CSTA Services Interface

Programming Guide

Order Number: 05-0992-001

Revision/Update Information: This is a new manual

Software/Version: CSTA Services Interface
Version 1.0

Publication date: December 3, 1997

For **Sales Offices** and other contact information, visit our website at **http://www.dialogic.com.**

Contents

About This Manual vii

1 Introduction

1.1 Purpose of the CSTA Services Interface 1-1
1.2 How You Use the CSTA Services Interface 1-1
1.3 The CSTA Services Interface Environment 1-3

2 Programming Guidelines

2.1 Overview 2-1
2.2 Introduction to Functions 2-1
2.3 The CSTA Services Interface Environment 2-2
2.3.1 Structure and Content of the Private Data Array 2-2
2.3.2 Connection ID 2-3
2.3.3 Device ID 2-4
2.3.4 Invoke ID 2-4
2.3.5 Link ID 2-4
2.3.6 Monitor ID 2-4
2.3.7 Event Generation 2-4
2.4 Creating Your Application 2-5
2.4.1 Function Calling Conventions 2-5
2.4.2 Initialisation and Management 2-5
2.4.3 Synchronous and Asynchronous Operation Using Threads 2-6
2.4.4 Synchronous and Asynchronous Processing Recommendations 2-8
2.5 Example Stub Function 2-8
2.6 Example Files 2-10
2.7 Compiling Your Code 2-11
2.8 Operational Verification 2-11

3 Management Operation Functions

csa_fCloseLink 3-2
csa_fInitialise 3-3
csa_fShutdown 3-5

4 Management Status Report Functions

csa_fErrorReport 4-2
csa_fLinkDown 4-3
csa_fLinkUp 4-4

5 Service Request Functions

csa_fAlternateCall 5-2
csa_fAnswerCall 5-4
csa_fAssociateData 5-6
csa_fCallCompletion 5-8
csa_fClearCall 5-10
csa_fClearConnection 5-12
csa_fConferenceCall 5-14
csa_fConsultationCall 5-16
csa_fDeflectCall 5-18
csa_fEscape 5-20
csa_fHoldCall 5-21
csa_fMakeCall 5-23
csa_fMakePredictiveCall 5-25
csa_fMonitorStart 5-27
csa_fMonitorStop 5-29
csa_fPickupCall 5-30
csa_fPickupGroupCall 5-32
csa_fQueryDevAgentState 5-34
csa_fQueryDevDND 5-36
csa_fQueryDevForwarding 5-38
csa_fQueryDevInfo 5-40
csa_fQueryDevMsgWaiting 5-42
csa_fReconnectCall 5-44
csa_fRetrieveCall 5-46
csa_fRouteEnd 5-48
csa_fRouteSelect 5-49
csa_fSendDTMF 5-51
csa_fSetAgentState 5-53
csa_fSetDoNotDisturb 5-55
csa_fSetForwarding 5-57
csa_fSetMsgWaiting 5-59
csa_fSingleStepTransfer 5-61
csa_fSnapshotDevice 5-63
csa_fTransferCall 5-65

6 Service Response Functions

csa_cbfAlternateCallResult 6-2
csa_cbfAnswerCallResult 6-3
csa_cbfAssociateDataResult 6-4
csa_cbfCallCompletionResult 6-5
csa_cbfClearCallResult 6-6
csa_cbfClearConnectionResult 6-7
csa_cbfConferenceCallResult 6-8

csa_cbfConsultationCallResult 6-10
csa_cbfDeflectCallResult 6-12
csa_cbfErrorResult 6-13
csa_cbfEscapeResult 6-14
csa_cbfHoldCallResult 6-15
csa_cbfMakeCallResult 6-16
csa_cbfMakePredictiveCallResult 6-18
csa_cbfMonitorStartResult 6-20
csa_cbfMonitorStopResult 6-21
csa_cbfPickupCallResult 6-22
csa_cbfPickupGroupCallResult 6-23
csa_cbfQueryDevAgentStateResult 6-24
csa_cbfQueryDevDNDResult 6-25
csa_cbfQueryDevForwardingResult 6-26
csa_cbfQueryDevInfoResult 6-29
csa_cbfQueryDevMsgWaitingResult 6-31
csa_cbfReconnectCallResult 6-32
csa_cbfRetrieveCallResult 6-33
csa_cbfSendDTMFResult 6-34
csa_cbfSetAgentStateResult 6-35
csa_cbfSetDoNotDisturbResult 6-36
csa_cbfSetForwardingResult 6-37
csa_cbfSetMsgWaitingResult 6-38
csa_cbfSingleStepTransferResult 6-39
csa_cbfSnapshotDeviceResult 6-41
csa_cbfTransferCallResult 6-44

7 Service Event Functions

csa_cbfAgentStateEvent 7-2
csa_cbfCallClearedEvent 7-4
csa_cbfCallInformationEvent 7-6
csa_cbfConferencedEvent 7-9
csa_cbfConnectionClearedEvent 7-13
csa_cbfDeliveredEvent 7-16
csa_cbfDivertedEvent 7-19
csa_cbfEstablishedEvent 7-22
csa_cbfFailedEvent 7-25
csa_cbfHeldEvent 7-28
csa_cbfNetworkReachedEvent 7-31
csa_cbfOriginatedEvent 7-34
csa_cbfPrivateEvent 7-37
csa_cbfQueuedEvent 7-38
csa_cbfRetrievedEvent 7-41
csa_cbfRouteEnd 7-44

csa_cbfRouteRequest 7-45
csa_cbfServiceInitiatedEvent 7-48
csa_cbfTransferredEvent 7-50

A Error and Event Codes

A.1 Introduction A-1
A.2 Management Functions Return Codes A-1
A.3 Service Request Return Codes A-2
A.4 Service Response Return Codes A-5
A.5 Service Event Return Codes A-5
A.6 CSTA Event Codes A-6

B CSTA Services and CT-Connect Functions

B.1 CSTA Services Mappings B-1

Index

Figures

1.1 CSTA Services Interface Environment 1-4
2.1 Synchronous Versus Asynchronous Operation 2-7
2.2 Example Stub Function 2-8

Tables

A.1 csa_tMgmtStatus Return Codes A-1
A.2 csa_tStatus Return Codes A-2
A.3 csa_tResponseStatus Return Codes A-5
A.4 csa_tEventStatus Return Codes A-5
A.5 csa_tEventCause Return Codes A-6
B.1 CSTA Services and CT-Connect Functions B-1

About This Manual

This manual describes how you use Dialogic's Computer Supported Telecommunications Applications (CSTA) Services Interface to develop a protocol converter between your switching system and CSTA.

Audience

This manual is for anyone using the CSTA Services Interface to develop protocol conversion software. Conversion software developers are assumed to have knowledge of the following:

- Writing programs in C or C++.
- CSTA Phase II services, parameters, and event messages.
- Protocol, service, and event messages generated by the switching system.

Associated Documentation

CSTA Services Interface

The CSTA Services Interface documentation set also includes the following:

- *CSTA Services Interface Installation Letter*

  This letter describes how you install the CSTA Services Interface software and example files on your system.

This manual and the installation letter are available on the CSTA Services Interface CD-ROM in PDF format for reading with Adobe® Acrobat®.

ECMA CSTA Standards

The CSTA protocol is defined by a standards body, the European Computer Manufacturers Association (ECMA). For details of CSTA services, parameters, and event messages, refer to the following ECMA standards:

- *Scenarios for Computer Supported Telecommunications Applications (CSTA) Phase II* - Technical Report ECMA-TR/68.
- *Services for Computer Supported Telecommunications Applications (CSTA) Phase*

II - Standard ECMA-217.

- *Protocol for Computer Supported Telecommunications Applications (CSTA) Phase II* - Standard ECMA-218.

CTI Server Documentation

Dialogic's Computer Telephony Integration (CTI) software, CT-Connect™ (CTC), is documented in the following:

- *CT-Connect Introduction*

  This manual provides an overview of CTC, computer telephony concepts, and telephony application functions.

- *CT-Connect Installation and Administration Guide*

  This manual describes how to install, configure, and monitor CTC.

- *CT-Connect Programming Guide*

  This manual provides detailed descriptions of the CTC Application Programming Interface (API) routines and guidelines for using these routines.

Switching System Documentation

Refer to the documentation supplied with your switching system for details of its features and characteristics.

Terms and Definitions

The following terms are used throughout this manual:

Term	Definition
CSTA Services Interface	The CSTA Services Interface software.
CSTA environment	A CSTA-compliant computer-based telephony environment.
Stub function	A brief version of a function to outline its structure and parameters. A stub function acts as a place-holder for where code will be implemented.
Switching system	Any system that uses the CSTA Services Interface to interact with a CSTA environment. For example, a traditional telephone system or an open switching system.

Conventions

The following conventions are used throughout this manual:

Convention	Meaning
`courier`	Monospaced typeface indicates code examples, example screen outputs, and commands you enter.
drive:	Italic (slanted) typeface indicates variable values, placeholders, and function arguments.

Chapter 1: Introduction

1.1: Purpose of the CSTA Services Interface

The CSTA protocol defines a non-proprietary standard for computer telephony integration between a switching system and a computing environment. By implementing the CSTA protocol, a switching system makes its services available to applications in an external CSTA-compliant computer-based environment (referred to as a **CSTA environment**).

The CSTA services and interface protocol are applicable to a broad range of switching systems, for example:

- Traditional telephone systems
- Open switching systems
- Public network switching services
- H.323 packet voice services

For ease of reference, this manual uses the term **switching system** to refer to any system that uses the CSTA Services Interface to interact with a CSTA environment.

The purpose of the CSTA Services Interface is to generate and receive the switching system side of the CSTA message stream, which is encoded in Abstract Syntax Notation 1 (ASN.1). ASN.1 is platform-independant and extensible to enable development of open communication applications. However, ASN.1 has a complex structure, which requires considerable development effort. The CSTA Services Interface provides you with ASN.1 translation so that you do not need to work directly with the ASN.1 encoding rules. You can use the CSTA Services Interface to implement a CSTA-compliant interface for your existing switching systems and for new switching systems you might be developing.

1.2: How You Use the CSTA Services Interface

The CSTA Services Interface provides you with a toolkit of functional software and example source code. These features enable you to integrate your switching system into a CSTA environment.

The CSTA Services Interface automatically performs three main tasks on your behalf:

- Initialises and manages links with the CSTA environment.
- Encodes and decodes the CSTA-compliant ASN.1 message flow to and from the CSTA environment.
- Processes and simplifies the CSTA messages to allow ease of handling.

To integrate the CSTA message flow with your switching system, you need to write code that performs these tasks:

- Processes CSTA Services Interface telephony function calls that represent requests arriving from the CSTA environment.
- Passes instructions to your switching system to perform the telephony actions requested.
- Receives request responses and event notifications from your switching system.
- Generates the request reponses and event notifications required by the CSTA protocol and passes them to the CSTA Services Interface for ASN.1 encoding and transmission.

With the addition of your software, the CSTA Services Interface acts as a protocol converter that receives CSTA messages from the CSTA environment and converts them into commands for execution by your switching system. Similarly, messages from your switching system are converted into CSTA-compliant messages for return to the CSTA environment.

The CSTA Services Interface generates messages complying with the ECMA CSTA standards. As such, the CSTA Services Interface can be used in CSTA environments with any CSTA-compliant CTI server. Dialogic's CT-Connect is a CSTA-compliant CTI server, but the CSTA Services Interface is not limited to use just with CT-Connect. If you intend to use other vendor's CTI servers, you will need to check for any specific requirements they have for the sequencing of CSTA messages.

To help you write your code, the CSTA Services Interface toolkit includes a set of example C files that contain:

- A sample main program, showing a typical application initialisation sequence.
- Outline functions (referred to as **stub functions)**, one for each supported CSTA request.
- Function prototypes, one for each supported CSTA response and event.

In addition, the CSTA Services Interface kit contains:

- Header and type definition files.
- Microsoft® Visual C++® project and workspace definitions.
- A run-time library.

This manual provides guidance on the syntax and use of the example functions and on how these are integrated with the CSTA Services Interface. Note that you are free to create all of the files required from scratch, but the descriptions in this manual assume that you are using the example files supplied.

