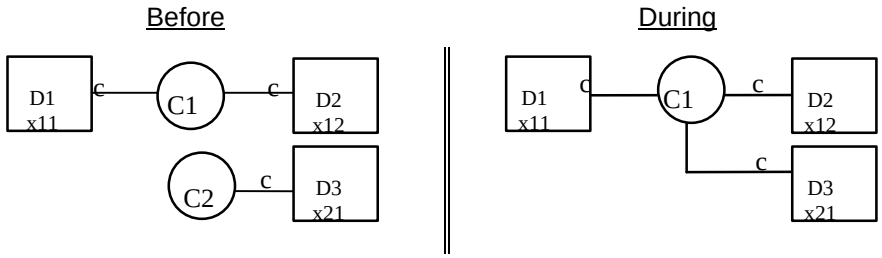
MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring Extension 11
Extension 11 is active on a call at an SA, DFT, or DPT button.	
During the call, Extension 11 enters an account code.	***CSTACallInformationEvent*** ***connection*** = D1C1 ***device*** = 11 ***accountCode*** = <account code>
Extension 11 hangs up.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE ***Private Data*** ***accountCode*** = <account code>
	CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11

Barge-In

Barge-In to Busy Extension

Extension 11 is busy on a call with Extension 12. Extension 21, attempting to call Extension 11, meets busy condition and will barge-in.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

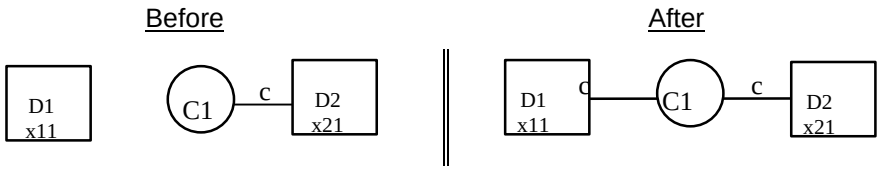
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on call C1.		
		Extension 21 calls Extension 11. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D3C2
		User at Extension 21 hears busy and presses BARGE-IN button.
		CSTAConnectionClearedEvent ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
Extension 21 barges into call C1 with Extensions 11, 21, and 21 hearing barge-in tone. Note that there is no event indicating that the barge-in has occurred. The “During” illustration in the figure above applies at this point.		
		User at Extension 21 hangs up.
No event occurs when the user that barged in drops off.		

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extensions 11 and 12 are connected on call C1.		
		Extension 21 calls Extension 11.. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
		CSTAServiceInitiatedEvent ***initiatedConnection*** = D3C2
		User at Extension 21 hears busy and presses BARGE-IN button.
		CSTAConnectionClearedEvent ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
Extension 21 barges into call C1 with Extensions 11, 21, and 21 hearing barge-in tone. Note that there is no event indicating that the barge-in has occurred. The “During” illustration in the figure above applies at this point.		
		User at Extension 21 hangs up.
		CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
No event occurs when the user that barged in drops off.		

Barge-In Overrides Do Not Disturb at Extension

Extension 21, attempting to call Extension 11, meets an active Do Not Disturb condition and will barge-in. An application will use ***cstaAnswerCall( )*** to answer the Barge-In call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 has an active Do Not Disturb condition.	
	Extension 21 calls Extension 11. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D2C1
	User at Extension 21 hears busy tone (because an active Do Not Disturb condition is encountered) and presses BARGE-IN button.
CSTADeliveredEvent ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NONE
Application answers barge-in call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C1	
CSTAAnswerCallConfEvent	
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NONE
	cstaClearConnection( ) ***call*** = D2C1
	CSTAClearConnectionConfEvent
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	

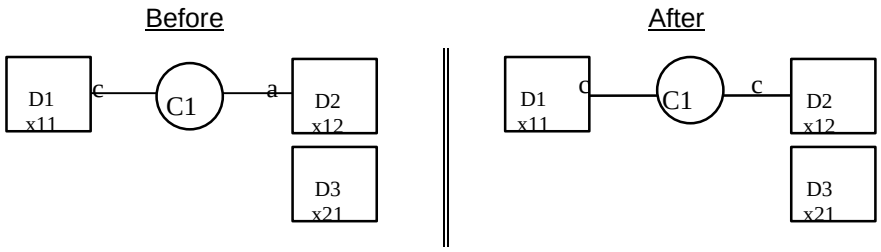
MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 21
Extension 11 has an active Do Not Disturb condition.	
	Extension 21 calls Extension 11. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D2C1 User at Extension 21 hears busy tone (because an active Do Not Disturb condition is encountered) and presses BARGE-IN button.
CSTADeliveredEvent ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL
Application answers barge-in call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C1	
CSTAAnswerCallConfEvent	
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_NEW_CALL
	cstaClearConnection( ) ***call*** = D2C1
	CSTAClearConnectionConfEvent
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	
	CSTAReadyEvent ***agentDevice*** = 21 ***agentID*** = 21

Call Forward/Follow Me

Forwarding Extension Answers (Forward to Internal Number Only)

Extension 11 places a call to forwarding Extension 12. The call will alert at both the Extension 12 and the forwarding destination, Extension 21. The call is answered at Extension 12.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1		
	Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	
	Application answers call at Extension 12 ***cstaAnswerCall( )*** ***alertingCall*** = D2C1	
	CSTAAnswerCallConfEvent	
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	

MERLIN MAGIX R2.0

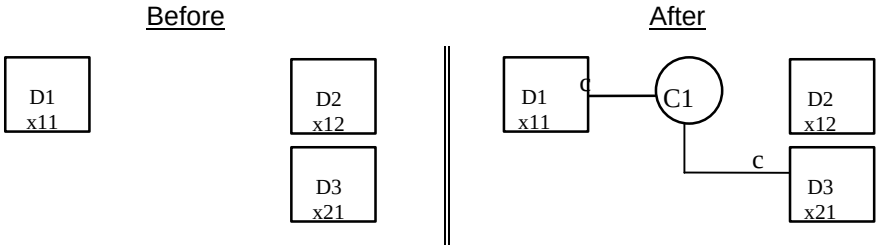
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
	Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	Call delivered to Extension 21. ***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12
	Application answers call at Extension 12. ***cstaAnswerCall( )*** ***alertingCall*** = D2C1	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	Call is cleared from Extension 21. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
	Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL		
		Call delivered to Extension 21. ***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS	
	Application answers call at Extension 12. ***cstaAnswerCall( )*** ***alertingCall*** = D2C1	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
		Call is cleared from Extension 21.
CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	

Forward-to Extension Answers

Extension 11 places a call to forwarding Extension 12. The call will alert at both the Extension 12 and the forwarding destination, Extension 21. The call is answered at Extension 21.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1		
Call is delivered to Extension 12 and forwarded to Extension 21, where it is alerting. There is no ***CSTADeliveredEvent*** when a forwarded call alerts at the extension to which it was forwarded.		
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	Forwarded call still has connection at the forwarding extension, so events continue to flow on that monitor. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	User at Extension 21 manually answers. An application monitoring an extension that manually answers a forwarded call will get a ***CSTAEstablishedEvent*** for the forwarded call. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	When Forward-to device answers, connection is cleared from the forwarding extension. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.0

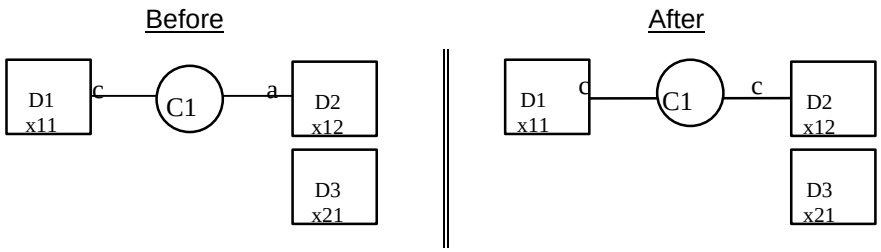
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
	Call forwards and alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12
		Application answers call at Extension 21. ***cstaAnswerCall( )*** ***alertingCall*** = D3C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	Forwarded call still has connection at the forwarding extension, so events continue to flow on that monitor. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	When Forward-to device answers, connection is cleared from the forwarding extension. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = = EC_CALL_CANCELLED

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
	Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL		
	Call forwards and alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 21 ***cause*** = EC_CALL_FORWARD_ALWAYS	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 21 ***cause*** = EC_CALL_FORWARD_ALWAYS	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS
		Application answers call at Extension 21. ***cstaAnswerCall( )*** ***alertingCall*** = D3C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS	Forwarded call still has connection at the forwarding extension, so events continue to flow on that monitor. ***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	When Forward-to device answers, connection is cleared from the forwarding extension. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = = EC_CALL_CANCELLED

Delayed Call Forwarding - Forwarding Extension Answers (Forward to Internal Number Only)

Extension 11 places a call to forwarding Extension 12. The call will alert for one or more ring cycles at the forwarding extension before being forwarded. The call is answered at Extension 12 before the call is forwarded.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1		
	Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	
	Application answers call before call is forwarded. ***cstaAnswerCall( )*** ***alertingCall*** = D2C1	
	CSTAAnswerCallConfEvent	
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	

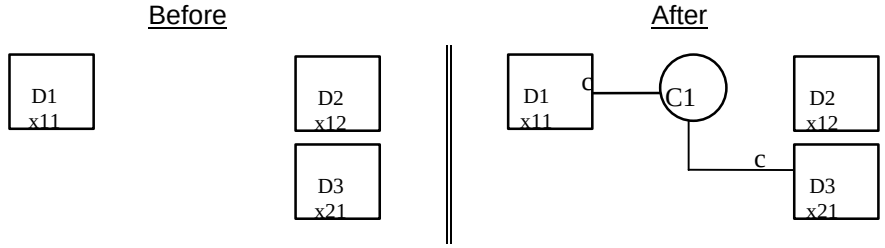
MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
	Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL		
	Application answers call before call is forwarded. ***cstaAnswerCall( )*** ***alertingCall*** = D2C1	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	

Call Forward on Busy

When an extension forwards a call using the Call Forward on Busy feature, the forwarded call does not appear at the forwarding extension. Thus, the event flow is similar to the other forwarding event flows, but the forwarding extension cannot connect to the call before it forwards.

Extension 11 calls Extension 12, who is busy and has the Call Forward on Busy feature active.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1		
Call cannot alert at Extension 12, so it forwards and alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
		User at Extension 21 manually answers.
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

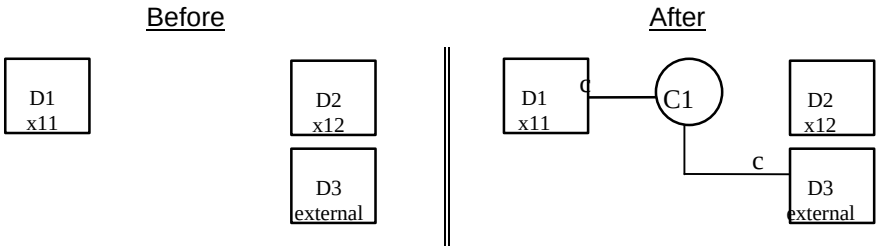
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
Call cannot alert at Extension 12, so it forwards and alerts at Extension 21.		
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL		***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL
		Application answers call at Extension 21. ***cstaAnswerCall( )*** ***alertingCall*** = D3C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevic***e = 12 ***cause*** = EC_CALL_FORWARD		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevic***e = 12 ***cause*** = EC_CALL_FORWARD

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
	Call cannot alert at Extension 12, so it forwards and alerts at Extension 21.	
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
		Application answers call at Extension 21. ***cstaAnswerCall( )*** ***alertingCall*** = D3C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevic***e = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevic***e = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevic***e = 12 ***cause*** = EC_CALL_FORWARD_ALWAYS

Remote Call Forwarding with Delay

Extension 11 calls Extension 12, where the call forwards with delay to an external number.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

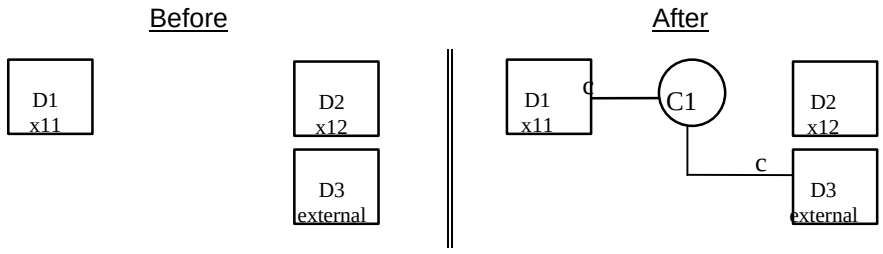
Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 calls Extension 12. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	Call delivered to Extension 12 and rings for delay period. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
Call forwards.	
CSTANetworkReachedEvent ***connection*** = D3C1 ***trunkUsed*** = T801 ***calledDevice*** = Unknown	Forwarded call still has connection at the forwarding extension, so events continue to flow on that monitor. ***CSTANetworkReachedEvent*** ***connection*** = D3C1 ***trunkUsed*** = T801 ***calledDevice*** = Unknown
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	When call forwards off the switch, connection is cleared from the forwarding extension. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	Call delivered to Extension 12 and rings for delay period. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
Call forwards.	
CSTANetworkReachedEvent ***connection*** = D3C1 ***trunkUsed*** = T801 ***calledDevice*** = Unknown	Forwarded call still has connection at the forwarding extension, so events continue to flow on that monitor. ***CSTANetworkReachedEvent*** ***connection*** = D3C1 ***trunkUsed*** = T801 ***calledDevice*** = Unknown
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	When call forwards off the switch, connection is cleared from the forwarding extension. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE

Remote Call Forwarding Without Delay

Extension 11 calls Extension 12 which forwards the call without delay to an external number. The call never has a connection to Extension 12.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

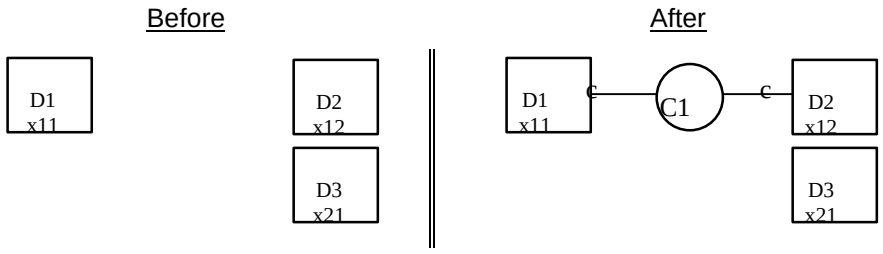
Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 calls Extension 12. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1	
Call forwards.	
CSTANetworkReachedEvent ***connection*** = D3C1 ***trunkUsed*** = T801 ***calledDevice*** = Unknown	

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 calls Extension 12. ***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
Call forwards.	
CSTANetworkReachedEvent ***connection*** = D3C1 ***trunkUsed*** = T801 ***calledDevice*** = Unknown	

Call Screening

Extension 11 places a call to Extension 12, where the Call Screening feature active. The call alerts at Extension 12, but is not answered, so the call receives Voice Mail treatment. When the call is answered by the Voice Mail port (Extension 21), Extension 12 is added to the call as a Call Screener. After establishing the identity of the caller and the reason for the call, the user at Extension 12 chooses to join the call as a regular call participant, causing the Voice Mail port to be dropped from the call.



MERLIN MAGIX R2.1 and later

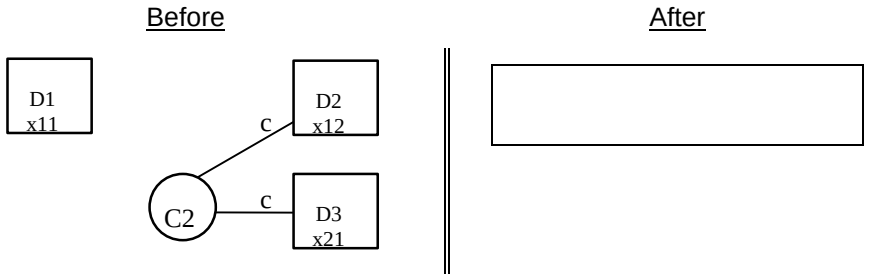
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1		
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = ID_NOT_KNOWN ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = ID_NOT_KNOWN ***cause*** = EC_NEW_CALL	

MERLIN MAGIX R2.1 and later (continued)

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
The call is not answered, so it receives Voice Mail treatment.		
CSTAQueuedEvent ***connection*** = D4C1 ***queue*** = D4 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	***CSTAQueuedEvent*** ***connection*** = D4C1 ***queue*** = D4 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	
CSTADivertedEvent ***connection*** = D4C1 ***divertingDevice*** = D4 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C1 ***divertingDevice*** = D4 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C1 ***divertingDevice*** = D4 ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = D4 ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = D4 ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = D4 ***cause*** = EC_REDIRECTED
The Voice Mail port answers the call.		
CSTAEstablishedEvent ***connection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = D4 ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***connection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = D4 ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***connection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = D4 ***cause*** = EC_REDIRECTED
Extension 12 is added to the call as a Call Screener.		
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAEstablishedEvent ***connection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = ID_NOT_KNOWN ***cause*** = EC_SILENT_MONITOR	
Extension 12 joins the call as a regular call participant, causing the Voice Mail port to be dropped from the call.		
CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

Call Waiting

User at Extension 11 places call to Extension 12 which waits. There is not a Delivered event for the arrival of that call at Extension 12 since it has not yet alerted there.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 calls Extension 12.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1	
Extension 12 is busy on another call and Extension 12 has Call Waiting, so call C1 waits on Extension 12.		
User at Extension 12 hangs up call C2.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12
Call C1 now alerts at Extension 12.	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12
Application answers call.		***cstaAnswerCall( )*** ***alertingCall*** = D2C1
		CSTAAnswerCallConfEvent
	CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12

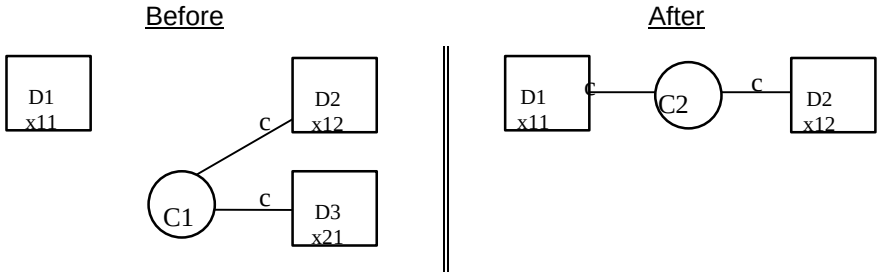
MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 calls Extension 12.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1	
Extension 12 is busy on another call and Extension 12 has Call Waiting, so call C1 waits on Extension 12.		
User at Extension 12 hangs up call C2.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12
Call C1 now alerts at Extension 12.	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
Application answers call.		***cstaAnswerCall( )*** ***alertingCall*** = D2C1
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12
	CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL

Callback Queuing (CBQ)

Callback - User Stays On Line

Extension user at Extension 11 places a call to Extension 12, which is busy. User at Extension 11 invokes the callback feature and stays connected to the call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

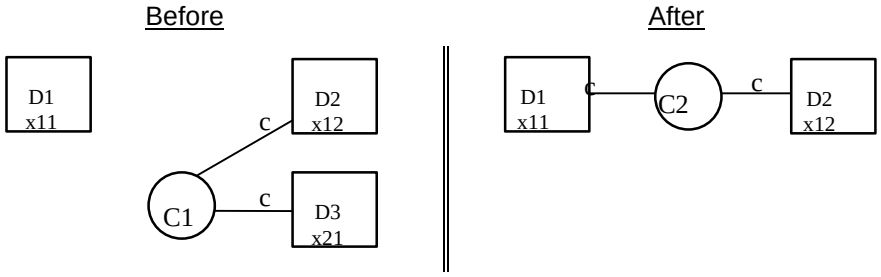
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 makes call C2 to Extension 12.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	
The caller hears busy tone and invokes CBQ. The user at Extension 11 hears queuing tone.		
User at Extension 12 hangs up call C1.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12
User at Extension 11 hears de-queue tone and call C2 now alerts at Extension 12.	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE
Application answers call.		***cstaAnswerCall( )*** ***alertingCall*** = D2C2
		CSTAAnswerCallConfEvent
	CSTAEstablishedEvent ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 makes call C2 to Extension 12.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
The caller hears busy tone and invokes CBQ. The user at Extension 11 hears queuing tone.		
User at Extension 12 hangs up call C1.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12
		CSTAReadyEvent ***agentDevice*** = 12 ***agentID*** = 12
User at Extension 11 hears de-queue tone and call C2 now alerts at Extension 12.	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
Application answers call.		***cstaAnswerCall( )*** ***alertingCall*** = D2C2
		CSTAAnswerCallConfEvent
		CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12
	CSTAEstablishedEvent ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL

Callback - Caller Goes On Hook on Callback Call

Extension user at Extension 11 places a call to Extension 12, which is busy. User at Extension 11 invokes the callback feature and hangs up.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

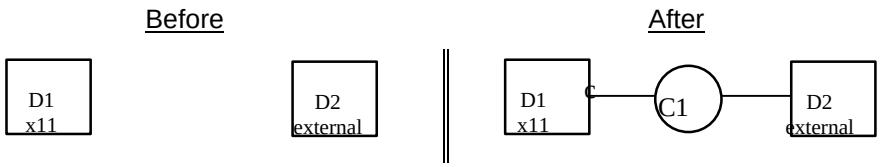
Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 makes call C2 to Extension 12.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2	
The caller hears busy tone and invokes CBQ. The user at Extension 11 hears queuing tone.		
Application hangs up callback call. The call transitions into Associative Hold at Extension 11.	***cstaClearConnection( )*** ***call*** = D1C2	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11 ***cause*** = EC_CANCELLED	
User at Extension 12 hangs up call C1.		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12
User at Extension 11 hears priority ring. Application cannot use ***cstaAnswerCall( )*** to answer this priority ring. User manually answers priority ring.		
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
User at Extension 11 hears de-queue tone and call C3 now alerts at Extension 12.	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11	Stream Monitoring Extension 12
User at Extension 11 makes call C2 to Extension 12.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
The caller hears busy tone and invokes CBQ. The user at Extension 11 hears queuing tone.		
Application hangs up callback call. The call transitions into Associative Hold at Extension 11.	***cstaClearConnection( )*** ***call*** = D1C2	
	CSTAClearConnectionConfEvent	
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	
	CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	
User at Extension 12 hangs up call C1.		***CSTAReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12
		CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED
User at Extension 11 hears priority ring. Application cannot use ***cstaAnswerCall( )*** to answer this priority ring. User manually answers priority ring.		
	CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C3	
User at Extension 11 hears de-queue tone and call C3 now alerts at Extension 12.	***CSTADeliveredEvent*** ***connection*** = D2C3 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C3 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL

Callback Queuing for Pool or ARS; Caller Waits Off-Hook

Extension user at Extension 11 places an external call using the ARS code or a Pool Code. The outgoing call queues for a facility. In this flow, the calling user does not go on hook.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

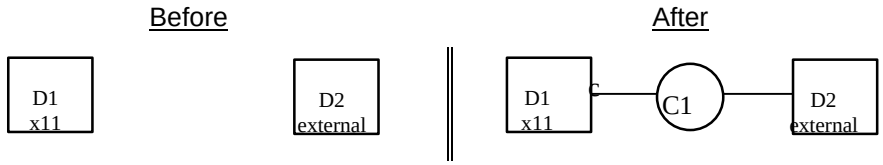
Activity	Stream Monitoring Extension 11
User at Extension 11 makes call C1 to external number.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1
The call queues for a facility. The user at Extension 11 hears queuing tone.	
The facility becomes available. User at Extension 11 hears de-queue tone and switch makes outbound call on trunk.	
Note that there is no ARS digit or Pool Code in the ***calledDevice*** parameter.	***CSTANetworkReachedEvent*** ***connection*** = D2C1 ***trunkUsed*** = T801 ***calledDevice*** = 5551234

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11
User at Extension 11 makes call C1 to external number.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C1
The call queues for a facility. The user at Extension 11 hears queuing tone.	
The facility becomes available. User at Extension 11 hears de-queue tone and switch makes outbound call on trunk.	
Note that there is no ARS digit or Pool Code in the ***calledDevice*** parameter.	***CSTANetworkReachedEvent*** ***connection*** = D2C1 ***trunkUsed*** = T801 ***calledDevice*** = 5551234

Callback Queuing for Pool or ARS; Caller Goes On Hook

Extension user at Extension 11 places an external call using the ARS code or a Pool Code. The outgoing call queues for a facility. In this flow, the calling user goes on hook.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11
User at Extension 11 makes call C1 to external number.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1
The call queues for a facility. The user at Extension 11 hears queuing tone.	
The user at Extension 11 goes on hook.	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE
The facility becomes available. User at Extension 11 hears priority ring. User cannot use answer to answer this priority ring. User manually answers priority ring.	
User at Extension 11 goes off-hook.	***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C2
Note that there is no ARS digit or Pool Code in the ***calledDevice*** parameter.	***CSTANetworkReachedEvent*** ***connection*** = D1C2 ***trunkUsed*** = T801 ***calledDevice*** = 5551234

MERLIN MAGIX R2.0 and later

Activity	Stream Monitoring Extension 11
User at Extension 11 makes call C1 to external number.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11 ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D1C1
The call queues for a facility. The user at Extension 11 hears queuing tone.	
The user at Extension 11 goes on hook.	***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE
The facility becomes available. User at Extension 11 hears priority ring. User cannot use answer to answer this priority ring. User manually answers priority ring.	
User at Extension 11 goes off-hook.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2
Note that there is no ARS digit or Pool Code in the ***calledDevice*** parameter.	***CSTANetworkReachedEvent*** ***connection*** = D1C2 ***trunkUsed*** = T801 ***calledDevice*** = 5551234

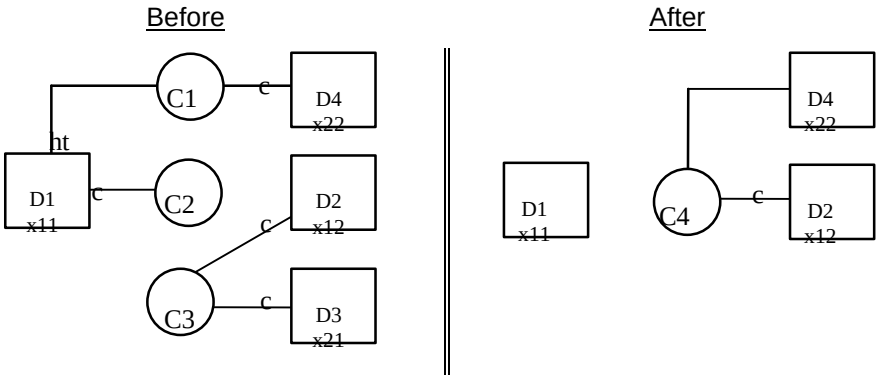
Camp On

The Camp On Feature allows a user to

- complete a transfer to a busy station;
- transfer a call and have the call return using the Camp On return timer rather than the Transfer return timer (The Camp On return timer is typically longer than the Transfer return timer).