This manual also provides some guidance on the requirements of the CSTA protocol, but does not do this in detail. The CSTA Services Interface helps you build, transmit, receive, and decode CSTA messages, but it does not ensure that you send these messages in a sequence that conforms to the CSTA standards. You must be familiar with, and refer to, the ECMA standards listed in the Preface to ensure that the message flow you generate using the CSTA Services Interface conforms to the CSTA protocol requirements.

Section 1.3 describes the CSTA Services Interface environment and outlines the major features and requirements that you need to understand before writing your code.

1.3: The CSTA Services Interface Environment

Figure 1.1 illustrates the major components of the CSTA Services Interface environment and how they interact. You are responsible for providing the application (called CSTA_SERVICES_MAIN.EXE in the figure) based upon the sample files supplied in the CSTA Services Interface kit:

- CSTA_SERVICES_SAMPLE_MAIN.C

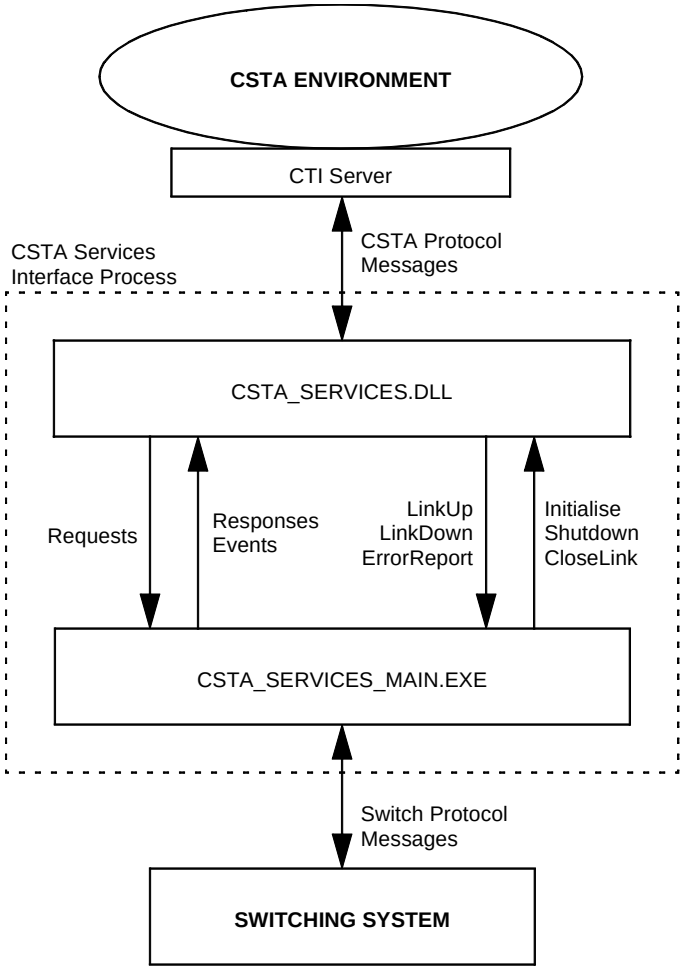
  Provides a sample application showing how the CSTA_SERVICES.DLL is initialised with pointers to the functions you have coded in the CSTA_SERVICES_STUBS.C file. Complete the application so that it behaves in the way you require.

- CSTA_SERVICES_STUBS.C

  Provides a set of stub functions to be called by the CSTA_SERVICES.DLL. The CSTA Services Interface defines a stub function for each supported CSTA request and management status operation. You have to complete the functions required to implement your application.

In addition, function prototypes for routines implemented in the CSTA_SERVICES.DLL are included in the CSTA_SERVICES_API.H file supplied. There is a function prototype for each CSTA response, CSTA event, and management control operation supported by the CSTA Services Interface. Your application has to call these functions to return the correct sequence of responses and events required by the CSTA environment and to control the CSTA Services Interface.

Figure 1.1: CSTA Services Interface Environment



Follow Figure 1.1 to understand how a typical flow of messages passes through the CSTA Services Interface environment after CSTA_SERVICES_MAIN.EXE has initialised CSTA_SERVICES.DLL:

1. The CSTA environment CTI server sends CSTA protocol messages to the system running the CSTA Services Interface process.
2. When a link is established, CSTA_SERVICES.DLL calls the Link Up function in CSTA_SERVICES_MAIN.EXE.
3. The CTI server sends CSTA requests to the CSTA Services Interface process.

4. CSTA_SERVICES.DLL translates the ASN.1 encoded stream of CSTA messages and calls the corresponding service request functions you have coded in CSTA_SERVICES_MAIN.EXE.
5. Your functions process the requested telephony actions and send the appropriate switch protocol messages to the switching system.
6. Upon receiving progress messages from the switching system, your CSTA_SERVICES_MAIN.EXE calls the corresponding functions in CSTA_SERVICES.DLL to generate the required CSTA responses and events for return to the CTI server.

To provide the CSTA_SERVICES_MAIN.EXE application, you complete the sample project provided in the CSTA Services Interface kit. The remainder of this manual contains the information you need to write your application:

- Chapter 2 provides programming guidelines.
- Chapters 3 to 7 define all of the functions and their parameters.
- Appendix A lists and describes all error messages and status return codes.
- Appendix B maps the CSTA Services Interface Service Requests to their CT-Connect API equivalents.

Chapter 2: Programming Guidelines

2.1: Overview

This chapter provides you with background information to help develop your application. This includes the requirements you must fulfill to work with the CSTA Services Interface as well as more general hints and tips.

In this chapter, it is assumed that you are developing your application using the project files provided with the CSTA Services Interface kit. If you wish to develop original code, you must create an application that contains functions to handle all exchanges between your switching system and the CSTA Services Interface.

2.2: Introduction to Functions

To handle all aspects of communications between the CSTA environment and your switching system, the CSTA Services Interface provides example files containing stub or prototype functions in the following categories:

- Management Operations

  These functions, provided by the CSTA_SERVICES.DLL, are called by your application to perform management actions, such as initialising and shutting down the CSTA Services Interface.

- Management Status Reports

  These functions, provided by your application, are called by CSTA_SERVICES.DLL whenever a significant management event occurs, such as a CTI server link up or down or upon error reports.

- Service Requests

  These functions, provided by your application, are called by CSTA_SERVICES.DLL whenever a CSTA service has been requested by the CTI server. Your functions must handle the information passed by CSTA_SERVICES.DLL and translate it to the format required by your switching system.

- Service Responses

  These functions, provided by CSTA_SERVICES.DLL, are called by your application

to generate the required CSTA responses for return to the CTI server.

- Service Events

  These functions, provided by CSTA_SERVICES.DLL, are called by your application to generate CSTA events for return to the CTI server.

The files contain stub functions for the Management Status Reports and Service Requests. The files contain function prototypes for Management Operations, Service Responses, and Service Events.

Chapters 3 to 7 describe the functions in detail. Each description includes C code for the function and defines its parameter values. Refer to these descriptions when writing your code. Once fully coded, you compile the functions to generate your application.

Section 2.3 describes characteristics of the CSTA Services Interface environment that you need to be aware of before writing your code.

2.3: The CSTA Services Interface Environment

You should refer to the ECMA CSTA standards listed in the Preface to understand the characteristics of the CSTA protocol and the environment it defines.

This section describes key characteristics of the CSTA Services Interface environment.

You must ensure that your use of the private data array takes into account the structure and content requirements of the CSTA environment (see Section 2.3.1).

You need to understand the meaning and use of the following identifier parameters (IDs):

- Connection ID (see Section 2.3.2)
- Device ID (see Section 2.3.3)
- Invoke ID (see Section 2.3.4)
- Link ID (see Section 2.3.5)
- Monitor ID (see Section 2.3.6)

You should take note also of CSTA Services Interface event generation (see Section 2.3.7).

2.3.1: Structure and Content of the Private Data Array

As described in the functional descriptions, the private data array has three identical data structures containing private data passed to, or returned from, your switching system.

Each data structure has three fields:

Manufacturer ID	Data Length	Data
(24 characters)	(Unsigned Integer)	(Up to 128 Unsigned Characters)

It is a requirement of the CSTA environment that the data in these fields conforms to a specific structure. When composing private data to pass to the CSTA_SERVICES.DLL in response or event function calls, your application must observe this structure. Similarly, your application should handle the same structure when private data is passed in from the CSTA_SERVICES.DLL in request function calls.

The required structure for each of the fields is as follows:

- **Manufacturer ID**

  Manufacturer IDs are allocated by the International Standards Organisation (ISO) and uniquely identify the manufacturer of a switching system (for example, 1.1). If you do not have a Manufacturer ID, contact ISO to have your own Manufacturer ID allocated. ISO can also provide a list of other Manufacturer IDs.

- **Data Length**

  Indicates the length of data written in the **Data** field (up to 128 bytes).

- **Data**

  The first two bytes of the Data field must contain the following values:

4	Length of remaining data (Up to 126 bytes)

  Note that the value in the second byte is the same as the value in the **Data Length** field minus these first two bytes (that is, up to 128 minus 2).

2.3.2: Connection ID

The Connection ID is a combination of the Device ID and the call reference identifier. The call reference identifier is allocated by your switching system, if supported, or by your application. Each call must have a unique call reference identifier. You should avoid reallocating a call reference identifier immediately after the end of a call to reduce the potential for confusion within the application in the CSTA environment.

The Connection ID represents the involvement of a particular device in a particular call and you use it extensively in the service response and service event functions.

2.3.3: Device ID

The Device ID uniquely identifies a device within a switching domain. The switching system controlling the switching domain allocates the Device ID.

The CSTA Services Interface supports both dialable numbers (DNs) and non-dialable device numbers as Device IDs. Non-dialable device numbers identify devices such as trunks. For example, a DN of 98005000 is a number you could physically dial to make a call whereas a device number of 253789 is simply an ASCII decimal representation of an integer value.

2.3.4: Invoke ID

The Invoke ID is used to match a request and a response. The CSTA Services Interface specifies the Invoke ID when calling the service request function.

You use the Invoke ID when calling the service response function corresponding to the service request. Each Invoke ID used on a particular link is unique for the duration of the request/response cycle.

2.3.5: Link ID

The Link ID is used to identify a link from a CTI server to the CSTA Services Interface. The CSTA Services Interface assigns the Link ID. There can be up to a maximum of 10 concurrent links from CTI servers to the CSTA Services Interface.

Link state changes are reported by the CSTA_SERVICES.DLL calling the csa_fLinkUp and csa_fLinkDown management functions in your application.

2.3.6: Monitor ID

The Monitor ID is used to identify each monitoring process on a device. Your application allocates the Monitor ID in response to a Monitor Start request.

Your application should use the Monitor ID when generating all events on the monitored device. If you perform more than one Monitor Start on a device, you need to call event generation functions for each Monitor ID.

2.3.7: Event Generation

The events that your code should generate as a result of switching operations are defined in the ECMA standards listed in the Preface. You need only generate events for a device which is being monitored.

2.4: Creating Your Application

This section describes points to note when you are creating your application, covering the following topics:

- Function calling conventions (see Section 2.4.1)
- Initialisation and management (see Section 2.4.2)
- Synchronous and asynchronous operation with threads (see Section 2.4.3)
- Synchronous and asynchronous processing recommendations (see Section 2.4.4)

2.4.1: Function Calling Conventions

The CSTA Services Interface CSTA_SERVICES.DLL calls functions in your application. To ensure compatibility with CSTA_SERVICES.DLL, all of your functions must be exported with undecorated names using the **stdcall** calling convention.

Using **stdcall** ensures that your functions conform to the standard WIN32 argument-passing convention. This convention defines the following:

- The right-hand parameter is pushed onto the stack first.
- Parameters are popped from the stack by the called function.
- Names are not decorated or otherwise amended.

2.4.2: Initialisation and Management

Most initialisation and management actions are handled automatically by the CSTA Services Interface. Your application invokes management operations and accepts management status reports using the CSTA Services Interface management functions.

At startup, the CSTA Services Interface initialises in these stages:

1. Your CSTA_SERVICES_MAIN.EXE performs any customized initialisation tasks you have coded.
2. Your CSTA_SERVICES_MAIN.EXE has to pass CSTA CTI link communications parameters and the function call back locations to CSTA_SERVICES.DLL using the csa_fInitialise function.
3. CSTA_SERVICES.DLL records the call back locations so that it can invoke the relevant function when a CSTA request arrives.

In your application, you call the management operation functions as follows:

- csa_fInitialise

  To initialise the CSTA Services Interface, pass communications parameters, and

pass function call back locations.

- csa_fCloseLink

  To request that the CSTA Services Interface closes down the link to a CTI server if the link from the switching system has failed.

- csa_fShutdown

  To shutdown the CSTA Services Interface (disconnecting any links to CTI servers).

Chapters 3 describes the management operation functions.

Following initialisation, the CSTA Services Interface accepts connections from CTI servers on the communications port you specify and sets up the required links. CSTA_SERVICES.DLL notifies link state changes (identifying the Link ID) and provides ASN.1 and ACSE error reports by calling the management status functions in your application:

- csa_fLinkUp
- csa_fLinkDown
- csa_fErrorReport

Chapter 4 describes the management status functions.

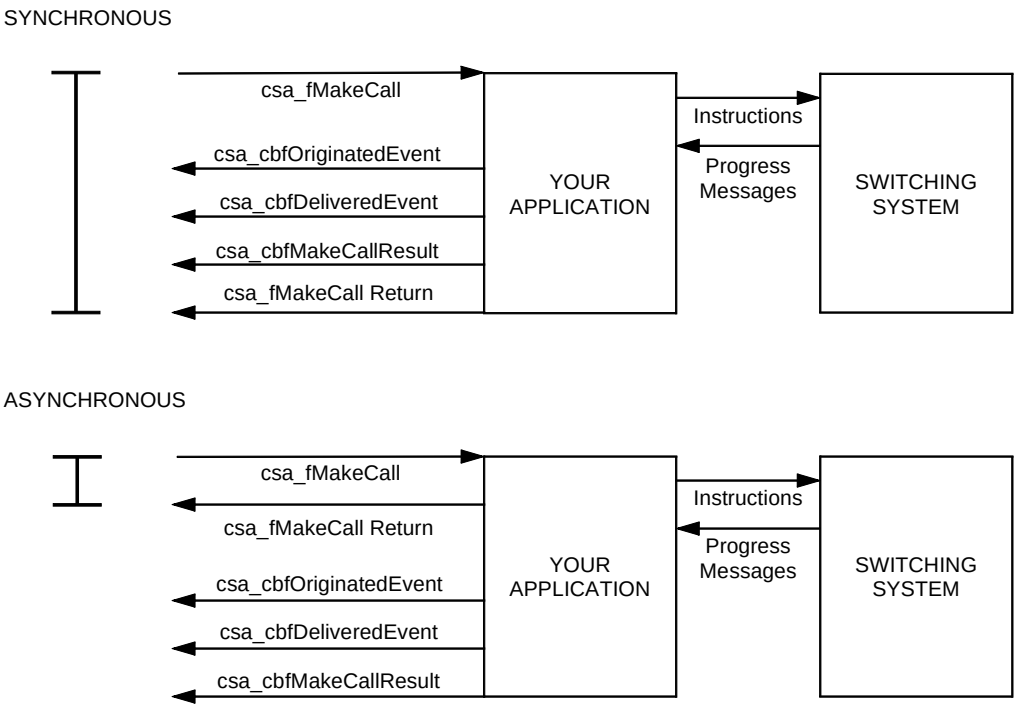
2.4.3: Synchronous and Asynchronous Operation Using Threads

The CSTA Services Interface functions operate synchronously, which means that they block the calling code (CSTA_SERVICES.DLL) until they return. When blocked, CSTA_SERVICES.DLL cannot process any incoming service requests from the CTI server on the same link.

Waiting for each function to return before another function can be processed may not be efficient for your switching system, so you might want to use an asynchronous approach. For example, your application might receive a csa_fMakeCall request and synchronously return a csa_fMakeCall return status code followed by a series of asynchronous call progress event messages.

Using simplified examples, Figure 2.1 contrasts synchronous and asynchronous operations. With time running from the top to the bottom in each case, the bar ⊥ represents how long CSTA_SERVICES.DLL is blocked. Note that you can return the function status code before returning service event reports during asynchronous processing.

Figure 2.1: Synchronous Versus Asynchronous Operation



You achieve synchronous and asynchronous processing of functions by writing multithreaded code:

- To respond synchronously to a service request, you call the response function using the same thread that called the request function. CSTA_SERVICES.DLL will block until your service request function returns its status code.
- To respond asynchronously to a service request, your request function returns its status code (normally, csa_kSuccess) immediately and calls the response function using a different thread than that which called the request function.

Since the CSTA Services Interface runs separate threads for each configured link, the same function may be called by multiple threads at the same time. Therefore, you should take special care with the design of your thread architecture. In particular, make your functions re-entrant or thread-safe if you intend to support links to more than one CTI server.

For more information about thread concepts and implementation, refer to the *Microsoft*

Programmer's Reference manual.

2.4.4: Synchronous and Asynchronous Processing Recommendations

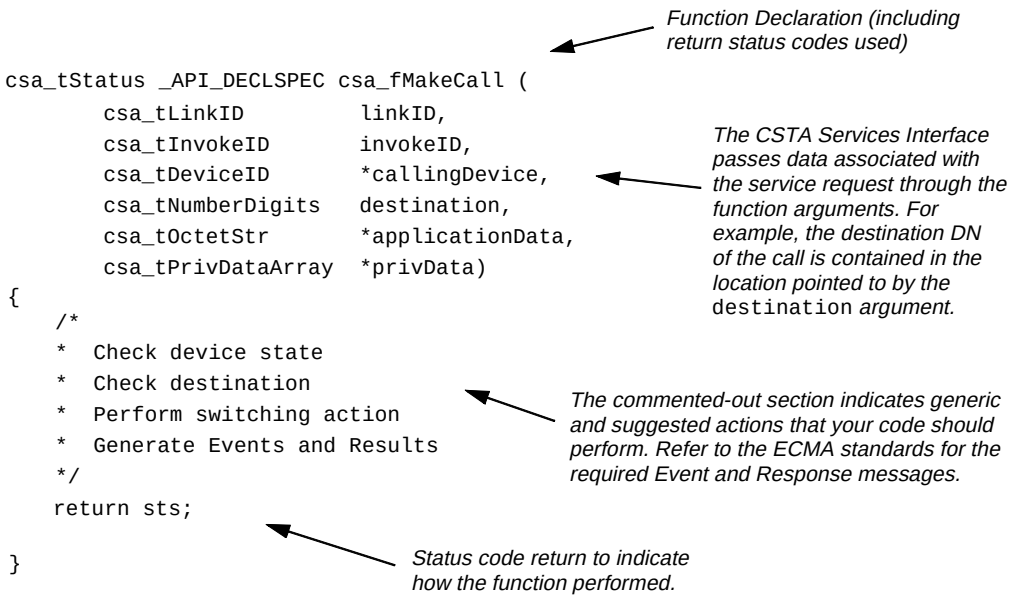
The CSTA Services Interface uses one thread for each connection to a CTI server. Each service request is read and decoded and the appropriate service request function is invoked. If your service request function is likely to take a significant time to complete, you should execute it asynchronously to prevent a backlog of service requests building up. This asynchronous execution can help to improve average response time.

2.5: Example Stub Function

Figure 2.2 shows an example stub function supplied with the CSTA Services Interface kit. The stub functions provide the basic structure and arguments you need. You complete the functions by adding code to implement the features you want and to handle the specific requirements of your switching system.

Opposite the example stub function are brief descriptions of what is in the code at each point. Afterwards, notes provide more detail on what you should do to complete the stub function with your code.

Figure 2.2: Example Stub Function



Notes:

1. An opening declaration, similar to the `csa_tStatus` declaration shown in Figure 2.2, indicates which set of return status codes will be used for the function. This declaration is shown in the function definitions in Chapters 3 - 7 and is one of:

`csa_tStatus`	Return codes for Service Request functions
`csa_tMgmtStatus`	Return codes for Management functions
`csa_tEventStatus`	Return codes for Service Event functions
`csa_tResponseStatus`	Return codes for Service Response functions

   Note that the csa_tStatus return codes are those defined by the CSTA protocol whereas the other return codes are internal to the CSTA Services Interface. Appendix A lists and describes all of the return codes.

2. In general, your code should perform these actions (taking into account the synchronous/asynchonous processing requirements described in Section 2.4.3):

   - Process the data passed by the function arguments and ready it for your switching system.
   - Invoke the switching system-specific commands that are required to perform the telephony actions requested.
   - Process return status and messages from your switching system.
   - Use the response and event functions to return the appropriate data to the CSTA Services Interface.

For every service request, you can return success or failure in two ways. To return success, you can:

- Return the csa_kSuccess status code for the service request function.
- Call the corresponding service response function.

To return failure, you can:

- Return the appropriate csa_tStatus error status code (for example, csa_kInvalidCallingDevice) for the service request function.
- Call the csa_cbfErrorResult service response function with the appropriate CSTA error code.

Chapters 3 to 7 describe all of the stub functions and function prototypes and their parameter definitions.

2.6: Example Files

Complete the example C++ source files from the kit to write your application:

Filename	Contents
CSTA_SERVICES_STUBS.C	Stub functions for each of the entry points supported by CSTA_SERVICES.DLL.
CSTA_SERVICES_SAMPLE_MAIN.C	Simple source code for your top-level application software component.

The kit also provides the following C++ definition and project files:

Filename	Contents
CSTA_SERVICES.LIB	CSTA Services Interface linkable object library.
CSTA_SERVICES_API.H	Function prototypes definitions.
CSTA_SERVICES_API_TYPES.H	Parameter type definitions.
CSTA_SERVICES_SAMPLE.DSP	Microsoft Visual C++ version 5.0 project file configured for the C source and header files listed in this table.
CSTA_SERVICES_SAMPLE.DSW	Microsoft Visual C++ version 5.0 workspace.

Use the example source files and the definition and project files as the framework to build your CSTA Services Interface application.

Your application must be accompanied by the following Dynamic Link Library (DLL) files, which provide the background functionality of the CSTA Services Interface:

Filename	Contents
CSTA_SERVICES.DLL	CSTA Services Interface functional code.
OSSAPI.DLL	Protocol support files.
OSSMEM.DLL	
SOEDBER.DLL	

2.7: Compiling Your Code

You can use the example project files supplied with the CSTA Services Interface kit to provide assistance in the linking and compiling of your code.

2.8: Operational Verification

To verify that your application is operating correctly, follow these steps:

1. Ensure that CSTA_SERVICES.DLL and the other CSTA Services Interface DLL files (listed in Section 2.6) are in the same directory as your application's EXE file.
2. Start your application.
3. Compare the events generated with the sequence of events defined in the CSTA scenarios definition (*Scenarios for Computer Supported Telecommunications Applications (CSTA) Phase II* - Technical Report ECMA TR/68).

Chapter 3: Management Operation Functions

This chapter describes all management operation functions and their parameters.

csa_fCloseLink

Format in C

```
csa_tMgmtStatus csa_fCloseLink (
                    csa_tLinkID         linkID,
                    csa_tReserved       reserved)
```

Description

This function is called by your application when the link to your switching system has failed.

This requests that the CSTA Services Interface closes down the link to the CTI server. CSTA_SERVICES.DLL calls the csa_fLinkDown function in your application to indicate that the link to the CTI server has closed down and that no further services requests will be invoked.

The CTI server may attempt to reconnect after a period of time, which will cause CSTA_SERVICES.DLL to call the csa_fLinkUp function in your application. If you wish to reject this reconnection (perhaps because the link to your switching system is still down) then your csa_fLinkUp function should return an error.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
reserved	reserved	Reserved.

csa_fInitialise

Format in C

```
csa_tMgmtStatus csa_fInitialise (
                    csa_tCommsSettings *commsSettings
                    csa_tCallbacks     *callbacks
                    csa_tReserved      reserved)
```

Description

This function is called by your application to initialise the CSTA Services Interface.