Camp On Completes Transfer to Busy Extension; Destination Comes Available and Answers

Extension 11 has placed a call with Extension 22 on hold for transfer (manually or with an application using ***cstaConsultationCall( )***). The user at Extension 11 transfers the call to Extension 12 (manually or using ***cstaTransferCall( )***), who is busy on a call with Extension 21. The transferred call camps on for Extension 12. Extension 12 hangs up on its call with Extension 21and the camped on call alerts at Extension 12, where the user answers it.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
Extension 11 presses TRANSFER button to transfer call with Extension 22 to Extension 12. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
Extension 11 dials Extension 12, hears busy and presses transfer. Call automatically camps on to Extension 12. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = /* none */ ***transferredConnections*** device after 22 D4C4		***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = /* none */ ***transferredConnections*** device after 22 D4C4
Transferred call is now camped on for Extension 12.		
	User at Extension 12 finishes with other call. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C3 ***releasingDevice*** = 12 ***cause*** = EC_NONE	
	CSTADeliveredEvent ***connection*** = D2C4 ***alertingDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D2C4 ***alertingDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22
	Application answers camped-on call. ***cstaAnswerCall( )*** ***alertingCall*** = D2C4	
	CSTAAnswerCallConfEvent	
	CSTAEstablishedEvent ***establishedConnection*** = D2C4 ***answeringDevice*** = 12 ***callingDevice*** =22 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C4 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
Extension 11 presses TRANSFER button to transfer call with Extension 22 to Extension 12. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
Extension 11 dials Extension 12, hears busy and presses transfer. Call automatically camps on to Extension 12. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = /* none */ ***transferredConnections*** device after 22 D4C2		***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = /* none */ ***transferredConnections*** device after 22 D4C2
Transferred call is now camped on for Extension 12.		
	User at Extension 12 finishes with other call. ***CSTAReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C3 ***releasingDevice*** = 12 ***cause*** = EC_NONE	
	CSTADeliveredEvent ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22
	Application answers camped-on call. ***cstaAnswerCall( )*** ***alertingCall*** = D2C2	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAEstablishedEvent ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER

MERLIN MAGIX R2.1 and later

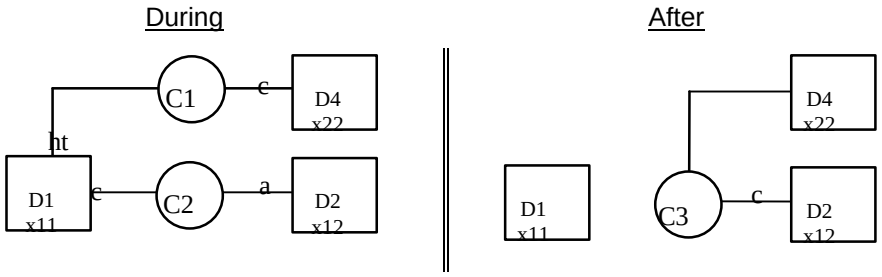
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
Extension 11 presses TRANSFER button to transfer call with Extension 22 to Extension 12. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
Extension 11 dials Extension 12, hears busy and presses transfer. Call automatically camps on to Extension 12. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = /* none */ ***transferredConnections*** device after 22 D4C2		***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = /* none */ ***transferredConnections*** device after 22 D4C2
Transferred call is now camped on for Extension 12.		
	User at Extension 12 finishes with other call. ***CSTAReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C3 ***releasingDevice*** = 12 ***cause*** = EC_NONE	
	CSTADeliveredEvent ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22
	Application answers camped-on call. ***cstaAnswerCall( )*** ***alertingCall*** = D2C2	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAEstablishedEvent ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER

NOTE:
The ***callingDevice*** parameter in the ***CSTADeliveredEvent*** and ***CSTAEstablishedEvent*** contains the transfer source (in this case an internal extension, but in the case of an outside call, the trunk identifier or ANI/ICLID).

Camp On Completes Transfer to Non-Busy Extension; Destination Answers

This use of Camp On extends causes the call to use the Camp On return timer instead of the transfer return timer before returning to the transfer originator.

Extension 11 has placed a call with Extension 22 on hold for transfer (manually or with an application using ***cstaConsultationCall( )***). The user at Extension 11 transfers the call to Extension 12 (manually or using ***cstaTransferCall( )***). The transferred call camps on for Extension 12. Extension 12 answers it. The event flow shows Extension 11 manually making the transfer and begins with the first press of the TRANSFER button to place the call with Extension 22 on hold-for-transfer.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
Extension 11 presses TRANSFER button to transfer call with Extension 22 to Extension 12. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	
The “During” illustration in the figure above applies at this point.		
Extension 11 invokes Camp On. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D2C3 22 D4C3	Extension 11 invokes Camp On. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D2C3 22 D4C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D1C3 22 D4C3
	Application answers camped-on call. ***cstaAnswerCall( )*** ***alertingCall*** = D2C3	
	CSTAAnswerCallConfEvent	
	CSTAEstablishedEvent ***establishedConnection*** = D2C3 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C3 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_NONE

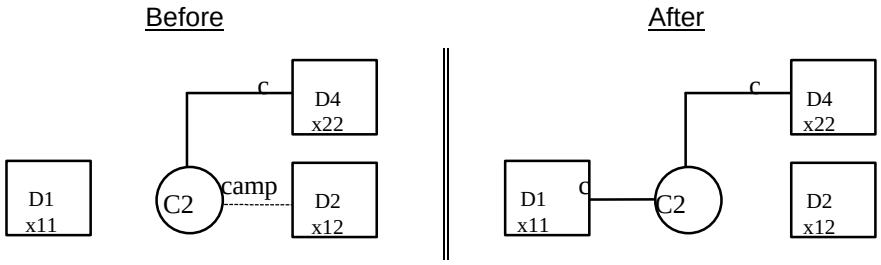
MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
Extension 11 presses TRANSFER button to transfer call with Extension 22 to Extension 12. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
The “During” illustration in the figure above applies at this point.		
Extension 11 invokes Camp On. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D2C2 22 D4C2	Extension 11 invokes Camp On. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D2C2 22 D4C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D1C2 22 D4C2
	Application answers camped-on call. ***cstaAnswerCall( )*** ***alertingCall*** = D2C2	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAEstablishedEvent ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
Extension 11 presses TRANSFER button to transfer call with Extension 22 to Extension 12. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTADeliveredEvent ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C2 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
The “During” illustration in the figure above applies at this point.		
Extension 11 invokes Camp On. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D2C2 22 D4C2	Extension 11 invokes Camp On. ***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D2C2 22 D4C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 12 ***transferredConnections*** device after 12 D1C2 22 D4C2
	Application answers camped-on call. ***cstaAnswerCall( )*** ***alertingCall*** = D2C2	
	CSTAAnswerCallConfEvent	
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAEstablishedEvent ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C2 ***answeringDevice*** = 12 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_TRANSFER

Camp On Return with Answer



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
User at Extension 11 has transferred a call to Extension 12, where it is camped On. The Camp On return timer now causes that call to re-alert at Extension 11.		
Camp On return timer expires. ***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22
Application uses answer to reconnect to returning camp on call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C2		
CSTAAnswerCallConfEvent		
Established event also indicates successful completion. ***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_NONE ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
User at Extension 11 has transferred a call to Extension 12, where it is camped On. The Camp On return timer now causes that call to re-alert at Extension 11.		
Camp On return timer expires.		
CSTADeliveredEvent ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22
Application uses answer to reconnect to returning camp on call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C2		
CSTAAnswerCallConfEvent		
CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11 Established event also indicates successful completion.		
CSTAEstablishedEvent ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 11 ***cause*** = EC_RECALL ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.1 and later

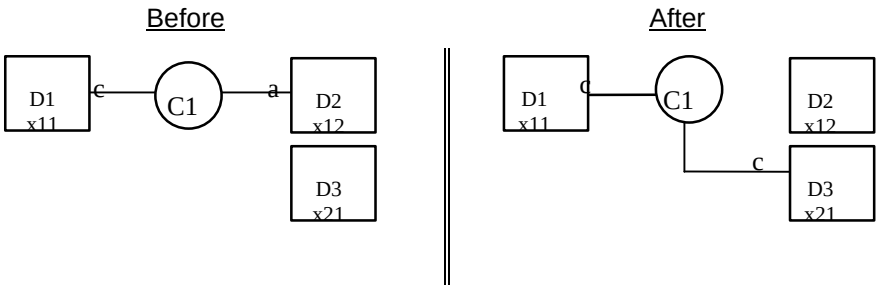
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 22
User at Extension 11 has transferred a call to Extension 12, where it is camped On. The Camp On return timer now causes that call to re-alert at Extension 11.		
Camp On return timer expires. ***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***connection*** = D1C2 ***alertingDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22
Application uses answer to reconnect to returning camp on call. ***cstaAnswerCall( )*** ***alertingCall*** = D1C2		
CSTAAnswerCallConfEvent		
CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11 Established event also indicates successful completion.		
CSTAEstablishedEvent ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_RECALL	***CSTAEstablishedEvent*** ***connection*** = D1C2 ***answeringDevice*** = 11 ***callingDevice*** = 22 ***calledDevice*** = 12-* ***cause*** = EC_RECALL
CSTAConnectionClearedEvent ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE

Coverage

Coverage allows a call ringing at one extension (a sender) to ring at another extension (or extensions) at the same time and to be answered at either extension.

Coverage; Receiver Answers

A call placed from Extension 11 to Extension 12. Extension 12 is covered by Extension 21. The call is answered at Extension 21 using a cover button.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	An application monitoring a receiver does not get a ***CSTADeliveredEvent*** when a call alerts on a COVER button.
		User at Extension 21 manually answers the call.
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
	When receiver uses COVER button to answer call, the call is removed from Extension 12.	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
		Call delivered to Extension 21.
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12
		User at Extension 21 manually answers the call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12
	When receiver uses COVER button to answer call, the call is removed from Extension 12.	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	

MERLIN MAGIX R2.1 and later

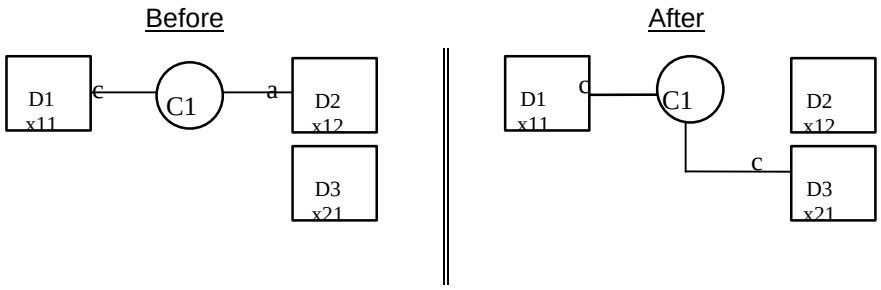
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
		Call delivered to Extension 21.
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD
		User at Extension 21 manually answers the call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD
	When receiver uses COVER button to answer call, the call is removed from Extension 12.	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	

Coverage; Calling Group is Receiver

When a calling group is a coverage receiver, the call is removed from the sending extension when the call leaves the coverage group and is sent to an available calling group member.

The event flow is similar to that in the previous event flow, but the ***CSTA-ConnectionClearedEvent*** flows because the connection is removed from the sending extension.