The parameters indicate the communication settings to use and the callback address for each of your management status and service request functions.

On receipt of a CSTA request, CSTA_SERVICES.DLL uses the callback address to invoke the function that matches the request. Specify the callback address as null when you do not implement a function. If a CTI server request then invokes that function, CSTA_SERVICES.DLL returns the csa_kInvalidFeature error code to the CTI server.

Arguments

Name	Type	Description
commsSettings	Pointer	A pointer to a data structure containing: • A value indicating the communications protocol: csa_kProtocol_TCPIP = 0 • A pointer to the protocol parameters data structure The protocol parameters data structure contains: • The TCP/IP port number to listen to for incoming CTI server requests (unsigned short integer) • The TCP/IP address string (unsigned characters) **Note that the TCP/IP address string is not currently supported.**

Name	Type	Description
callbacks	Pointer	A pointer to the callbacks data structure. Each entry in the callbacks data structure is a pointer to a function callback address, as follows: pFN_fLinkUp pFN_fLinkDown pFN_fErrorReport pFN_fAlternateCall pFN_fAnswerCall pFN_fCallCompletion pFN_fClearCall pFN_fClearConnection pFN_fConferenceCall pFN_fConsultationCall pFN_fDeflectCall pFN_fPickupCall pFN_fPickupGroupCall pFN_fHoldCall pFN_fMakeCall pFN_fMakePredictiveCall pFN_fQueryDevMsgWaiting pFN_fQueryDevDND pFN_fQueryDevForwarding pFN_fQueryDevInfo pFN_fQueryDevAgentState pFN_fReconnectCall pFN_fRetrieveCall pFN_fSetAgentState pFN_fSetForwarding pFN_fSetMsgWaiting pFN_fSetDoNotDisturb pFN_fTransferCall pFN_fRouteSelect pFN_fRouteEnd pFN_fMonitorStart pFN_fMonitorStop pFN_fSnapshotDevice pFN_fSendDTMF pFN_fAssociateData pFN_fSingleStepTrans pFN_fEscape
reserved	reserved	Reserved.

csa_fShutdown

Format in C

```
csa_tMgmtStatus csa_fShutdown (
                    csa_tReserved       reserved)
```

Description

This function is called by your application to shut down the CSTA Services Interface.

This disconnects any links between the CTI server and the CSTA Services Interface and invokes no further callback functions.

Arguments

Name	Type	Description
reserved	reserved	Reserved.

Chapter 4: Management Status Report Functions

This chapter describes all management status report functions and their parameters.

csa_fErrorReport

Format in C

```
csa_tMgmtStatus csa_fErrorReport (
                    csa_tErrorCode      errorCode
                    csa_tErrorText      errorText)
```

Description

This function is called by CSTA_SERVICES.DLL to report internal errors, ASN.1 parsing and encoding errors, and ACSE errors.

Although your application can ignore these error reports, and the CSTA Services Interface may continue functioning, it is good practice to log the contents to aid problem solving. For internal errors, the error text will normally identify the file and line number that is the source of the error. This data will help Dialogic diagnose a problem if you call for support.

Arguments

Name	Type	Description
errorCode	Integer	A value indicating the error type: csa_kInternalError = 0 csa_kCSTAParseError = 1 csa_kCSTAEncodeError = 2 csa_kCSTARXError = 3 csa_kCSTARXReject = 4 The sources of these errors are: 0 Code Internals 1 ASN.1 2 ASN.1 3 ACSE 4 ACSE
errorText	Pointer	A pointer to the error text character string.

csa_fLinkDown

Format in C

```
csa_tMgmtStatus csa_fLinkDown (
                    csa_tLinkID         linkID
                    csa_tReserved       reserved)
```

Description

This function is called by the CSTA_SERVICES.DLL whenever a CTI server disconnects from a link to the switching system.

The function should perform any link shut down actions required by your application (for example, releasing data structures).

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
reserved	reserved	Reserved.

csa_fLinkUp

Format in C

```
csa_tMgmtStatus csa_fLinkUp (
                    csa_tLinkID         linkID
                    csa_tReserved       reserved)
```

Description

This function is called by the CSTA_SERVICES.DLL whenever a CTI server attempts to connect to the switching system.

The function should perform any initialisation required by your application (for example, setting up data structures). The link ID is used by other functions and should be supplied when you generate responses and events.

If you did not set up the link to your switching system during initialisation, your application may use csa_fLinkUp to set up the link to your switching system now. Do not block the CTI server connection attempt for 10 or more seconds as the connection attempt may be aborted.

When a link to the switching system has not been established (either during initialisation or with this function) your application should return an error. Returning an error directs the CSTA_SERVICES.DLL to reject the connection request from the CTI server.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
reserved	reserved	Reserved.

Chapter 5: Service Request Functions

This chapter gives detailed descriptions of all service request functions and their parameters.

The service request functions are invoked on receipt of CSTA requests from the CTI server.

csa_fAlternateCall

Format in C

```
csa_tStatus csa_fAlternateCall (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tConnectionID    *activeConnection,
                csa_tConnectionID    *heldConnection,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fAlternateCall function places the active call on hold and retrieves a previously held call.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
activeConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the active call • The device identifier data structure for the active call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
heldConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the held call • The device identifier data structure for the held call (the device identifier data structure is described in the **activeConnection** argument)
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fAnswerCall

Format in C

```
csa_tStatus csa_tStatus csa_fAnswerCall (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *connectionToAnswer,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fAnswerCall function connects an alerting or queued call. The call must be associated with a device that can answer a call without user intervention.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
connectionToAnswer	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the call • The device identifier data structure for the call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fAssociateData

Format in C

```
csa_tStatus csa_fAssociateData (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tConnectionID    *existingCall,
                csa_tOctetStr        *applicationData,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fAssociateData function associates data with a call. This can be data such as an account code or an authorisation code.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
existingCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the call with which data will be associated • The device identifier data structure for the device performing the association The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fCallCompletion

Format in C

```
csa_tStatus csa_fCallCompletion (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tConnectionID    *callerConnection,
                csa_tComplete        completionType,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fCallCompletion function completes a call that may otherwise fail because the destination device is busy or not answering. By indicating a completion type, the function defines how the call should react to the device being busy or not answering.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
callerConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the call to complete • The device identifier data structure for the device performing the completion The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
completionType	Integer	A value indicating the type of call completion request: csa_kCampon = 1 csa_kCallback = 2 csa_kIntrude = 3
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fClearCall

Format in C

```
csa_tStatus csa_fClearCall (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *callToClear,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fClearCall function releases all devices from a call and eliminates the call itself.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
callToClear	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the call to clear • The device identifier data structure for the device initiating the clear The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fClearConnection

Format in C

```
csa_tStatus csa_fClearConnection (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *connectionToClear,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fClearConnection function releases a specific device from a call and leaves the connection in the Null state.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
connectionToClear	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the call • The device identifier data structure for the device leaving the call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fConferenceCall

Format in C

```
csa_tStatus csa_fConferenceCall (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tConnectionID    *activeConnection,
                csa_tConnectionID    *heldConnection,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fConferenceCall function creates a conference between a held call and an active call at a device. The two calls are merged into a single call and the two connections are resolved into a single connection in the Connected state.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
activeConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the active call • The device identifier data structure for the active call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
heldConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the held call • The device identifier data structure for the held call (the device identifier data structure is described in the **activeConnection** argument)
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fConsultationCall

Format in C

```
csa_tStatus csa_fConsultationCall (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *existingConnection,
                csa_tDeviceID           *consultedDevice,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fConsultationCall function places the existing call on hold and makes a new call to another device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
existingConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the existing call • The device identifier data structure for the existing connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
consultedDevice	Pointer	A pointer to the consulted device identifier data structure. The device identifier data structure is described in the **existingConnection** argument
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fDeflectCall

Format in C

```
csa_tStatus csa_fDeflectCall (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *connectionToDeflect,
                csa_tDeviceID           *newDestination,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fDeflectCall function deflects a call alerting or queued at a device to another device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.

Name	Type	Description
connectionToDeflect	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection to deflect • The device identifier data structure for the connection to deflect The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
newDestination	Pointer	A pointer to the destination device identifier data structure. The device identifier data structure is described in the **connectionToDeflect** argument.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fEscape

Format in C

```
csa_tStatus csa_fEscape (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fEscape function allows private data to be exchanged with the switching system. The kind of data exchanged depends on the characteristics of the specific switching system.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fHoldCall

Format in C

```
csa_tStatus csa_fHoldCall (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *connectionToHold,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fHoldCall function places an existing call on hold.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
connectionToHold	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection to hold • The device identifier data structure for the connection to hold The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fMakeCall

Format in C

```
csa_tStatus csa_fMakeCall (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tDeviceID        *callingDevice,
                csa_tNumberDigits    destination,
                csa_tOctetStr        *applicationData,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fMakeCall function makes a call between two devices.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
callingDevice	Pointer	A pointer to the calling device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
destination	Pointer	A pointer to the destination number string.

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fMakePredictiveCall

Format in C

```
csa_tStatus csa_fMakePredictiveCall (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tDeviceID        *callingDevice,
                csa_tNumberDigits    destination,
                csa_tAllocState      allocation,
                csa_tOctetStr        *applicationData,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fMakePredictiveCall function makes a call between two devices. The connection is made only when the called device is in the state defined by the value in the **allocation** argument.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
callingDevice	Pointer	A pointer to the calling device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
destination	Pointer	A pointer to the destination number string.

Name	Type	Description
allocation	Integer	A value indicating the call allocation state at which the call is delivered to the agent: csa_kDelivered = 0 csa_kEstablished = 1
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fMonitorStart

Format in C

```
csa_tStatus csa_fMonitorStart (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tDeviceID        *device,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fMonitorStart function requests the start of reporting of telephony events at a device.

Following this message, you should call service event functions to inform the CSTA Services Interface of events associated with the device. These events are then passed back to the CSTA environment.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
device	Pointer	A pointer to the device identifier data structure for the device to be monitored containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fMonitorStop

Format in C

```
csa_tStatus csa_fMonitorStop (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tMonXRef         monitorID,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fMonitorStop function stops the reporting of telephony events at the device monitored with the specified monitor ID.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
monitorID	Unsigned integer	The monitor ID cross-reference number.
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fPickupCall

Format in C

```
csa_tStatus csa_fPickupCall (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *connectionToPickup,
                csa_tDeviceID           *requestingDevice,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fPickupCall function allows a ringing device to be answered by another device. The number of the ringing device must be specified.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.