In the event flow below, Extension 21 is a coverage group member that becomes available.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	
Call covers to receiving coverage group. Appearance remains at sending extension.		
Group member Extension 21 becomes available and call is sent to Extension 21.		
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE
Extension 21 answers coverage call.		
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE		***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE

MERLIN MAGIX R2.0

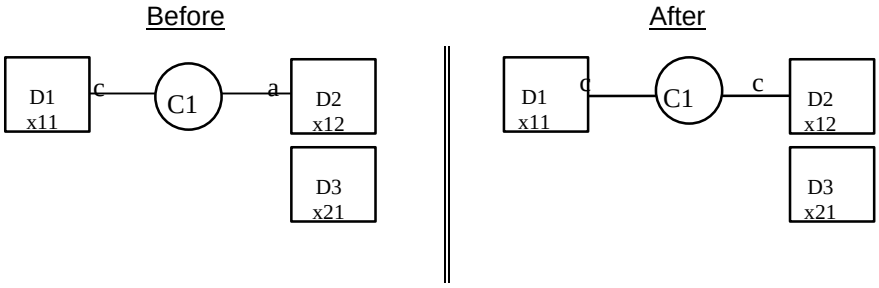
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent	***CSTADeliveredEvent***	
connection = D2C1	***connection*** = D2C1	
alertingDevice = 12	***alertingDevice*** = 12	
callingDevice = 11	***callingDevice*** = 11	
calledDevice = 12	***calledDevice*** = 12	
cause = EC_NEW_CALL	***cause*** = EC_NEW_CALL	
	Call covers to receiving coverage group. Appearance remains at sending extension.	
CSTAQueuedEvent	***CSTAQueuedEvent***	
queuedConnection = D4C1	***queuedConnection*** = D4C1	
queue = <DGC>	***queue*** = <DGC>	
callingDevice = 11	***callingDevice*** = 11	
calledDevice = <DGC>	***calledDevice*** = <DGC>	
numberQueued = 1	***numberQueued*** = 1	
cause = EC_NONE	***cause*** = EC_NONE	
Private Data	***Private Data***	
originalCallInfo	***originalCallInfo***	
callingDevice = 11	***callingDevice*** = 11	
calledDevice = 12	***calledDevice*** = 12	
	Group member Extension 21 becomes available and call is sent to Extension 21.	
		CSTALoggedOnEvent
		agentDevice = 21
		agentID = 21
		agentGroup = <DGC>
CSTADivertedEvent	***CSTADivertedEvent***	***CSTADivertedEvent***
connection = D4C1	***connection*** = D4C1	***connection*** = D4C1
divertingDevice = <DGC>	***divertingDevice*** = <DGC>	***divertingDevice*** = <DGC>
newDestination = 21	***newDestination*** = 21	***newDestination*** = 21
cause = EC_REDIRECTED	***cause*** = EC_REDIRECTED	***cause*** = EC_REDIRECTED
CSTAConnectionClearedEvent	***CSTAConnectionClearedEvent***	
droppedConnection = D2C1	***droppedConnection*** = D2C1	
releasingDevice = 12	***releasingDevice*** = 12	
cause = EC_NONE	***cause*** = EC_NONE	
CSTADeliveredEvent		***CSTADeliveredEvent***
connection = D3C1		***connection*** = D3C1
alertingDevice = 21		***alertingDevice*** = 21
callingDevice = 11		***callingDevice*** = 11
calledDevice = 21		***calledDevice*** = 21
lastRedirectionDevice = <DGC>		***lastRedirectionDevice*** = <DGC>
cause = EC_REDIRECTED		***cause*** = EC_REDIRECTED
Private Data		***Private Data***
originalCallInfo		***originalCallInfo***
callingDevice = 11		***callingDevice*** = 11
calledDevice = 12		***calledDevice*** = 12
	Extension 21 answers coverage call.	
CSTAEstablishedEvent		***CSTAEstablishedEvent***
establishedConnection = D3C1		***establishedConnection*** = D3C1
answeringDevice = 21		***answeringDevice*** = 21
callingDevice = 11		***callingDevice*** = 11
calledDevice = 21		***calledDevice*** = 21
lastRedirectionDevice = <DGC>		***lastRedirectionDevice*** = <DGC>
cause = EC_REDIRECTED		***cause*** = EC_REDIRECTED
Private Data		***Private Data***
originalCallInfo		***originalCallInfo***
callingDevice = 11		***callingDevice*** = 11
calledDevice = 12		***calledDevice*** = 12

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call delivered to Extension 12.		
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
Call covers to receiving coverage group. Appearance remains at sending extension.		
CSTAQueuedEvent ***queuedConnection*** = D4C1 ***queue*** = <DGC> ***callingDevice*** = 11 ***calledDevice*** = 12 ***numberQueued*** = 1 ***cause*** = EC_NONE	***CSTAQueuedEvent*** ***queuedConnection*** = D4C1 ***queue*** = <DGC> ***callingDevice*** = 11 ***calledDevice*** = 12 ***numberQueued*** = 1 ***cause*** = EC_NONE	
Group member Extension 21 becomes available and call is sent to Extension 21.		
		CSTALoggedOnEvent ***agentDevice*** = 21 ***agentID*** = 21 ***agentGroup*** = <DGC>
CSTADivertedEvent ***connection*** = D4C1 ***divertingDevice*** = <DGC> ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C1 ***divertingDevice*** = <DGC> ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D4C1 ***divertingDevice*** = <DGC> ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = <DGC> ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = <DGC> ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = <DGC> ***cause*** = EC_REDIRECTED
Extension 21 answers coverage call.		
		CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = <DGC> ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = <DGC> ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = <DGC> ***cause*** = EC_REDIRECTED

Coverage; Sender Answers

A call placed from Extension 11 to Extension 12. Extension 12 is covered by Extension 21. The call is answered at Extension 11 using a SA button.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	An application monitoring a receiver does not get a ***CSTADeliveredEvent*** when a call alerts on a COVER button.
User at Extension 12 answers the call. An application could use ***cstaAnswerCall( )*** as well.		
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	

MERLIN MAGIX R2.0

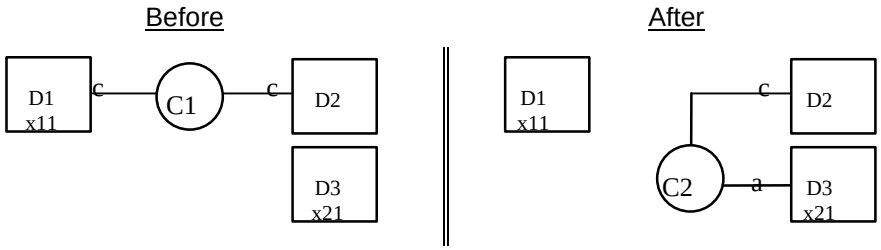
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
		Call delivered to Extension 21
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12
User at Extension 12 answers the call. An application could use ***cstaAnswerCall( )*** as well.		
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Call delivered to Extension 12.	
CSTADeliveredEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
		Call delivered to Extension 21
CSTADeliveredEvent ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	***CSTADeliveredEvent*** ***connection*** = D3C1 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = 12 ***cause*** = EC_CALL_FORWARD
User at Extension 12 answers the call. An application could use ***cstaAnswerCall( )*** as well.		
	CSTANotReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_NONE
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NONE	

Direct Voice Mail – Transfer and Dial Feature Code

D1 is on a call. D1 pressess the transfer button and dials #5612 (#56 is the feature code for Direct Voice Mailand 12 is the extension od the station voice mail is for). The call is transferred to D3, which is a Voice Mail port. D4 (not shown) is ts extension 12 and D5 (not shown) is the DGC Queue for Voice Mail.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 21
User at Extension 11 presses TRANSFER, then dials #5612 to go to Voice Mail for 12	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
CSTADeliveredEvent ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NONE	
cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2	
CSTADeliveredEvent ***connection*** = D3C3 ***alertingDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_NONE	
CSTATransferCallConfEvent ***newCall*** = D3C3	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device: 12, 21 before: D2C1, D3C2 after: D2C3, D3C3	

MERLIN MAGIX R2.0

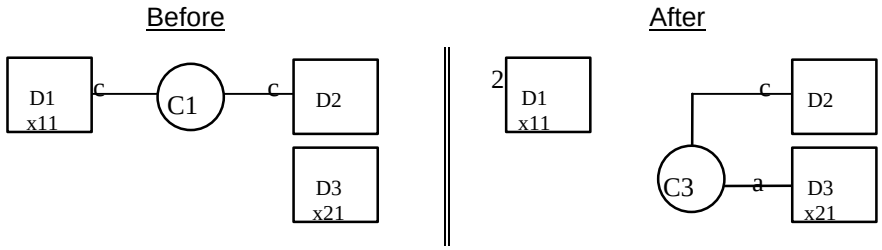
Stream Monitoring Extension 11	Stream Monitoring Extension 21
User at Extension 11 presses TRANSFER, then dials #5612 to go to Voice Mail for 12	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
CST ***ADivertedEvent*** ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
CSTADeliveredEvent ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED
	CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED
cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2 ***CSTATransferCallConfEvent*** ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device after 12 D2C2 21 D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device after 12 D2C2 21 D3C2

MERLIN MAGIX R2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 21
User at Extension 11 presses TRANSFER, then dials #5612 to go to Voice Mail for 12	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
CSTAQueuedEvent ***queuedConnection*** = D1C1 ***queue*** = QDGCQ ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_CALL_FORWARD	
CSTADivertedEvent ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
CSTADeliveredEvent ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED
	CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED
cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2 ***CSTATransferCallConfEvent*** ***newCall*** = D1C2	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C2 21 / D3C2

Direct Voice Mail – Use Feature or Programmed Button

D1 is on a call. D1 pressess the Direct Voice Mail button and then dials 12 (who is the coverage sender). D3 is a member of the calling Group (DGC) proving coverage.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
User at Extension 11 presses DVM BUTTON, then dials 12 to go to Voice Mail for 12	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
cstaTransferCall( ) ***heldCall*** = D1C1 ***activeCall*** = D1C2	
Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
CSTATransferCallConfEvent ***newCall*** = D3C3	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / before / after 12 / D2C1 / D2C3 21 / D3C2 / D3C3	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after 12 / D2C3 21 / D3C3

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 21
User at Extension 11 presses DVM button then dials 12 to go to Voice Mail fot extension 12	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
CSTATransferCallConfEvent ***newCall*** = D3C3	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after QDGCQ / D5C2 22 / D2C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device / after QDGCQ / D5C2 22 / D2C2
CSTAConnectionClearedEvent ***droppedConnection*** = D4C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D4C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE
CSTADivertedEvent ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
CSTADeliveredEvent ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED	***CSTADeliveredEvent*** ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED
	CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED
CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = D1 ***cause*** = EC_CALL_CANCELLED	

MERLIN MAGIX R2.1 and later

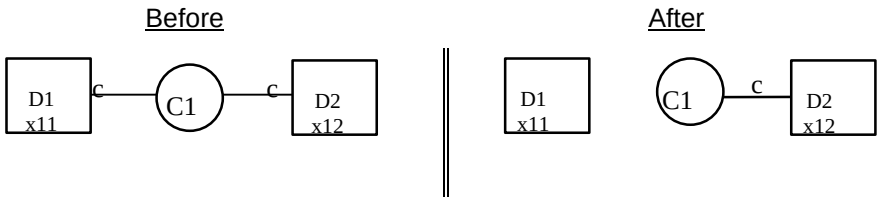
Stream Monitoring Extension 11	Stream Monitoring Extension 21
User at Extension 11 presses DVM button then dials 12 to go to Voice Mail fot extension 12	
CSTAHeldEvent ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2	
CSTAQueuedEvent ***queuedConnection*** = D5C2 ***queue*** = QDGCQ ***callingDevice*** = 11 ***calledDevice*** = 12 ***lastRedierctionDevice*** = 12 ***cause*** = EC_CALL_FORWARD	
CSTATransferCallConfEvent ***newCall*** = D3C3	
CSTATransferredEvent ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device after QDGCQ D5C2 22 D2C2	***CSTATransferredEvent*** ***primaryOldCall*** = D1C1 ***secondaryOldCall*** = D1C2 ***transferringDevice*** = 11 ***transferredDevice*** = 21 ***transferredConnections*** device after QDGCQ D5C2 22 D2C2
CSTAConnectionClearedEvent ***droppedConnection*** = D4C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D4C2 ***releasingDevice*** = 12 ***cause*** = EC_NONE
CSTADivertedEvent ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED	***CSTADivertedEvent*** ***connection*** = D5C2 ***divertingDevice*** =QDGC ***newDestination*** = 21 ***cause*** = EC_REDIRECTED
CSTADeliveredEvent ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_REDIRECTED ***lastRedirectionDevice*** = QDGC ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22	***CSTADeliveredEvent*** ***alertingConnection*** = D3C2 ***alertingDevice*** = 21 ***callingDevice*** = 22 ***calledDevice*** = 12 ***cause*** = EC_REDIRECTED ***lastRedirectionDevice*** = QDGC ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 22
	CSTANotReadyEvent ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 22 ***calledDevice*** = 12 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C2 ***answeringDevice*** = 21 ***callingDevice*** = 22 ***calledDevice*** = 12 ***lastRedirectionDevice*** = QDGC ***cause*** = EC_REDIRECTED

Stream Monitoring Extension 11	Stream Monitoring Extension 21
CSTAConnectionClearedEvent	***CSTAConnectionClearedEvent***
droppedConnection = D1C2	***droppedConnection*** = D1C2
releasingDevice = D1	***releasingDevice*** = D1
cause = EC_CALL_CANCELLED	***cause*** = EC_CALL_CANCELLED

Park

Parking a Call

The user at Extension 12 parks a call with Extension 11.



MERLIN MAGIX Release 2.0

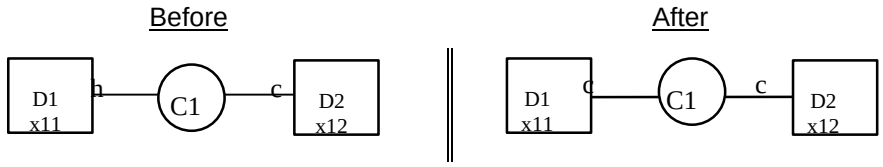
Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extensions 11 and 12 are connected on call C1.	
Extension 11 presses TRANSFER button. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2 Extension 11 dials "11", and presses TRANSFER again to complete park operation.	
CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	
The call is parked.	

MERLIN MAGIX Release 2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extensions 11 and 12 are connected on call C1.	
Extension 11 presses TRANSFER button. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11 ***cause*** = EC_TRANSFER
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2 Extension 11 dials "11", and presses TRANSFER again to complete park operation.	
CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED	
The call is parked.	

Reconnecting to Parked Call Before Timer Expires

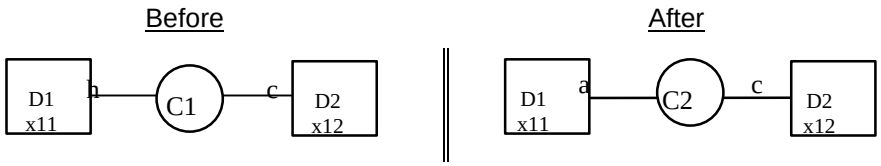
Extension 11 has parked a call (as in the scenario above) and now uses ***cstaRetrieveCall( )*** to access that call. Manual operation to access that call will also cause the ***CSTARetrievedEvent*** to flow.



Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 has parked call C1 (to which Extension 12 is connected).	
cstaRetrieveCall( ) ***heldCall*** = D1C1	
CSTARetrieveCallConfEvent	
CSTARetrievedEvent ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

Parked Call Returns

When a call is parked and remains parked long enough that the Park Return Timer expires, then the held connection for the call is cleared at the parking party and the parked call returns to the parking party and alerts.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 has parked call C1 (to which Extension 12 is connected) and the Park Return Tmer expires.	
CSTAConnectionClearedEvent	***CSTAConnectionClearedEvent***
droppedConnection = D1C1	***droppedConnection*** = D1C1
releasingDevice = 11	***releasingDevice*** = 11
cause = EC_CALL_CANCELLED	***cause*** = EC_CALL_CANCELLED
CSTADeliveredEvent	***CSTADeliveredEvent***
connection = D1C1	***connection*** = D1C1
alertingDevice = 11	***alertingDevice*** = 11
callingDevice = 12	***callingDevice*** = 12
calledDevice = 11	***calledDevice*** = 11
cause = EC_NONE	***cause*** = EC_NONE
Private Data	***Private Data***
originalCallInfo	***originalCallInfo***
callingDevice = 11	***callingDevice*** = 11
calledDevice = 12	***calledDevice*** = 12

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12
Extension 11 has parked call C1 (to which Extension 12 is connected) and the Park Return Tmer expires.	
CSTAConnectionClearedEvent	***CSTAConnectionClearedEvent***
droppedConnection = D1C1	***droppedConnection*** = D1C1
releasingDevice = 11	***releasingDevice*** = 11
cause = EC_CALL_CANCELLED	***cause*** = EC_CALL_CANCELLED
CSTADeliveredEvent	***CSTADeliveredEvent***
connection = D1C1	***connection*** = D1C1
alertingDevice = 11	***alertingDevice*** = 11
callingDevice = 12	***callingDevice*** = 12
calledDevice = 11	***calledDevice*** = 11
cause = EC_RECALL	***cause*** = EC_RECALL

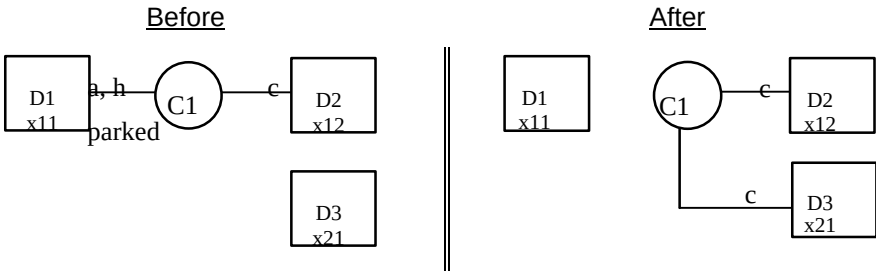
NOTE:
In some circumstances the call identifiers C1 and C2 will be the same; in others, they will be different.

Pickup

A user may use the pickup feature to pickup a parked call, an alerting call, or a held call.

Pickup Parked, Alerting, or Held Internal Call

Extension 11 has an alerting, held, or parked call C1 with Extension 12. Extension 21 will use the Pickup feature to pickup that call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
		Extensions 21 goes off-hook to pickup call and dials feature code. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D3C2
The call pickup feature now connects the call to the party that is picking up.		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	
		After feature access, the activating connection clears. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_NONE
		No events flow on this monitor for the call that has been picked up.
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	Extension 12 hangs up. No event on this monitor.

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
		Extensions 21 goes off-hook to pickup call and dials feature code. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
		CSTAServiceInitiatedEvent ***initiatedConnection*** = D3C2
		After feature access, the activating connection clears. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_CALL_PICKUP ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_CALL_PICKUP ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 21 ***cause*** = EC_CALL_PICKUP ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 12 ***calledDevice*** = 11
The call pickup feature now connects the call to the party that is picking up.		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE
	Extension 12 hangs up. ***CSTAReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
		CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.1 and later

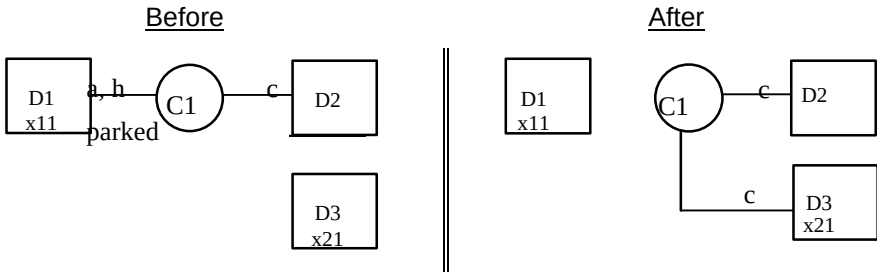
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
		Extensions 21 goes off-hook to pickup call and dials feature code. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
		CSTAServiceInitiatedEvent ***initiatedConnection*** = D3C2
		After feature access, the activating connection clears. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_CALL_PICKUP	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_CALL_PICKUP	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 12 ***calledDevice*** = 11 ***cause*** = EC_CALL_PICKUP
The call pickup feature now connects the call to the party that is picking up.		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE
	Extension 12 hangs up. ***CSTAReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
		CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

NOTE:

There ***CSTAConnectionClearedEvent*** will have a cause of EC_NONE when the picked up call is alerting and EC_CALL_CANCELLED when the call is parked or on hold.

Pickup Parked, Alerting, or Held External Call

Extension 11 has an alerting, held, or parked call C1 with an outside party. Extension 21 will use the Pickup feature to pickup that call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 21
	Extensions 21 goes off-hook to pickup call and dials feature code. ***CSTAServiceInitiatedEvent*** ***initiatedConnection*** = D3C2
The call pickup feature now connects the call to the party that is picking up.	
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	
	After feature access, the activating connection clears. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_NONE
	No events flow on this monitor for the call that has been picked up.
	Extension 12 hangs up. No event on this monitor.

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	**Stream Monitoring Extension 21**
	Extensions 21 goes off-hook to pickup call and dials feature code. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D3C2
	After feature access, the activating connection clears. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
The call pickup feature now connects the call to the party that is picking up.	
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_PICKUP ***Private Data*** ***trunkUsed*** = <trunk>	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_PICKUP ***Private Data*** ***trunkUsed*** = <trunk>
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE

MERLIN MAGIX R2.1 and later

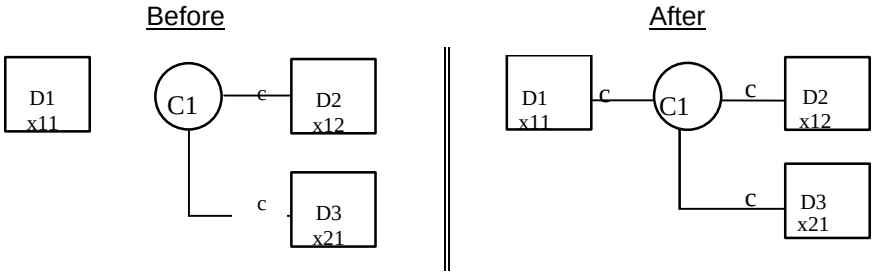
Stream Monitoring Extension 11	Stream Monitoring Extension 21
	Extensions 21 goes off-hook to pickup call and dials feature code. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
	CSTAServiceInitiatedEvent ***initiatedConnection*** = D3C2
	After feature access, the activating connection clears. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D3C2 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED
The call pickup feature now connects the call to the party that is picking up.	
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_PICKUP ***Private Data*** ***trunkUsed*** = <trunk>	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_PICKUP ***Private Data*** ***trunkUsed*** = <trunk>
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE

NOTE:
There ***CSTAConnectionClearedEvent*** will have a cause of EC_NONE when the picked up call is alerting and EC_CALL_CANCELLED when the call is parked or on hold.

Service Observing (MERLIN MAGIX Release 2.0 and Later)

A user may observe an active call.

Extension 12 has an active call. Station 11 observes the call.



Observer Starts Observing Before Call

MERLIN MAGIX Release 2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Station 11 observes the call		
CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11		
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 11 ***cause*** = EC_SILENT_MONITOR		
Station 11 stops observing the call		
CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_SILENT_MONITOR		

MERLIN MAGIX Release 2.1 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Station 11 observes the call		
CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11		
CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = 21 ***calledDevice*** = 12 ***cause*** = EC_SILENT_MONITOR		
Station 11 stops observing the call		
CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_SILENT_MONITOR		

Observer Starts Observing After Call Exists

MERLIN MAGIX Release 2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	Station 11 observes the call	
CSTANotReadyEvent ***agentDevice*** = 11 ***agentID*** = 11		
CSTAServiceInitiatedEvent ***initiatedConnection*** = D1C2		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C2 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED		
A ***CSTAEstablishedEvent*** is not provided when the Service Observer is added to the observed call if that call existed prior to Service Observing feature activation..		

Shared System Access Event Flows

An understanding of Shared System Access (SSA) terminology and its relationship to the TSAPI model will help in understanding the TSAPI event flows that occur when connections interact with Shared System Access buttons.

An SSA button on an extension provides an appearance of an SA button at another extension. Using SSA buttons may cause connections at the SA button to transition into ***associative states*** that the MERLIN LEGEND and MERLIN MAGIX switches term ***associative active*** and ***associative held***. The MERLIN LEGEND and MERLIN MAGIX switches make a distinction between the TSAPI connected and held states and the associative states (which TSAPI does not model).

In MERLIN LEGEND and MERLIN MAGIX terminology, when a call is alerting at an SA button and a user at another station presses an SSA button and connects to that call, that user has ***answered*** the call. The state of the call at the SA button changes to associative active. The state of the call at the SSA is connected (a TSAPI state). Thus, an application monitoring an extension where an SSA answers a call will receive further events about the call.

When a call is active at an SA button and a user at another station presses an SSA button and connects to that call, the user ***bridged*** onto the call. The state of the call at the SA button remains active. The state of the call at the SSA is ***bridged*** (not a TSAPI state). Thus, an application monitoring an extension where an SSA bridges onto a call will not receive further events about the call.

Depending on whether an SSA user answers a call or bridges onto a call, event flows will differ for an application monitoring the extension with the SSA button.