Name	Type	Description
connectionToPickup	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection to pickup • The device identifier data structure for the connection to pickup The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
requestingDevice	Pointer	A pointer to the requesting device's information data structure. The device identifier data structure is described in the **connectionToPickup** argument.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fPickupGroupCall

Format in C

```
csa_tStatus csa_fPickupGroupCall (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tDeviceID        *requestingDevice,
                csa_tOctetStr        *applicationData,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fPickupCall function allows a ringing device in a group to be answered by another device in the same group. Because the devices are in the same group, the number of the ringing device does not need to be specified.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
requestingDevice	Pointer	A pointer to the requesting device's information data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fQueryDevAgentState

Format in C

```
csa_tStatus csa_fQueryDevAgentState (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *deviceToQuery,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fQueryDevAgentState function requests information on the state of an agent at a device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
deviceToQuery	Pointer	A pointer to the device identifier data structure for the device to query containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fQueryDevDND

Format in C

```
csa_tStatus csa_fQueryDevDND (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *deviceToQuery,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fQueryDevDND function requests the do-not-disturb setting of a device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
deviceToQuery	Pointer	A pointer to the device identifier data structure for the device to query containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fQueryDevForwarding

Format in C

```
csa_tStatus csa_fQueryDevForwarding (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *deviceToQuery,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fQueryDevForwarding function requests the forwarding status of a device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
deviceToQuery	Pointer	A pointer to the device identifier data structure for the device to query containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fQueryDevInfo

Format in C

```
csa_tStatus csa_fQueryDevInfo (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *deviceToQuery,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fQueryDevInfo function requests the class and type of a device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
deviceToQuery	Pointer	A pointer to the device identifier data structure for the device to query containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fQueryDevMsgWaiting

Format in C

```
csa_tStatus csa_fQueryDevMsgWaiting (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *deviceToQuery,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fQueryDevMsgWaiting function requests whether there are any messages waiting at a device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
deviceToQuery	Pointer	A pointer to the device identifier data structure for the device to query containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fReconnectCall

Format in C

```
csa_tStatus csa_fReconnectCall (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tConnectionID    *connectionToClear,
                csa_tConnectionID    *connectionToRetrieve,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fReconnectCall function clears an active call and retrieves a held call.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
connectionToClear	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection to clear • The device identifier data structure for the connection to clear The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
connectionToRetrieve	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection to retrieve • The device identifier data structure for the connection to retrieve (the device identifier data structure is described in the **connectionToClear** argument)
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fRetrieveCall

Format in C

```
csa_tStatus csa_fRetrieveCall (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *connectionToRetrieve,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fRetrieveCall function connects a held call.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
connectionToRetrieve	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection to retrieve • The device identifier data structure for the connection to retrieve The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fRouteEnd

Format in C

```
csa_tStatus csa_fRouteEnd (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tRouteXRef       routeRef,
                csa_tStatus          errorNum,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fRouteEnd function is used to end a routing process. The call may have been successfully routed or cleared, or there may have been no route provided within a time limit.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
routeRef	Unsigned integer	The routing cross-reference ID number.
errorNum	Integer	A value indicating the error code to be returned. Section A.3 contains a full description of the error codes.
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fRouteSelect

Format in C

```
csa_tStatus csa_fRouteSelect (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tRouteXRef          routeRef,
                csa_tDeviceID           *routeSelected,
                csa_tBoolean            routeUsed,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fRouteSelect function changes the routing destination that was set by a previous routing, rerouting request, or default route. By setting the routeUsed boolean variable, this function can also ask for notification of when the route is used.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
routeRef	Unsigned integer	The routing cross-reference ID number.
routeSelected	Pointer	A pointer to the new destination device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
routeUsed	Unsigned integer	A boolean variable (0 = False) to indicate whether notification of use of the route is requested or not.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fSendDTMF

Format in C

```
csa_tStatus csa_fSendDTMF (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *connectionToSendTones,
                csa_tNumberDigits       digits,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fSendDTMF function requests the generation of DTMF tones onto a connection.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.

Name	Type	Description
connectionToSendTones	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection on which to send tones • The device identifier data structure for the connection on which to send tones The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
digits	Pointer	A pointer to a string of DTMF digits.
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fSetAgentState

Format in C

```
csa_tStatus csa_fSetAgentState (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *device,
                csa_tAgentStateReq      agentState,
                csa_tOctetStr           *agentID,
                csa_tOctetStr           *agentPassword,
                csa_tDeviceID           agentGroup,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fSetAgentState function changes the setting of an agent state.

The agentID, agentPassword, and agentGroup are only set when the agent state is logged on or logged off.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
device	Pointer	A pointer to the device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
agentState	Integer	A value indicating the agent state to be set: csa_kReqLoggedOn = 1 csa_kReqLoggedOff = 2 csa_kReqNotReady = 3 csa_kReqReady = 4 csa_kReqWorkNotReady = 5 csa_kReqWorkReady = 6 csa_kReqBusy = 7 csa_kReqWorkingAfterCall = 8
agentID	Pointer	A pointer to the data structure containing the agent ID.
agentPassword	Pointer	A pointer to the data structure containing the agent password.
agentGroup	Pointer	A pointer to the agent group device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fSetDoNotDisturb

Format in C

```
csa_tStatus csa_fSetDoNotDisturb (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tDeviceID        *device,
                csa_tBoolean         doNotDisturb,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fSetDoNotDisturb function changes the do-not-disturb setting of a device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
device	Pointer	A pointer to the device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
doNotDisturb	Unsigned integer	A boolean variable (0 = False) indicating whether do-not-disturb is to be set on.

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fSetForwarding

Format in C

```
csa_tStatus csa_fSetForwarding (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tDeviceID        *device,
                csa_tForwardType     forwardType,
                csa_tNumberDigits    destination,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fSetForwarding function sets up call forwarding on a device, defines the type of forwarding, and specifies the forwarding destination.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
device	Pointer	A pointer to the device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
forwardType	Integer	A value indicating the type of call forwarding: csa_kForwardImmediateOn = 0 csa_kForwardImmediateOff = 1 csa_kForwardBusyOn = 2 csa_kForwardBusyOff = 3 csa_kForwardNoAnsOn = 4 csa_kForwardNoAnsOff = 5 csa_kForwardBusyIntOn = 6 csa_kForwardBusyIntOff = 7 csa_kForwardBusyExtOn = 8 csa_kForwardBusyExtOff = 9 csa_kForwardNoAnsIntOn = 10 csa_kForwardNoAnsIntOff = 11 csa_kForwardNoAnsExtOn = 12 csa_kForwardNoAnsExtOff = 13 csa_kForwardImmIntOn = 14 csa_kForwardImmIntOff = 15 csa_kForwardImmExtOn = 16 csa_kForwardImmExtOff = 17
destination	Pointer	A pointer to the destination number string.
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fSetMsgWaiting

Format in C

```
csa_tStatus csa_fSetMsgWaiting (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tDeviceID        *device,
                csa_tBoolean         msgWaiting,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fSetMsgWaiting function sets the message waiting indicator on a device on or off.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
device	Pointer	A pointer to the device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
msgWaiting	Unsigned integer	A boolean variable (0 = False) indicating whether the message waiting indicator is to be set on.

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fSingleStepTransfer

Format in C

```
csa_tStatus csa_fSingleStepTransfer (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *activeConnection,
                csa_tDeviceID           *deviceToTransferTo,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fSingleStepTransfer function transfers an active call to another device without first putting the active call on hold or performing a consultation call.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.

Name	Type	Description
activeConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the active connection • The device identifier data structure for the active connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
deviceToTransferTo	Pointer	A pointer to the target device identifier data structure. The device identifier data structure is described in the **activeConnection** argument.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fSnapshotDevice

Format in C

```
csa_tStatus csa_fSnapshotDevice (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *device,
                csa_tPrivDataArray      *privData)
```

Description

The csa_fSnapshotDevice function requests status information on calls associated with a device. For each call, the information includes the call's identification and local connection state or simple call state.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
device	Pointer	A pointer to the device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_fTransferCall

Format in C

```
csa_tStatus csa_fTransferCall (
                csa_tLinkID          linkID,
                csa_tInvokeID        invokeID,
                csa_tConnectionID    *activeConnection,
                csa_tConnectionID    *heldConnection,
                csa_tPrivDataArray   *privData)
```

Description

The csa_fTransferCall function transfers a held call to another device. An active call is in progress between the holding device and the destination device.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
activeConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the active connection • The device identifier data structure for the active connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
heldConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the held connection • The device identifier data structure for the held connection (the device identifier data structure is described in the **activeConnection** argument) The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
privData	Pointer	A pointer to an array of three data structures containing private data to be passed to the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

Chapter 6: Service Response Functions

This chapter describes all response generation functions and their parameters.

csa_cbfAlternateCallResult

Format in C

```
csa_tResponseStatus csa_cbfAlternateCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fAlternateCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfAnswerCallResult

Format in C

```
csa_tResponseStatus csa_cbfAnswerCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fAnswerCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfAssociateDataResult

Format in C

```
csa_tResponseStatus csa_cbfAssociateDataResult (
                    csa_tLinkID          linkID,
                    csa_tInvokeID        invokeID,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to return the response to a csa_fAssociateData service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfCallCompletionResult

Format in C

```
csa_tResponseStatus csa_cbfCallCompletionResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fCallCompletion service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the connection to the CTI serverserver.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfClearCallResult

Format in C

```
csa_tResponseStatus csa_cbfClearCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fClearCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfClearConnectionResult

Format in C

```
csa_tResponseStatus csa_cbfClearConnectionResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fClearConnection service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfConferenceCallResult

Format in C

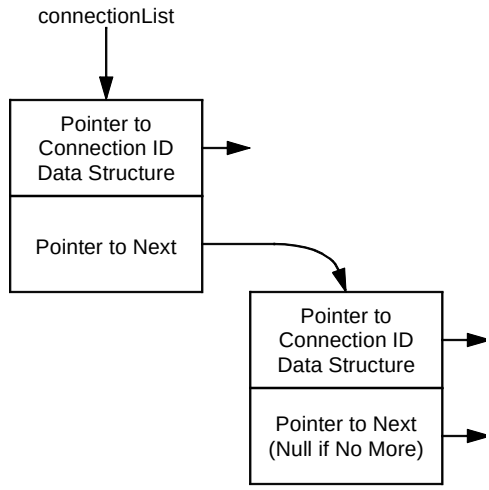
```
csa_tResponseStatus csa_cbfConferenceCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *resultingConnection,
                csa_tConnIDList         *connectionList
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fConferenceCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
resultingConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the conference call • The device identifier data structure for the conference call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
connectionList	Pointer	A pointer to a linked list of connection ID information. Each entry in the list contains: • A pointer to the data structure described for the **resultingConnection** argument • A pointer to the next entry (null if there are no more entries) The linked list has this structure: 
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfConsultationCallResult

Format in C

```
csa_tResponseStatus csa_cbfConsultationCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *initiatedConnection,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fConsultationCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
initiatedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the consultation call • The device identifier data structure for the consultation call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfDeflectCallResult

Format in C

```
csa_tResponseStatus csa_cbfDeflectCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fDeflectCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfErrorResult

Format in C

```
csa_tResponseStatus csa_cbfErrorResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tStatus             error)
```

Description

Use this function to respond when there has been an operations, state, system resource, subscription resource, performance, or security error during performance of a service request.

This function is also invoked by CSTA_SERVICES.DLL on your behalf if your application simply returns an error status code after processing a service request function.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
error	Integer	A value indicating the specific error. Section A.3 lists the error values and describes the error conditions to which they relate.

csa_cbfEscapeResult

Format in C

```
csa_tResponseStatus csa_cbfEscapeResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fEscape service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfHoldCallResult

Format in C

```
csa_tResponseStatus csa_cbfHoldCallResult (
                    csa_tLinkID          linkID,
                    csa_tInvokeID        invokeID,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to return the response to a csa_fHoldCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfMakeCallResult

Format in C

```
csa_tResponseStatus csa_cbfMakeCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *initiatedConnection,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fMakeCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
initiatedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the initiated connection • The device identifier data structure for the initiated connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfMakePredictiveCallResult

Format in C

```
csa_tResponseStatus csa_cbfMakePredictiveCallResult (
        csa_tLinkID         linkID,
        csa_tInvokeID       invokeID,
        csa_tConnectionID   *initiatedConnection,
        csa_tPrivDataArray  *privData)
```

Description

Use this function to return the response to a csa_fMakePredictiveCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
initiatedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the initiated connection • The device identifier data structure for the initiated connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfMonitorStartResult

Format in C

```
csa_tResponseStatus csa_cbfMonitorStartResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tMonXRef            monID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fMonitorStart service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
monID	Unsigned integer	The monitoring cross-reference ID.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfMonitorStopResult

Format in C

```
csa_tResponseStatus csa_cbfMonitorStopResult (
                    csa_tLinkID          linkID,
                    csa_tInvokeID        invokeID,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to return the response to a csa_fMonitorStop service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfPickupCallResult

Format in C

```
csa_tResponseStatus csa_cbfPickupCallResult (
                    csa_tLinkID          linkID,
                    csa_tInvokeID        invokeID,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to return the response to a csa_fPickupCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfPickupGroupCallResult

Format in C

```
csa_tResponseStatus csa_cbfPickupGroupCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fPickupGroupCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfQueryDevAgentStateResult

Format in C

```
csa_tResponseStatus csa_cbfQueryDevAgentStateResult (
        csa_tLinkID         linkID,
        csa_tInvokeID       invokeID,
        csa_tAgentState     state,
        csa_tPrivDataArray  *privData)
```

Description

Use this function to return the response to a csa_fQueryDevAgentState service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
state	Integer	A value indicating the agent state: csa_kASNotReady = 0 csa_kASNull = 1 csa_kASReady = 2 csa_kASWorkNotReady = 3 csa_kASWorkReady = 4 csa_kASBusy = 5 csa_kASWorkingAfterCall = 6
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfQueryDevDNDResult

Format in C

```
csa_tResponseStatus csa_cbfQueryDevDNDResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tBoolean            doNotDisturb,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fQueryDevDND service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
doNotDisturb	Unsigned integer	A boolean variable (0 = False) indicating whether do-not-disturb is set on the device.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfQueryDevForwardingResult

Format in C

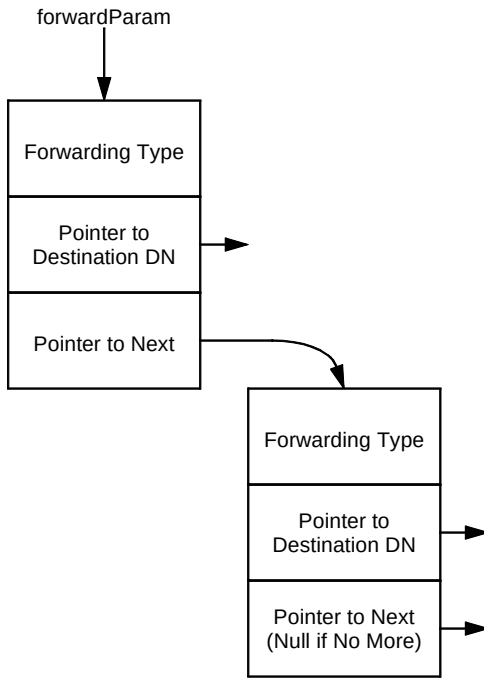
```
csa_tResponseStatus csa_cbfQueryDevForwardingResult (
                    csa_tLinkID             linkID,
                    csa_tInvokeID           invokeID,
                    csa_tForwardParam       *forwardParams,
                    csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fQueryDevForwarding service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.

Name	Type	Description
forwardParams	Pointer	A pointer to a linked list of forwarding parameters. Each entry in the list contains: • Forwarding type (integer). The possible values are described after the linked list structure diagram. • Pointer to the location of the destination DN. • Pointer to the next entry (null if there are no more entries) The linked list has this structure: 

Name	Type	Description
		The forwarding type is a value indicating one of the following: csa_kForwardImmediateOn = 0 csa_kForwardImmediateOff = 1 csa_kForwardBusyOn = 2 csa_kForwardBusyOff = 3 csa_kForwardNoAnsOn = 4 csa_kForwardNoAnsOff = 5 csa_kForwardBusyIntOn = 6 csa_kForwardBusyIntOff = 7 csa_kForwardBusyExtOn = 8 csa_kForwardBusyExtOff = 9 csa_kForwardNoAnsIntOn = 10 csa_kForwardNoAnsIntOff = 11 csa_kForwardNoAnsExtOn = 12 csa_kForwardNoAnsExtOff = 13 csa_kForwardImmIntOn = 14 csa_kForwardImmIntOff = 15 csa_kForwardImmExtOn = 16 csa_kForwardImmExtOff = 17
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfQueryDevInfoResult

Format in C

```
csa_tResponseStatus csa_cbfQueryDevInfoResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tDeviceID           *deviceID,
                csa_tDeviceType         deviceType,
                csa_tDeviceClass        deviceClass,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fQueryDevInfo service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
deviceID	Pointer	A pointer to the device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
deviceType	Integer	A value indicating the device type: csa_kDevTypeStation = 0 csa_kDevTypeLine = 1 csa_kDevTypeButton = 2 csa_kDevTypeACD = 3 csa_kDevTypeTrunk = 4 csa_kDevTypeOperator = 5 csa_kDevTypeStationGroup = 16 csa_kDevTypeLineGroup = 17 csa_kDevTypeButtonGroup = 18 csa_kDevTypeACDGroup = 19 csa_kDevTypeTrunkGroup = 20 csa_kDevTypeOperatorGroup = 21 csa_kDevTypeOther = 255
deviceClass	Integer	A value indicating the device class: csa_kDevClassVoice = 0x80 csa_kDevClassData = 0x40 csa_kDevClassImage = 0x20 csa_kDevClassOther = 0x10
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfQueryDevMsgWaitingResult

Format in C

```
csa_tResponseStatus csa_cbfQueryDevMsgWaitingResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tBoolean            msgWaiting,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fQueryDevMsgWaiting service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
msgWaiting	Unsigned integer	A boolean variable (0 = False) indicating whether a message is waiting or not.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfReconnectCallResult

Format in C

```
csa_tResponseStatus csa_cbfReconnectCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fReconnectCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfRetrieveCallResult

Format in C

```
csa_tResponseStatus csa_cbfRetrieveCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fRetrieveCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfSendDTMFResult

Format in C

```
csa_tResponseStatus csa_cbfSendDTMFResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fSendDTMF service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfSetAgentStateResult

Format in C

```
csa_tResponseStatus csa_cbfSetAgentStateResult (
                    csa_tLinkID          linkID,
                    csa_tInvokeID        invokeID,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to return the response to a csa_fSetAgentState service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfSetDoNotDisturbResult

Format in C

```
csa_tResponseStatus csa_cbfSetDoNotDisturbResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fSetDoNotDisturb service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfSetForwardingResult

Format in C

```
csa_tResponseStatus csa_cbfSetForwardingResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fSetForwarding service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfSetMsgWaitingResult

Format in C

```
csa_tResponseStatus csa_cbfSetMsgWaitingResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fSetMsgWaiting service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfSingleStepTransferResult

Format in C

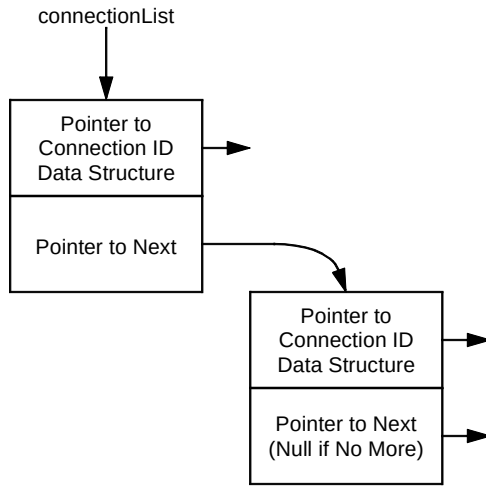
```
csa_tResponseStatus csa_cbfSingleStepTransferResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *transferredCall
                csa_tConnIDList         *connectionList
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fSingleStepTransfer service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
transferredCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the transferred call • The device identifier data structure for the transferred call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
connectionList	Pointer	A pointer to a linked list of connection ID information. Each entry in the list contains: • A pointer to the data structure described for the **transferredCall** argument • A pointer to the next entry (null if there are no more entries) The linked list has this structure: 
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfSnapshotDeviceResult

Format in C

```
csa_tResponseStatus csa_cbfSnapshotDeviceResult (
                    csa_tLinkID             linkID,
                    csa_tInvokeID           invokeID,
                    csa_tCallStateList      *callStateList,
                    csa_tPrivDataArray      *privData)
```

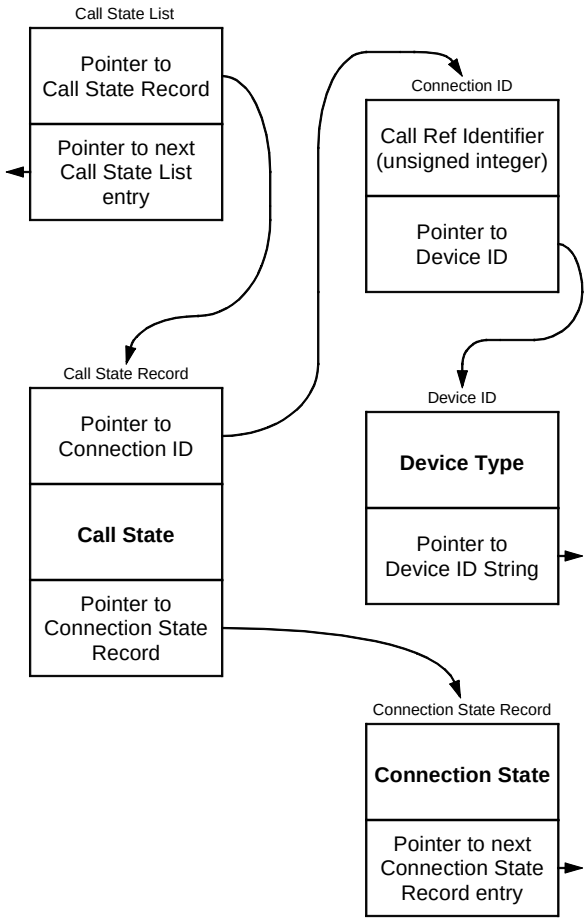
Description

Use this function to return the response to a csa_fSnapshotDevice service request.

The CSTA Services Interface supports simple call states for calls involving only two devices and connection lists for calls involving more than two devices.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.

Name	Type	Description
callStateList	Pointer	A pointer to the Call State List data structure. In the Call State Record, the pointer to the Connection State Record is null if the call involves only two devices. The values of **Call State**, **Connection State**, and **Device Type** are defined after the Call State List structure diagram: 

Name	Type	Description
		The Call State is indicated by the value of an integer: css_kCallStateNull = 0 css_kCallStatePending = 1 css_kCallStateOriginated = 3 css_kCallStateDelivered = 35 css_kCallStateDeliveredHeld = 36 css_kCallStateReceived = 50 css_kCallStateEstablished = 51 css_kCallStateEstablishedHeld = 52 css_kCallStateReceivedOnHold = 66 css_kCallStateEstablishedOnHold = 67 css_kCallStateQueued = 83 css_kCallStateQueuedHeld = 84 css_kCallStateFailed = 99 css_kCallStateFailedHeld = 100 The Connection State is indicated by the value of an integer: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6 The Device Type is indicated by the value of an integer: csa_kDialingNumber = 0 csa_kDeviceNumber = 1
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfTransferCallResult

Format in C

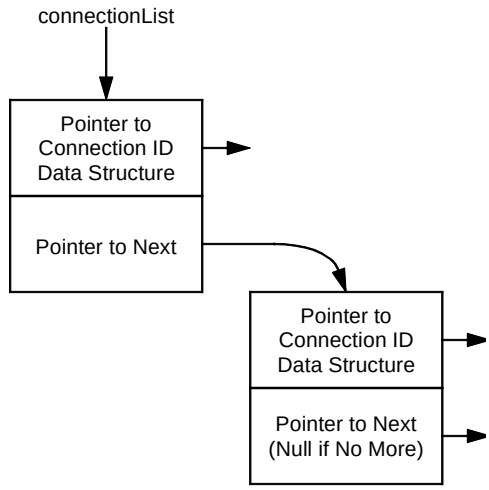
```
csa_tResponseStatus csa_cbfTransferCallResult (
                csa_tLinkID             linkID,
                csa_tInvokeID           invokeID,
                csa_tConnectionID       *transferredCall,
                csa_tConnIDList         *connectionList,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to return the response to a csa_fTransferCall service request.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
invokeID	Unsigned integer	The operation request sequence number.
transferredCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the call transferred • The device identifier data structure for the call transferred The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
connectionList	Pointer	A pointer to a linked list of connection ID information. Each entry in the list contains: • A pointer to the data structure described for the **transferredCall** argument • A pointer to the next entry (null if there are no more entries) The linked list has this structure: 
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

Chapter 7: Service Event Functions

This chapter describes all service event functions and their parameters.

csa_cbfAgentStateEvent

Format in C

```
csa_tEventStatus csa_cbfAgentStateEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tAgentStateReq      agentState,
                csa_tDeviceID           *agentDevice,
                csa_tOctetStr           *agentID,
                csa_tDeviceID           *agentGroup,
                csa_tOctetStr           *password,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back agent state events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.
agentState	Integer	A value indicating the agent's state: csa_kReqLoggedOn = 1 csa_kReqLoggedOff = 2 csa_kReqNotReady = 3 csa_kReqReady = 4 csa_kReqWorkNotReady = 5 csa_kReqWorkReady = 6 csa_kReqBusy = 7 csa_kReqWorkingAfterCall = 8

Name	Type	Description
agentDevice	Pointer	A pointer to the agent device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
agentID	Pointer	A pointer to a data structure containing details of the agent ID octet string. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
agentGroup	Pointer	A pointer to the agent group device identifier data structure. The device identifier data structure is described in the **agentDevice** argument.
password	Pointer	A pointer to a data structure containing details of the agent password octet string. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfCallClearedEvent

Format in C

```
csa_tEventStatus csa_cbfCallClearedEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *clearedCall,
                csa_tConnState          localConnectionState,
                csa_tEventCause         eventCause,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call cleared events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID of the connection to the CTI Server.
monID	Unsigned integer	The monitoring cross-reference ID.
clearedCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the cleared call • The device identifier data structure for the cleared call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfCallInformationEvent

Format in C

```
csa_tEventStatus csa_cbfCallInformationEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *enteringConnection,
                csa_tDeviceID           *enteringDevice,
                csa_tOctetStr           *accountInfo,
                csa_tOctetStr           *authCode,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call information events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.