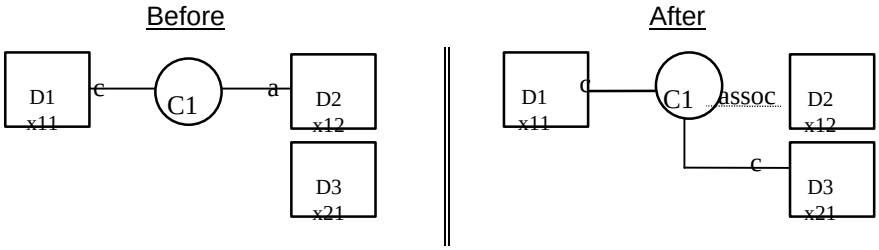
The following rules govern event flows when SSA buttons interact with calls:

- The MERLIN LEGEND and MERLIN MAGIX switches consider connections that transition into the associative or bridged states as having left the defined TSAPI model. As a result, they are considered to have been cleared from the device where this transition occurred, and any applications monitoring the device with the SA button where this occurs will receive a ***CSTAConnectionClearedEvent*** event the first time a connection transitions into an associative state.
  - Once the MERLIN LEGEND or MERLIN MAGIX system supplies a ***CSTAConnectionClearedEvent*** when a connection transitions into an associative state at a device, the system will not supply any further events for that connection at that device. The device may reconnect to the call and the system will not supply any further events. (Note that the call is still in an associative state.)
  - An application monitoring an extension where an SSA ***answers a call*** will receive events for that call (so long as the call does not enter an associative state due to some later feature interaction).

- An application monitoring an extension where an SSA ***bridges onto a call*** will not receive events for that call.
- Applications monitoring an extension having an SSA button do not receive any events about an incoming call on the corresponding SA button unless a user at the extension with the SSA button uses the SSA button to ***answer*** the call. Of special interest is the fact that such an application will not receive a ***CSTADeliveredEvent***. Thus, the application cannot be aware of the call on the corresponding SA button and the user must manually answer the call on the SSA button.

SSA Button Answers Alerting Call; Call Activity Follows on SA and SSA

A call placed from Extension 11 to Extension 12 is alerting at Extension 12. A user at Extension 21 uses an SSA button (for Extension 12) to ***answer*** the call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
A call from Extension 11 is delivered to Extension 12 (before diagram). ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	
		User at Extension 21 uses SSA to answer call.
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21
	At this point, connection D2C1 goes to associative active state.	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
	The connection still appears at Extension 12 in the associative active state.	
Call activity at Extension 11 or 21 will cause events to flow here.	Call activity at Extension 11 or 21 will ***not*** cause events to flow here.	Call activity at Extension 11 or 21 will cause events to flow here.
No event flows here.	Extension 12 manually bridges back onto call.	No event flows here.
No event flows here.	Extension 12 hangs up. The connection still appears at Extension 12 in the associative active state.	No event flows here.
Extension 11 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
Extension 11 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11		***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
		Extension 21 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D3C1 ***holdingDevice*** = 21
		Extension 21 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D3C1 ***retrievingDevice*** = 21
Extension 11 hangs up the call. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE
		CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.0

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
A call from Extension 11 is delivered to Extension 12 (before diagram). ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
		User at Extension 21 uses SSA to answer call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 21 ***cause*** = EC_NEW_CALL ***Private Data*** ***originalCallInfo*** ***callingDevice*** = 11 ***calledDevice*** = 12
	At this point, connection D2C1 goes to associative active state.	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
	The connection still appears at Extension 12 in the associative active state.	
Call activity at Extension 11 or 21 will cause events to flow here.	Call activity at Extension 11 or 21 will ***not*** cause events to flow here.	Call activity at Extension 11 or 21 will cause events to flow here.

MERLIN MAGIX R2.0, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
No event flows here.	Extension 12 manually bridges back onto call. ***CSTANotReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	No event flows here.
No event flows here.	Extension 12 hangs up. The connection still appears at Extension 12 in the associative active state. ***CSTAReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	No event flows here.
Extension 11 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
Extension 11 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11		***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11
		Extension 21 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D3C1 ***holdingDevice*** = 21
		Extension 21 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D3C1 ***retrievingDevice*** = 21
Extension 11 hangs up the call ***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE
		CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

MERLIN MAGIX R2.1 and later

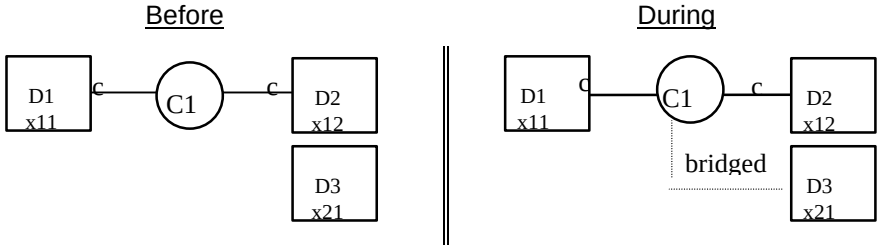
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
A call from Extension 11 is delivered to Extension 12 (before diagram). ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
		User at Extension 21 uses SSA to answer call. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
CSTAEstablishedEvent ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D3C1 ***answeringDevice*** = 21 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL
	At this point, connection D2C1 goes to associative active state.	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
	The connection still appears at Extension 12 in the associative active state.	
Call activity at Extension 11 or 21 will cause events to flow here.	Call activity at Extension 11 or 21 will ***not*** cause events to flow here.	Call activity at Extension 11 or 21 will cause events to flow here.

MERLIN MAGIX R2.1, continued

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
No event flows here.	Extension 12 manually bridges back onto call. ***CSTANotReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	No event flows here.
No event flows here.	Extension 12 hangs up. The connection still appears at Extension 12 in the associative active state. ***CSTAReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	No event flows here.
Extension 11 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11		***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
Extension 11 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11		***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11
		Extension 21 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D3C1 ***holdingDevice*** = 21
		Extension 21 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D3C1 ***retrievingDevice*** = 21
Extension 11 hangs up the call ***CSTAReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11		
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE		***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE
		CSTAConnectionClearedEvent ***droppedConnection*** = D3C1 ***releasingDevice*** = 21 ***cause*** = EC_CALL_CANCELLED

SSA Button Bridges onto Call at SA Button; Call Activity Follows on SA and SSA

Extensions 11 and 12 are connected on a call. A user at Extension 21 uses an SSA button (for Extension 12) to ***bridge*** onto the call.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
CSTAEstablishedEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	A call from Extension 11 is answered at Extension 12 (before diagram). ***CSTAEstablishedEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	
		User at Extension 21 uses SSA to bridge onto call. Connection at Extension 21 is in bridged state.
The “During” illustration in the figure above applies at this point.		
		The connection appears in the bridged state on the SSA.
Call activity at Extension 11 will cause events to flow here.	Call activity at Extension 11 will cause events to flow here. Call activity at Extension 21 will ***not*** cause events to flow here.	Call activity at Extension 11 or 21 will ***not*** cause events to flow here.
Extension 11 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
Extension 11 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	
		Extension 21 puts the call on hold.
		Extension 21 retrieves the call.
Extension 11 hangs up the call. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	
	CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	

MERLIN MAGIX R2.0 and later

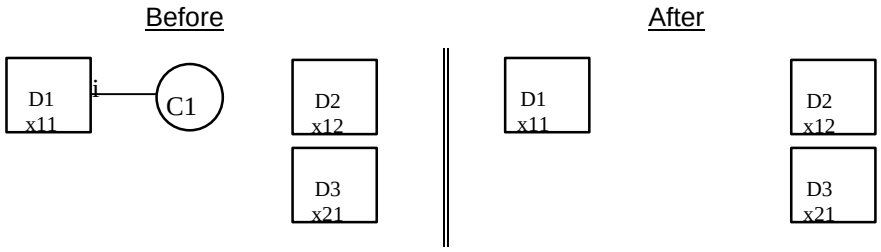
Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
	A call from Extension 11 is answered at Extension 12 (before diagram). ***CSTANotReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	
CSTAEstablishedEvent ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	
		User at Extension 21 uses SSA to bridge onto call. Connection at Extension 21 is in bridged state. ***CSTANotReadyEvent*** ***agentDevice*** = 21 ***agentID*** = 21
The “During” illustration in the figure above applies at this point.		
		The connection appears in the bridged state on the SSA.
Call activity at Extension 11 will cause events to flow here.	Call activity at Extension 11 will cause events to flow here. Call activity at Extension 21 will ***not*** cause events to flow here.	Call activity at Extension 11 or 21 will ***not*** cause events to flow here.
Extension 11 puts the call on hold. ***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11	
Extension 11 retrieves the call. ***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11	
		Extension 21 puts the call on hold.
		Extension 21 retrieves the call.
Extension 11 hangs up the call. ***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_NONE	
CSTAReadyEvent ***agentDevice*** = 11 ***agentID*** = 11	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	

CAUTION:

Because there are no events that reflect connections being in non-TSAPI states, an application here does not receive events about the bridged connection. Notice that the application cannot assume that if, according to the events, only one connection remains, that the call will be torn down. There may be another connection in a non-TSAPI state.

Call Activity on an SA button Where There is an Associated SSA Button at Another Extension (that has Not Answered or Bridged)

A user at Extension 12 answers a call from Extension 11. Extension 21 has an SSA button (for Extension 12).



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	No event on this monitor.
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	User at Extension 12 answers call. ***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12	No event on this monitor.
	cstaClearConnection( ) ***call*** = D2C1	
	CSTAClearConnectionConfEvent	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE	No event on this monitor.
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED		

MERLIN MAGIX R2.0 and later

Stream Monitoring Extension 11	Stream Monitoring Extension 12	Stream Monitoring Extension 21
Call delivered to Extension 12. ***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	No event on this monitor.
	User at Extension 12 answers call. ***CSTANotReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12	
CSTAEstablishedEvent ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	***CSTAEstablishedEvent*** ***establishedConnection*** = D2C1 ***answeringDevice*** = 12 ***callingDevice*** = 11 ***calledDevice*** = 12 ***cause*** = EC_NEW_CALL	No event on this monitor.
	cstaClearConnection( ) ***call*** = D2C1	
	CSTAClearConnectionConfEvent	
CSTAConnectionClearedEvent ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_CALL_CANCELLED	No event on this monitor.
	CSTAReadyEvent ***agentDevice*** = 12 ***agentID*** = 12	
CSTAConnectionClearedEvent ***droppedConnection*** = D1C1 ***releasingDevice*** = 11 ***cause*** = EC_CALL_CANCELLED		

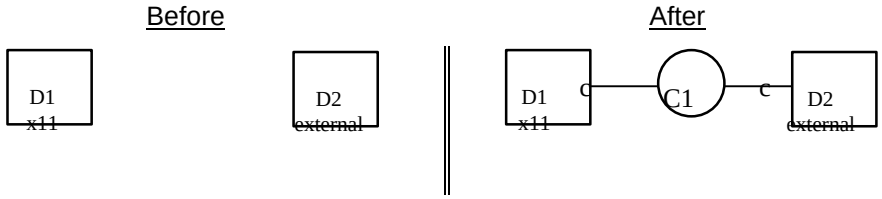
Direct Facility Termination Event Flows

Direct Facility Termination (DFT) or Personal Line buttons are treated in the same way as SSA buttons. The various terms (answer, bridge) and associative states also apply to the interaction between DFT and SA buttons. The rules governing the events that flow when interactions occur (seen in the SSA section) also apply to DFTs.

One difference between DFT and SSA buttons is that a DFT button may appear at the same extension as the SA button that it is interacting with.

Incoming Call on DFT; Call Activity Follows

Extension 11 receives an incoming trunk call on a DFT. Extension 11 answers the call on the DFT. Because the DFT answers the call, an application monitoring Extension 11 will see call events as call activity occurs on that DFT. The application cannot control the call on the DFT in releases prior to MERLIN MAGIX 2.0. Beginning in MERLIN MAGIX 2.0, the call can be controlled via an application.



MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

Activity	Stream Monitoring Extension 11
Extension 11 receives an incoming trunk call on a DFT.	
User at Extension 11 answers call.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT>
User at Extension 11 presses HOLD.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
User at Extension 11 reconnects to the call.	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

MERLIN MAGIX R2.0

Activity	Stream Monitoring Extension 11
Extension 11 receives an incoming trunk call on a DFT.	***CSTADeliveredEvent*** ***alertingConnection*** = D1C1 ***alertinggDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>
User at Extension 11 answers call.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>
User at Extension 11 presses HOLD.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
User at Extension 11 reconnects to the call.	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

MERLIN MAGIX R2.1 and later

Activity	Stream Monitoring Extension 11
Extension 11 receives an incoming trunk call on a DFT.	***CSTADeliveredEvent*** ***alertingConnection*** = D1C1 ***alertinggDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>
User at Extension 11 answers call.	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11
	CSTAEstablishedEvent ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/EXT> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>
User at Extension 11 presses HOLD.	***CSTAHeldEvent*** ***heldConnection*** = D1C1 ***holdingDevice*** = 11
User at Extension 11 reconnects to the call.	***CSTARetrievedEvent*** ***retrievedConnection*** = D1C1 ***retrievingDevice*** = 11

DFT Bridges onto Call at SA; Call Activity Follows

A user at Extension 11 answers a call on an SA (In MERLIN MAGIX 2.0, this will hold true for a DFT). Once the call is answered, a DFT at Extension 12 bridges onto the call.