Name	Type	Description
enteringConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the connection that entered the call information • The device identifier data structure for the connection that entered the call information The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
enteringDevice	Pointer	A pointer to the device identifier data structure. The device identifier data structure is described in the **enteringConnection** argument.
accountInfo	Pointer	A pointer to the data structure containing details of the account data octet string sent on the call. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
authCode	Pointer	A pointer to the data structure containing details of the octet string authorization code. The data structure is described in the **accountInfo** argument.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure is described in the **accountInfo** argument.

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfConferencedEvent

Format in C

```
csa_tEventStatus csa_cbfConferenceEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *primaryOldCall,
                csa_tConnectionID       *secondaryOldCall,
                csa_tDeviceID           *confController,
                csa_tDeviceID           *addedDevice,
                csa_tConnIDList         *confConnections,
                csa_tConnState          localConnectionState,
                csa_tEventCause         eventCause,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

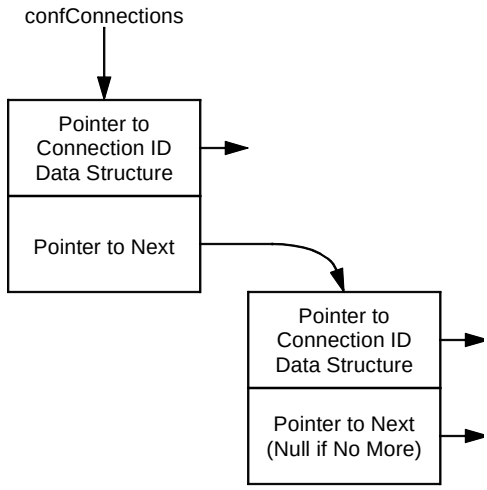
Description

Use this function to report back conference events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.

Name	Type	Description
primaryOldCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the primary old call • The device identifier data structure for the primary old call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
secondaryOldCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the secondary old call • The device identifier data structure for the secondary old call (the device identifier data structure is described in the **primaryOldCall** argument)
confController	Pointer	A pointer to the conference controller device identifier data structure. The device identifier data structure is described in the **primaryOldCall** argument.
addedDevice	Pointer	A pointer to the added device identifier data structure. The device identifier data structure is described in the **primaryOldCall** argument.

Name	Type	Description
confConnections	Pointer	A pointer to a linked list of connection ID information. Each entry in the list contains: • A pointer to the data structure described for the **primaryOldCall** argument • A pointer to the next entry (null if there are no more entries) The linked list has this structure: 
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes.

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfConnectionClearedEvent

Format in C

```
csa_tEventStatus csa_cbfConnectionClearedEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *droppedConnection,
                csa_tDeviceID           *releasingDevice,
                csa_tConnState          localConnectionState,
                csa_tEventCause         eventCause,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back connection cleared events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.

Name	Type	Description
droppedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the dropped connection • The device identifier data structure for the dropped connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
releasingDevice	Pointer	A pointer to the releasing device identifier data structure. The device identifier data structure is described in the **primaryOldCall** argument.
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfDeliveredEvent

Format in C

```
csa_tEventStatus csa_cbfDeliveredEvent (
                    csa_tLinkID             linkID,
                    csa_tMonXRef            monID,
                    csa_tConnectionID       *alertingConnection,
                    csa_tDeviceID           *alertingDevice,
                    csa_tDeviceID           *callingDevice,
                    csa_tDeviceID           *calledDevice,
                    csa_tDeviceID           *lastRedirDevice,
                    csa_tConnectionID       *originatingConnection,
                    csa_tConnState          localConnectionState,
                    csa_tEventCause         eventCause,
                    csa_tOctetStr           *applicationData,
                    csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call delivered events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor cross-reference number.

Name	Type	Description
alertingConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the alerting connection • The device identifier data structure for the alerting connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
alertingDevice	Pointer	A pointer to the alerting device identifier data structure. The device identifier data structure is described in the **alertingConnection** argument.
callingDevice	Pointer	A pointer to the calling device identifier data structure. The device identifier data structure is described in the **alertingConnection** argument.
calledDevice	Pointer	A pointer to the called device identifier data structure. The device identifier data structure is described in the **alertingConnection** argument.
lastRedirDevice	Pointer	A pointer to the last redirected destination device identifier data structure. The device identifier data structure is described in the **alertingConnection** argument.

Name	Type	Description
originatingConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the originating connection • The device identifier data structure for the originating connection (the device identifier data structure is described in the **alertingConnection** argument)
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfDivertedEvent

Format in C

```
csa_tEventStatus csa_cbfDivertedEvent (
                    csa_tLinkID             linkID,
                    csa_tMonXRef            monID,
                    csa_tConnectionID       *divertConnection,
                    csa_tDeviceID           *divertingDevice,
                    csa_tDeviceID           *newDestinationDevice,
                    csa_tConnState          localConnectionState,
                    csa_tEventCause         eventCause,
                    csa_tOctetStr           *applicationData,
                    csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call diverted events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor cross-reference number.

Name	Type	Description
divertConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the diverted connection • The device identifier data structure for the diverted connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
divertingDevice	Pointer	A pointer to the diverting device identifier data structure. The device identifier data structure is described in the **divertConnection** argument.
newDestinationDevice	Pointer	A pointer to the new destination device identifier data structure. The device identifier data structure is described in the **divertConnection** argument.
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfEstablishedEvent

Format in C

```
csa_tEventStatus csa_cbfEstablishedEvent (
                    csa_tLinkID          linkID,
                    csa_tMonXRef         monID,
                    csa_tConnection      *establishedConnection,
                    csa_tDeviceID        *answeringDevice,
                    csa_tDeviceID        *callingDevice,
                    csa_tDeviceID        *calledDevice,
                    csa_tDeviceID        *lastRedirDevice,
                    csa_tConnectionID    *originatingConnection,
                    csa_tConnState       localConnectionState,
                    csa_tEventCause      eventCause,
                    csa_tOctetStr        *applicationData,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to report back call established events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.

Name	Type	Description
establishedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the established connection • The device identifier data structure for the established connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
answeringDevice	Pointer	A pointer to the answering device identifier data structure. The device identifier data structure is described in the **establishedConnection** argument.
callingDevice	Pointer	A pointer to the calling device identifier data structure. The device identifier data structure is described in the **establishedConnection** argument.
calledDevice	Pointer	A pointer to the called device identifier data structure. The device identifier data structure is described in the **establishedConnection** argument.
lastRedirDevice	Pointer	A pointer to the last redirection destination device identifier data structure. The device identifier data structure is described in the **establishedConnection** argument.

Name	Type	Description
originatingConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the originating connection • The device identifier data structure for the originating connection (the device identifier data structure is described in the **establishedConnection** argument)
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfFailedEvent

Format in C

```
csa_tEventStatus csa_cbfFailedEvent (
                    csa_tLinkID             linkID,
                    csa_tMonXRef            monID,
                    csa_tConnectionId       *failedConnection,
                    csa_tDeviceID           *failingDevice,
                    csa_tDeviceID           *calledDevice,
                    csa_tConnState          localConnectionState,
                    csa_tEventCause         eventCause,
                    csa_tOctetStr           *applicationData,
                    csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call failed events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor cross-reference number.

Name	Type	Description
failedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the failed connection • The device identifier data structure for the failed connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
failingDevice	Pointer	A pointer to the failing device identifier data structure. The device identifier data structure is described in the **failedConnection** argument.
calledDevice	Pointer	A pointer to the called device identifier data structure. The device identifier data structure is described in the **failedConnection** argument.
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfHeldEvent

Format in C

```
csa_tEventStatus csa_cbfHeldEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *heldConnection,
                csa_tDeviceID           *holdingDevice,
                csa_tConnState          localConnectionState,
                csa_tEventCause         eventCause,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call held events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor cross-reference ID number.

Name	Type	Description
heldConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the held connection • The device identifier data structure for the held connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
holdingDevice	Pointer	A pointer to the holding device identifier data structure. The device identifier data structure is described in the **heldConnection** argument.
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfNetworkReachedEvent

Format in C

```
csa_tEventStatus csa_cbfNetworkReachedEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *outboundConnection,
                csa_tDeviceID           *trunkUsed,
                csa_tDeviceID           *calledDevice,
                csa_tConnState          localConnectionState,
                csa_tEventCause         eventCause,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back network reached events to the CSTA Services Interface. This refers to the point when an outgoing call enters another network (for example, when the call reaches a trunk).

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor cross-reference ID number.

Name	Type	Description
outboundConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the outbound connection • The device identifier data structure for the outbound connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
trunkUsed	Pointer	A pointer to the trunk device identifier data structure. The device identifier data structure is described in the **outboundConnection** argument.
calledDevice	Pointer	A pointer to the called device identifier data structure. The device identifier data structure is described in the **outboundConnection** argument.
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfOriginatedEvent

Format in C

```
csa_tEventStatus csa_cbfOriginatedEvent (
                    csa_tLinkID             linkID,
                    csa_tMonXRef            monID,
                    csa_tConnectionID       *originatedConnection,
                    csa_tDeviceID           *callingDevice,
                    csa_tDeviceID           *calledDevice,
                    csa_tDeviceID           *originatingDevice,
                    csa_tConnState          localConnectionState,
                    csa_tEventCause         eventCause,
                    csa_tOctetStr           *applicationData,
                    csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call originated events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor cross-reference ID number.

Name	Type	Description
originatedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the originated connection • The device identifier data structure for the originated connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
callingDevice	Pointer	A pointer to the calling device identifier data structure. The device identifier data structure is described in the **originatedConnection** argument.
calledDevice	Pointer	A pointer to the called device identifier data structure. The device identifier data structure is described in the **originatedConnection** argument.
originatingDevice	Pointer	A pointer to the originating device identifier data structure. The device identifier data structure is described in the **originatedConnection** argument.
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfPrivateEvent

Format in C

```
csa_tEventStatus csa_cbfPrivateEvent (
                    csa_tLinkID          linkID,
                    csa_tMonXRef         monID,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to report back private data events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfQueuedEvent

Format in C

```
csa_tEventStatus csa_cbfQueuedEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *queuedConnection,
                csa_tDeviceID           *queue,
                csa_tDeviceID           *callingDevice,
                csa_tDeviceID           *calledDevice,
                csa_tDeviceID           *lastRedirDevice,
                csa_tCallsInQueue       *numCallsInQueue,
                csa_tCallsInFront       *numCallsInFront,
                csa_tConnState          localConnectionState,
                csa_tEventCause         eventCause,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back call queued events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.

Name	Type	Description
queuedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the queued connection • The device identifier data structure for the queued connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
queue	Pointer	A pointer to the queue device identifier data structure. The device identifier data structure is described in the **queuedConnection** argument.
callingDevice	Pointer	A pointer to the calling device identifier data structure. The device identifier data structure is described in the **queuedConnection** argument.
calledDevice	Pointer	A pointer to the called device identifier data structure. The device identifier data structure is described in the **queuedConnection** argument.
lastRedirDevice	Pointer	A pointer to the last redirection destination device identifier data structure. The device identifier data structure is described in the **queuedConnection** argument.
numCallsInQueue	Pointer	A pointer to an unsigned integer containing the number of calls in the queue.
numCallsInFront	Pointer	A pointer to an unsigned integer containing the number of calls ahead of the call when it was enqueued.

Name	Type	Description
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfRetrievedEvent

Format in C