MERLIN LEGEND R5.0, 6.0, 6.1, 7.0 & MERLIN MAGIX R1.0 and 1.5

	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Call delivered to Extension 11.	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/TRUNK>	
User at Extension 11 answers call.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/TRUNK>	
DFT button bridges onto call. The call is active at Extension 11 and bridged at Extension 12.		
	Call activity at Extension 11 will cause events to flow on this stream.	Call activity at Extension 11 will not cause any events to flow on this stream.

MERLIN MAGIX R2.0

	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Call delivered to Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/TRUNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/TRUNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>
User at Extension 11 answers call	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/TRUNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
DFT button bridges onto call. The call is active at Extension 11 and bridged at Extension 12.		***CSTANotReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12
	Call activity at Extension 11 will cause events to flow on this stream.	Call activity at Extension 11 will not cause any events to flow on this stream.

MERLIN MAGIX R2.1 and later

	Stream Monitoring Extension 11	Stream Monitoring Extension 12
Call delivered to Extension 11.	***CSTADeliveredEvent*** ***connection*** = D1C1 ***alertingDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>	***CSTADeliveredEvent*** ***connection*** = D2C1 ***alertingDevice*** = 12 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <UNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>
User at Extension 11 answers call	***CSTANotReadyEvent*** ***agentDevice*** = 11 ***agentID*** = 11	
.	***CSTAEstablishedEvent*** ***establishedConnection*** = D1C1 ***answeringDevice*** = 11 ***callingDevice*** = <ANI/ICLID/UNK> ***calledDevice*** = <DNIS/TRUNK> ***cause*** = EC_NEW_CALL ***Private Data*** ***trunkUsed*** = <trunk>	***CSTAConnectionClearedEvent*** ***droppedConnection*** = D2C1 ***releasingDevice*** = 12 ***cause*** = EC_NONE
DFT button bridges onto call. The call is active at Extension 11 and bridged at Extension 12.		***CSTANotReadyEvent*** ***agentDevice*** = 12 ***agentID*** = 12
	Call activity at Extension 11 will cause events to flow on this stream.	Call activity at Extension 11 will not cause any events to flow on this stream.

Appendix A

A

Contents

Appendix A

A

Supported MERLIN LEGEND Station Types

MLX Terminals

MLX-5	Supported
MLX-5 w/ Display	Supported
MLX-10	Supported
MLX-10 w/ Display	Supported
MLX-16 w/ Display	Supported
MLX-28 w/ Display	Supported
MLX-20L	Supported
Multi-Function Module	Not Supported

MLX Adjuncts

PassageWay Direct Connection	Not Supported
IROB (505A)	Not Supported
DSS	Not Supported
Headsets (Supra)	Not Supported

ATL Terminals

5 Button Membrane	Not Supported
10 Button Membrane	Not Supported
34 Button Membrane	Not Supported
34 Button Deluxe	Not Supported
34 Button SR w/ Display	Not Supported
BIS-10	Supported
BIS-22	Supported
BIS-22D	Supported
BIS-34	Supported
BIS-34 w/ Display	Supported
Attendant Console	Not Supported
MERLIN PFC Telephone	Not Supported

Tip/Ring Terminals

Door Phone Controller	Not Supported
Hands Free Unit (S203A)	Not Supported
Answering Machine	Not Supported
Transaction Phone	Not Supported
Facsimile	Not Supported
AA/VMS	Not Supported
External Alert	Not Supported
Announcement Unit	Not Supported
Dial Dictation	Not Supported
Modem	Not Supported
IROB	Not Supported
OPRE	Not Supported
CVIS INTRO	Not Supported

LS/GS CO Line

Magic On Hold w/ Music Coupler	Not Supported
Door Phone Controller	Not Supported
Loudspeaker Paging System	Not Supported
Music on Hold	Not Supported

ATL Adjuncts

Hands Free Unit (S102A & S202A)	Not Supported
Headset Adapter	Not Supported
General Purpose Adapter	Not Supported
Starset II & Supra Headsets	Not Supported
Starmate Headset	Not Supported
IROB (341)	Not Supported

Tip/Ring Sets

500 Sets	Not Supported
2500 Sets	Not Supported
2500 MMGL sets w/ Display	Not Supported
8110 & 8102 Sets	Not Supported
Videophone 2500	Not Supported
Picasso Still Image Phone	Not Supported

ETR Terminals

MLS-6	Supported
MLS-12	Supported
MLS-12D	Supported
MLS-18D	Supported
MLS-34D	Supported
ETR-6	Supported
ETR-18	Supported
ETR-18D	Supported
ETR-34D	Supported

STU Sets

ATL III	Not Supported
T/R III	Not Supported
MLX 5	Not Supported
MLX 10	Not Supported
MLX 20L	Not Supported
MLX 28	Not Supported

ATL Cordless Sets

Transtalk	Not Supported

BRI Sets

7500 Data	Not Supported

Telephone Set Model	TSAPI application can monitor set?	Generates TSAPI events when party to a call?	TSAPI application can control call?
MLX-5	yes	yes	yes
MLX-5D	yes	yes	yes
MLX-10	yes	yes	yes
MLX-10D	yes	yes	yes
MLX10-DP	yes	yes	yes
MLX-16DP	yes	yes	yes
MLX-20L	yes	yes	yes
MLX-20L as QCC	no	no	no
MLX-28D	yes	yes	yes
ATL 5-line membrane*	yes	yes	no
ATL 10-line membrane*	yes	yes	no
ATL 34-line membrane*	yes	yes	no
ATL 34-line Deluxe membrane*	yes	yes	no
ATL 10-line button HFAI*	yes	yes	no
ATL 34-line button BIS*	yes	yes	yes
ATL 34-line button BIS/DIS*	yes	yes	yes
ATL BIS-10	yes	yes	yes
ATL BIS-22	yes	yes	yes
ATL BIS-22D	yes	yes	yes
ATL BIS-34*	yes	yes	yes
ATL BIS-34D	yes	yes	yes
ATL MLC-5 Cordless	yes	yes	no
MDC 9000 Cordless	yes	yes	no
MDW 9000 Cordless/Wireless	yes	yes	no
MERLIN PFC Telephone	no	yes	no
Single line set - rotary	no	yes	no
Single line set - DTMF	no	yes	no
BRI 7500 data set	no	no	no

* These are vintage telephone models; no longer available for sale or lease.

Supported MERLIN MAGIX Station Types

MLX Terminals

MLX-5	Supported
MLX-5 w/ Display	Supported
MLX-10	Supported
MLX-10 w/ Display	Supported
MLX-16 w/ Display	Supported
MLX-28 w/ Display	Supported
MLX-20L	Supported
Multi-Function Module	Not Supported

MLX Adjuncts

PassageWay Direct Connection	Not Supported
IROB (505A)	Not Supported
DSS	Not Supported
Headsets (Supra)	Not Supported

Tip/Ring Terminals

Door Phone Controller	Supported
Hands Free Unit (S203A)	Supported
Answering Machine	Supported
Transaction Phone	Supported
Facsimile	Supported
AA/VMS	Supported
External Alert	Supported
Announcement Unit	Supported
Dial Dictation	Supported
Modem	Supported
IROB	Supported
OPRE	Not Supported
CVIS INTRO	Not Supported

LS/GS CO Line

Magic On Hold w/ Music Coupler	Not Supported
Door Phone Controller	Not Supported
Loudspeaker Paging System	Not Supported
Music on Hold	Not Supported

Tip/Ring Sets

500 Sets	Supported
2500 Sets	Supported
2500 MMGL sets w/ Display	Supported
8110 & 8102 Sets	Supported

STU Sets

T/R III	Not Supported
MLX 5	Not Supported
MLX 10	Not Supported
MLX 20L	Not Supported
MLX 28	Not Supported

BRI Sets

7500 Data	Not Supported

4400-series Terminals

4400	Supported
4400D	Supported
4406D+	Supported
4412D+	Supported
4424D+	Supported
4424LD+	Supported

4400-series Adjuncts

IROB (505A)	Not Supported
DSS	Not Supported
Headsets (Supra)	Not Supported

ETR Terminals

MLS-6	Supported
MLS-12	Supported
MLS-12D	Supported
MLS-18D	Supported
MLS-34D	Supported
ETR-6	Supported
ETR-18	Supported
ETR-18D	Supported
ETR-34D	Supported

Telephone Set Model	TSAPI application can monitor set?	Generates TSAPI events when party to a call?	TSAPI application can control call?
MLX-5	yes	yes	yes
MLX-5D	yes	yes	yes
MLX-10	yes	yes	yes
MLX-10D	yes	yes	yes
MLX10-DP	yes	yes	yes
MLX-16DP	yes	yes	yes
MLX-20L	yes	yes	yes
MLX-20L as QCC	no	no	no
MLX-28D	yes	yes	yes
4400	yes	yes	no
4400D	yes	yes	no
4406D+	yes	yes	no
4412D+	yes	yes	no
4424D+	yes	yes	no
4424LD*	yes	yes	yes
4424LD+ as QCC	no	no	no
Single line set - rotary	yes	yes	yes
Single line set - DTMF	yes	yes	yes
BRI 7500 data set	no	no	no

Abbreviations

A

ACD
Automatic Call Distribution

ALS
Automatic Line Selection

ANI
Automatic Number Identification

ARS
Automatic Route Selection

B

BRI
Basic Rate Interface

C

CBQ
Call Back Queueing

CSTA
Computer Supported Telephony Applications

CTI
Computer Telephony Integration

D

DFT
Direct Facility Termination

DGC
Directed Group Calling

DID
Direct Inward Dial

DLC
Direct Line Console

DLL
Dynamic Link Library

DND
Do Not Disturb

DNIS
Dialed Number Identification Service

DTAC
Direct Termination Attendant Console

E

ECMA
European Computer Manufacturers' Association

ESR
Event Service Routine

G

GPA
General Purpose Adapter

H

HFAI
Hands Free Answer on Intercom

HFU
Hands Free Unit

I

ICLID
Individual Call Line Identification

L

LND
Last Number Dialed – Renamed to Redial in MERLIN MAGIX 1.0

M

MLX
Multiline Telephone

N

NT2
Network Termination 2

O

OA&M
Operations, Administration and Maintenance

P

PRI
Primary Rate Interface

Q

QCC
Queued Call Console

R

RLP
Ringing Line Preference

S

SA
System Access

T

TCP/IP
Transmission Control Protocol/Internet Protocol

TSAPI
Telephony Services Application Programming Interface

U

UDP
Uniform Dial Plan

V

VMI
Voice Messaging Interface

VMS/AA
Voice Mail System/Auto-Attendant

Glossary

4424LD+ Telephone
A 24-line button digital telephone with a 7-line by 24-character display. *See also* Queued Call Console (QCC).

4400-series Telephone
A digital telephone that provides multiple line buttons for making or receiving calls or programming features.

A

API Control Services (ACS)
An application uses ACS (a subset of TSAPI) to open, close, and control a communication channel (known as a stream) to a Telephony Server. Once an application opens a stream, the application uses other TSAPI function calls on the stream to request CSTA services from the Telephony Server.

Associative Active
State of a MERLIN LEGEND SA button. An SA button is in an Associative Active state if a shared SA button for this SA is participating in a call.

Associative Hold
State of a MERLIN LEGEND SA button. An SA button is in an Associative Hold state if a shared SA button for this SA has a call on hold.

Automatic Line Selection
Programmed order in which the switch makes outside lines available to the user.

Automatic Number Identification (ANI)
Network service that automatically identifies a caller's billing number and transmits that number from the caller's local central office to another point on or off the public network.

Automatic Route Selection
Switch feature that automatically routes calls on outside trunks according to the number dialed and trunk availability.

B

Basic Rate Interface (BRI)
Standard digital frame format that specifies the protocol between the communications system and a terminal. BRI runs at 192 kbps and provides two 64-kbps voice (or B-channels) and one 16 kbps signaling (or D-channel) per port. The remaining 48 kbps are used for framing and D-channel contention.

"Behind the switch mode"
A MERLIN LEGEND switch-administered mode where trunk lines from the MERLIN LEGEND switch connect to station ports on another switch. The MERLIN LEGEND switch user accesses features on the other switch (switchhook flashes, for example, pass through the MERLIN LEGEND switch to the other switch).

C

Callback Screening
Feature that allows a user to listen in while a caller is leaving a Voice Mail message. If desired, the user may interrupt Voice Mail treatment and join the call as a regular call participant.

Callback Queuing (CBQ)
Feature that completes calls to busy extensions or pools.

Collected Digits
User-entered digits that have been collected by a voice prompter unit. *See also* Prompted Digits.

Computer-Supported Telephony Applications (CSTA)
CSTA is a European Computer Manufacturers' Association (ECMA) standard that defines a standard set of Telephony Services, responses, and events. The CSTA definitions form the foundation for PassageWay Telephony Services. Although CSTA provides standard service and event definitions, it does not provide an Application Programming Interface (API) definition. TSAPI provides the API for PassageWay Telephony Services.

Cold Start
Type of MERLIN LEGEND switch restart. A cold start tears down all calls but retains administered translations.

Computer Telephony Integration (CTI)
The integration of services provided by a telephone and a computer.

CSTA Standard 217
CSTA Services standard of December 1994 (as opposed to the earlier June 1992 Services standard)

CTI link
A link between a Telephony Server and a switch. In this product, this is an NT2 connection between the Telephony Server and MERLIN LEGEND switch. *See also* NT2.

D

Destination Digits
String of digits that is used to dial a call. These digits may include routing or facilities access digits.

Dialed Number Identification Service (DNIS)
Service provided by the network. Identifies which group was called based on the 800 or 900 service number dialed and makes the number dialed available to the switch.

Direct Facility Termination (DFT)
A programmable button on a MERLIN LEGEND phone that is programmed to be a central office line. Also known as a "Personal Line" button. The button connects to a Central Office trunk. The user may use the button to place outgoing calls or answer incoming calls on that trunk.

Direct Inward Dial (DID)
Service that transmits from the telephone company central office and routes incoming calls directly to the called extension, calling group, or outgoing trunk pool, bypassing the system operator.

Direct Line Console (DLC)
Telephone used by a system operator to answer outside calls (not directed to an individual or a group) and inside calls, transfer calls, make outside calls for users with outward calling restrictions, set up conference calls, and monitor system operation.

Direct Termination Attendant Console (DTAC)
See Direct Line Console (DLC).

Directed Group Calling (DGC)
See Group Calling.

Dynamic Link Library
A library of compiled subroutines that are linked dynamically to a Windows executable program at the time it is run.

Do Not Disturb (DND)
A MERLIN LEGEND OR MERLIN MAGIX switch feature that prevents calls arriving at a user's phone from ringing at that phone.

E

End of Dialing
This term is used primarily with external trunk calls to indicate that the user has completed dialing a call. For analog trunks (non-PRI trunks), the switch determines "end-of-dialing" when a timer expires after the last digit has been dialed. For PRI trunks, "end-of-dialing" is signaled on the CO trunk by the central office. When this occurs, it indicates that the user has dialed a valid telephone address.

Enhanced Tip/Ring (MLX)
An analog or digital telephone that provides multiple line buttons for making or receiving calls or programming features.

Event Service Routine (ESR)
In some operating system environments (Windows and Windows NT), an application can use an ESR to receive asynchronous notification of arriving events. The ESR mechanism notifies the application of arriving events, but does not remove the events from the event queue. The application must use ***acsGetEventBlock()***, ***acsGetEventPoll()***, or ***eventNotify()*** to receive the message. The application can use an ESR to trigger a specific action when an event arrives in the event queue. For more information, see *Telephony Services Application Programming Interface (TSAPI), Version 2*.

G

General Purpose Adapter (GPA)
An adjunct used with ATL telephones to add a tip/ring device such as a fax or modem or answering machine. ATL sets are not supported beginning in MERLIN MAGIX 1.5.

Group Calling
A MERLIN LEGEND OR MERLIN MAGIX switch feature that directs incoming calls to a specific group of telephones (a *calling group*). A calling group is a team of individuals who answer and handle the same type of calls, for example, high-volume work groups such as sales, service, marketing, repair, and technical support. Also fax machines that receive a large number of fax messages can be placed in a calling group to allow multiple calls to be sent. (Up to thirty-two calling groups, with up to twenty members in each group, are supported.)

Through Group Calling, all members in the calling group are assigned to a single extension number. Specific trunks can be assigned to ring directly into the calling group so that outside callers can dial a published telephone number to reach the group, bypassing the operator.

H

Hands Free Answer on Intercom (HFAI)
A feature that allows a user to answer a voice-announced call.

Hands Free Unit
A speakerphone used with ATL telephones. See also “Headset Adapter 502C.” ATL sets are not supported beginning in MERLIN MAGIX 1.5.

Headset Adapter 502C
An adjunct used with ATL telephones to add a headset. ATL sets are not supported beginning in MERLIN MAGIX 1.5.

I

Individual Call Line Identification (ICLID)
Commonly known as Caller ID. A service provided by some local telephone companies (if local regulations allow) that supplies the calling party telephone number. In Release 3.0 and later, an 800 GS/LS-ID module on the system can capture this information and display it on the screens of MLX telephones. Beginning in MAGIX Release 1.0, the 800 GS/LS-ID module, the 412 LS-ETR and 412 LS-TDL module can can capture this information and display it on the screens of MLX, ETR and 4400-series telephones.

Invoke ID
An identifier within TSAPI (and CSTA) that allows an application to correlate the service confirmation events with requests in the context of a TSAPI stream.

IPX/SPX
LAN communication protocol used between a client PC and a NetWare server.

I-use call
The current active call at a telephone. The red LED is lit at the button for this call. The user is off-hook on this call.

L

Last Number Dialed (LND)
A feature that re-dials the last number a user has called without the need for the user to re-enter the dialed digits. See also “Redial”.

M

MERLIN LEGEND PBX Driver
The MERLIN LEGEND PBX Driver is a software module on a Telephony Server that interfaces switch-independent Telephony Server software to the MERLIN LEGEND Communications System. This software terminates and manages the MERLIN LEGEND CTI link.

MLX-20L Telephone
A 20-line button digital telephone with a 7-line by 24-character display. *See also* Queued Call Console (QCC).

Multiline Telephone (MLX)
An analog or digital telephone that provides multiple line buttons for making or receiving calls or programming features.

Multiline Telephone (MLX) Adjunct
An MLX or 4400-series telephone adjunct used to add a tip/ring device (such as a fax, modem, or answering machine) or an additional ringer.

Multi-function module
An MLX or 4400-series telephone adjunct used to add a tip/ring device (such as a fax, modem, or answering machine) or an additional ringer.

N

Network Termination 2 (NT2)
ISDN protocol designed to support MERLIN LEGEND terminal endpoints at the NT2 reference point defined by CCITT I.411. NT2 is a line protocol for the MLX terminal family that provides standards-compliant channel access plus advanced local features.

Normal Mode
Condition of a MERLIN LEGEND telephone. The telephone is in one of the following states in Normal Mode: forced idle; program mode; maintenance mode; administration mode; test mode; private directory program; maintenance busy; inspect mode; entering pound code, star code, or feature code; turning on/off night service with a password; entering an account code; entering an authorization code; activating Direct Voice Mail; activating Drop; changing Extension Status when in calling group or hotel/motel mode; activating/deactivating Forward; activating/deactivating Follow Me; activating Send/Remove Message; activating Leave Message without Calling; activating Cancel Message Sent; activating Posted Message; entering Night Service password; or activating/deactivating Reminder Service.

P

Primary Rate Interface (PRI)
A standard Integrated Services Digital Network (ISDN) frame format that specifies the protocol used between two or more communications systems. North American PRI runs at 1.544 Mbps and provides 23 64-Kbps B-channels (voice or data) and one 64-Kbps D-channel (signaling). The D-channel is the 24th channel of the interface and contains multiplexed signaling information for the other 23 channels**.**

Private Data
Private Data is a TSAPI mechanism that allows a vendor to enhance TSAPI services and events and even provide new services within the TSAPI framework. The MERLIN LEGEND PBX Driver uses private data to provide any call prompting digits that have been collected for a call. The programming interface to MERLIN LEGEND private data features may only be used with a MERLIN LEGEND switch.

More specifically, the privateData parameter carries MERLIN LEGEND private data in those events where MERLIN LEGEND CTI supplies Private Data. This document defines a C structure that overlays the ***privateData*** parameter and gives programmers access to MERLIN LEGEND Private Data.

Prompted
During a Make Call request, the originator's telephone has been "prompted" when the MERLIN LEGEND OR MERLIN MAGIX switch has cued the user to go off-hook on the speakerphone.

Prompted Digits
An industry term having the same meaning as "collected digits." *See also* Collected Digits.

Q

Queued Call Console (QCC)
An MLX-20Lor 4424LD+ telephone used by a system operator in Hybrid/PBX mode only. Used to answer outside calls (directed to a system operator position) and inside calls, direct inside and outside calls to an extension or an outside telephone number, serve as a message center, make outside calls for users with outward calling restrictions, set up conference calls, and monitor system operation. *See also* MLX-20L or 4424LD+ Telephone.

R

Redial
Beginning in MERLIN MAGIX Release 1.0, the "Last Number Dialed" feature was renamed to "Redial. A feature that re-dials the last number a user has called without the need for the user to re-enter the dialed digits. See also "Last Number Dialed".

Responding Mode
Describes the condition of a telephone. The telephone is in Responding Mode when it is plugged in and has a MERLIN LEGEND OR MERLIN MAGIX switch recognized class mark.

Ringing Line Preference (RLP)
Feature which selects a preferred line appearance when a call arrives.

Routing Digits
Digits in the Destination Digits that either select an outgoing trunk facility or direct the Automatic Route Selection feature to choose the trunk route for an outgoing call.

S

SA Button
See System Access Button.

Senderization
Point in the placement of an outgoing call where the originating extension is dialing and the MERLIN LEGEND switch has selected the outgoing trunk but the MERLIN LEGEND OR MERLIN MAGIX switch has not connected the originating extension to the trunk. The MERLIN LEGEND OR MERLIN MAGIX switch is providing dialing feedback to the originating station and passing the extension's dialed digits out over the trunk.

Service Observing
Feature that adds an extension (the Service Observer) to a call with a listen-only connection whenever the observed extension is active on a call.

Supervised Transfer
A transfer where the consulting party waits for the consultated party to answer before completing the transfer.

System Access (SA) Button
A type of line button on a MERLIN LEGEND switch station set (used in Hybrid/PBX mode) to make or receive inside or outside calls. A user typically has several of these buttons on a telephone set. Calls appear on SA buttons (as well as other types of buttons.) There are various types of SA buttons: SA-Ring, SA-Voice, SA-Originate-Only-Ring, SA-Originate-Only-Voice, SSA-Shared SA.

T

TCP/IP
Communications protocol used between a client PC and a server.

Telephony Server
A Telephony Server is a server on a local area network that provides Telephony Services to client applications. The Telephony Server has a Computer Telephony Integration (CTI) link to a MERLIN LEGEND Communications System. A client application makes TSAPI requests of the Telephony Server. The Telephony Server passes these requests to the MERLIN LEGEND PBX Driver, which, in turn, passes them over the CTI link to the MERLIN LEGEND switch. The MERLIN LEGEND switch processes the request and returns responses and call events through the Telephony Server to the requesting application.

Telephony Services Application Programming Interface (TSAPI)
TSAPI is the C programming language interface to CentreVu Telephony Services. Application programmers use TSAPI to access CSTA services, responses, and events. TSAPI is switch-independent and supports many Telephony Services-compliant drivers, including the MERLIN LEGEND PBX Driver.

TSAPI Cross Reference ID
An identifier within TSAPI (and CSTA) that allows an application to correlate events with the monitor request in the context of a TSAPI stream.

TSAPI stream
A connection between a PassageWay Telephony Services application and a MERLIN LEGEND PBX Driver over which TSAPI requests, acknowledgments, events, etc. flow.

U

Uniform Dial Plan (UDP)
A MERLIN LEGEND OR MERLIN MAGIX switch feature (Release 6.0 and later) that allows a caller at any station in a private network to dial the same number of digits (i.e., without the need to dial an access code) to reach any other station in the same private network, even if the originating station set is physically connected to one MERLIN LEGEND OR MERLIN MAGIX switch and the terminating station set is physically connected to a different MERLIN LEGEND OR MERLIN MAGIX switch (e.g., one MERLIN LEGEND OR MERLIN MAGIX switch of the private network is in California and the other is in New Jersey). If the originating station set and the terminating station set are connected to the same MERLIN LEGEND OR MERLIN MAGIX switch, the UDP is called a *local* UDP. If the originating station set and the terminating station set are connected to different MERLIN LEGEND OR MERLIN MAGIX switches, the UDP is called a *non-local* UDP.

Unsupervised Transfer
A transfer where the consulting party completes the transfer without waiting for the consulting party to answer.

V

Voice Mail System/Auto-Attendant (VMS/AA)
Application that allows users to send messages, forward messages with comments, and reply to messages to other extensions in the VMS.

Bibliography

Telephony Services Application Programming Interface (TSAPI), Version 2

Standard ECMA-217 Services for Computer-Supported Telecommunications Applications (CSTA), European Computer Manufacturers' Association, December 1994

MERLIN LEGEND Communications System Release 7.0 Feature Reference, 555-670-110

MERLIN MAGIX Integrated System Feature Reference – Release 2.2 and Earlier, 555-722-110

Network Manager's Guide for MERLIN LEGEND® Advanced Communications System Windows NT® Driver

Network Manager's Guide for MERLIN MAGIX® Integrated System PBX Driver

CentreVu® Computer-Telephony - Telephony Services and CallVisor® PC Installation

CentreVu® Computer-Telephony - Telephony Services Administration and Maintenance