```
csa_tEventStatus csa_cbfRetrievedEvent (
                    csa_tLinkID          linkID,
                    csa_tMonXRef         monID,
                    csa_tConnectionID    *retrievedConnection,
                    csa_tDeviceID        *retrievingDevice,
                    csa_tConnState       localConnectionState,
                    csa_tEventCause      eventCause,
                    csa_tOctetStr        *applicationData,
                    csa_tPrivDataArray   *privData)
```

Description

Use this function to report back call retrieved events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.

Name	Type	Description
retrievedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the retrieved connection • The device identifier data structure for the retrieved connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
retrievingDevice	Pointer	A pointer to the retrieving device identifier data structure. The device identifier data structure is described in the **retrievedConnection** argument.
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfRouteEnd

Format in C

```
csa_tEventStatus csa_cbfRouteEnd (
                csa_tLinkID             linkID,
                csa_tRouteXRef          routeRef,
                csa_tStatus             errorNum,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back routing dialog end events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
routeRef	Unsigned integer	The routing cross-reference ID number.
errorNum	Integer	A value indicating the error condition. Table A.2 lists these values and the error conditions to which they relate.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfRouteRequest

Format in C

```
csa_tEventStatus csa_cbfRouteRequest (
                csa_tLinkID             linkID,
                csa_tRouteXRef          routeRef,
                csa_tDeviceID           *currentRoute,
                csa_tDeviceID           *callingDevice,
                csa_tDeviceID           *routingDevice,
                csa_tConnectionID       *routedCall,
                csa_tRouteSelAlg        routeAlgorithm,
                csa_tBoolean            *priority,
                csa_tOctetStr           *setupInfo,
                csa_tOctetStr           *applicationData,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back route request events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
routeRef	Unsigned integer	The routing cross-reference ID number.
currentRoute	Pointer	A pointer to the current route device identifier data structure containing: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
callingDevice	Pointer	A pointer to the calling device identifier data structure. The device identifier data structure is described in the **currentRoute** argument.
routingDevice	Pointer	A pointer to the routing device identifier data structure. The device identifier data structure is described in the **currentRoute** argument.
routedCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the routed connection • The device identifier data structure for the routed connection (The device identifier data structure is described in the **currentRoute** argument)
routeAlgorithm	Integer	A value indicating the routing algorithm: csa_kNormal = 0 csa_kLeastCost = 1 csa_kEmergency = 2 csa_kACD = 3 csa_kUserDefined = 4 csa_kNotSpecified = 5
priority	Unsigned integer	A boolean variable (0 = False) to indicate whether the call has priority or not.
setupInfo	Pointer	A pointer to a data structure containing details of the octet string to be used as set up information for the route. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure is described in the **setupInfo** argument.

Name	Type	Description
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfServiceInitiatedEvent

Format in C

```
csa_tEventStatus csa_cbfServiceInitiatedEvent (
                csa_tLinkID             linkID,
                csa_tMonXRef            monID,
                csa_tConnectionID       *initiatedConnection,
                csa_tConnState          localConnectionState,
                csa_tEventCause         eventCause,
                csa_tPrivDataArray      *privData)
```

Description

Use this function to report back service-initiated events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.
initiatedConnection	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the initiated connection • The device identifier data structure for the initiated connection The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string

Name	Type	Description
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

csa_cbfTransferredEvent

Format in C

```
csa_tEventStatus csa_cbfTransferredEvent (
                    csa_tLinkID             linkID,
                    csa_tMonXRef            monID,
                    csa_tConnectionID       *primaryOldCall,
                    csa_tConnectionID       *secondaryOldCall,
                    csa_tDeviceID           *transferringDevice,
                    csa_tDeviceID           *transferredDevice,
                    csa_tConnIDList         *connectionList,
                    csa_tConnState          localConnectionState,
                    csa_tEventCause         eventCause,
                    csa_tOctetStr           *applicationData,
                    csa_tPrivDataArray      *privData)
```

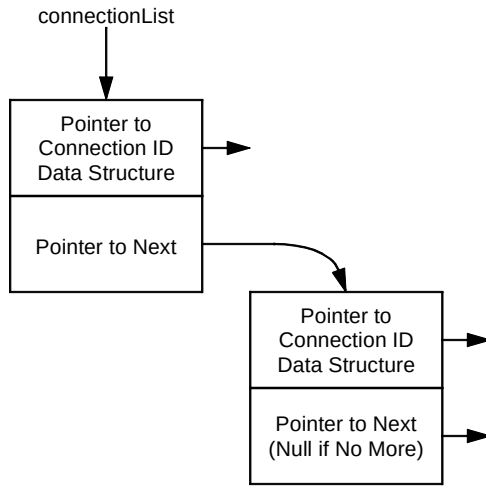
Description

Use this function to report back call transferred events to the CSTA Services Interface.

Arguments

Name	Type	Description
linkID	Unsigned integer	The ID number of the link to the CTI server.
monID	Unsigned integer	The monitor ID cross-reference number.

Name	Type	Description
primaryOldCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the primary old call • The device identifier data structure for the primary old call The device identifier data structure contains: • A value indicating whether the device ID character string is a dialable number or a non-dialable device number, as follows: csa_kDialingNumber = 0 csa_kDeviceNumber = 1 • A pointer to the device ID character string
secondaryOldCall	Pointer	A pointer to a connection identifier data structure containing: • The call reference identifier of the secondary old call • The device identifier data structure for the secondary old call (the device identifier data structure is described in the **primaryOldCall** argument)
transferringDevice	Pointer	A pointer to the transferring device identifier data structure. The device identifier data structure is described in the **primaryOldCall** argument.
transferredDevice	Pointer	A pointer to the transferred device identifier data structure. The device identifier data structure is described in the **primaryOldCall** argument.

Name	Type	Description
connectionList	Pointer	A pointer to a linked list of connection ID information. Each entry in the list contains: • A pointer to the data structure described for the **primaryOldCall** argument • A pointer to the next entry (null if there are no more entries) The linked list has this structure: 
localConnectionState	Integer	A value indicating the connection state: csa_kConnStateNull = 0 csa_kConnStateInitiated = 1 csa_kConnStateAlerting = 2 csa_kConnStateConnected = 3 csa_kConnStateHold = 4 csa_kConnStateQueued = 5 csa_kConnStateFailed = 6
eventCause	Integer	A value indicating the cause of the event. Table A.5 lists these values and the event causes to which they relate.

Name	Type	Description
applicationData	Pointer	A pointer to the data structure containing details of the octet string to be associated with the call. Note that this is not always null terminated. The data structure contains the following: • Length of the string (unsigned integer) • Pointer to the location of the string
privData	Pointer	A pointer to an array of three data structures containing private data returned from the switching system. The data structure has three fields: • Manufacturer ID (24 characters) • Data length (unsigned integer) • Data (128 unsigned characters)

Appendix A: Error and Event Codes

A.1: Introduction

This appendix lists and describes the error and event status codes that can be returned by the CSTA Services Interface functions.

Each set of functions has its own return codes, which are referenced in the function declarations as follows:

`csa_tMgmtStatus`	Return codes for Management functions
`csa_tStatus`	Return codes for Service Request functions
`csa_tResponseStatus`	Return codes for Service Response functions
`csa_tEventStatus`	Return codes for Service Event functions

The return codes are listed and described in the following sections:

csa_tMgmtStatus	Section A.2
csa_tStatus	Section A.3
csa_tResponseStatus	Section A.4
csa_tEventStatus	Section A.5

CSTA events are reported by csa_tEventCause return codes in the Service Event functions. Section A.6 lists and describes the CSTA event return codes.

A.2: Management Functions Return Codes

Table A.1 lists and describes the csa_tMgmtStatus return codes for the Management Operation and Management Status functions.

Table A.1: csa_tMgmtStatus Return Codes

Return Code	Value	Meaning
csa_kMgmtSuccess	1	Management function completed successfully.
csa_kMgmtInitialiseFailed	2	Switching system initialization failed.

Table A.1: csa_tMgmtStatus Return Codes (Continued)

Return Code	Value	Meaning
csa_kMgmtShutdownFailed	3	Switching system shutdown failed.
csa_kMgmtConfigurationError	4	Link configuration error.
csa_kMgmtOutOfMemory	5	Dynamic memory allocation failed.
csa_kMgmtInvalidTraceType	6	Invalid Trace options specified.
csa_kMgmtInitLinkUpFailed	7	Link initialization failed.
csa_kMgmtLinkDownFailed	8	Link shutdown failed.

A.3: Service Request Return Codes

Table A.2 lists and describes the csa_tStatus return codes for the Service Request functions.

Table A.2: csa_tStatus Return Codes

Return Status	Value	Description
No Errors		
csa_kSuccess	0	Request completed successfully.
Operations Errors		
csa_kOperations_generic	1	Unspecific operations error.
csa_kRequestIncompatibleWithObject	2	Request not compatible with the object.
csa_kValueOutOfRange	3	Parameter value outside the defined range.
csa_kObjectNotKnown	4	Parameter value not known to the service.
csa_kInvalidCallingDevice	5	Calling device not valid.
csa_kInvalidCalledDevice	6	Called device not valid.
csa_kInvalidForwardingDestination	7	Forwarding destination device not valid.
csa_kPrvlgVltnnOnSpcfdDvc	8	Device not authorized for requested service.
csa_kPrvlgVltnOnClldDvc	9	Called device not authorized for requested service.

Table A.2: csa_tStatus Return Codes (Continued)

Return Status	Value	Description
csa_kPrvlgVltnOnCllngdvc	10	Calling device not authorized for requested service.
csa_kInvalidCSTACallIdentifier	11	Call identifier not valid.
csa_kInvalidCSTADeviceIdentifier	12	Device identifier not valid.
csa_kInvalidCSTAConnectionIdentifier	13	Connection identifier not valid.
csa_kInvalidDestination	14	Forwarding destination device not valid.
csa_kInvalidFeature	15	Feature specified not valid.
csa_kInvalidAllocationState	16	Allocation condition not valid.
csa_kInvalidCrossRefID	17	Cross-reference identifier not valid.
csa_kInvalidObjectType	18	Object type not in valid range.
csa_kSecurityViolation	19	Request would violate a security requirement.
State Errors		
csa_kStateIncompatibility_generic	21	Unspecific state error.
csa_kIncorrectObjectState	22	Object in incorrect state for the service.
csa_kInvalidConnectionID	23	Connection identifier invalid.
csa_kNoActiveCall	24	No active call for requested service to operate on.
csa_kNoHeldCall	25	No held call for requested service to operate on.
csa_kNoCallToClear	26	No call to clear on connection identifier specified.
csa_kNoConnectionToClear	27	No connection to clear on connection identifier specified.
csa_kNoCallToAnswer	28	No active call to be answered on connection identifier specified.
csa_kNoCallToComplete	29	No call to complete on connection identifier specified.

Table A.2: csa_tStatus Return Codes (Continued)

Return Status	Value	Description
System Resource Errors		
csa_kSystmRsrcAvlblty_generic	31	Unspecific system resource error.
csa_kServiceBusy	32	Service temporarily unavailable.
csa_kResourceBusy	33	Resource temporarily unavailable.
csa_kResourceOutOfService	34	Resource out of service.
csa_kNetworkBusy	35	Server network sub-domain busy.
csa_kNetworkOutOfService	36	Server network sub-domain out of service.
csa_kOverallMonitorLimitExceeded	37	Service exceeds server's monitor limit.
csa_kConferenceMemberLimitExceeded	38	Service exceeds server's conference members limit.
Subscribed Resource Errors		
csa_kSubscribedResAvail_generic	41	Unspecific subscribed resources error.
csa_kObjectMonitorLimitExceeded	42	Service exceeds server's monitor limit for specified object.
csa_kExternalTrunkLimitExceeded	43	Service exceeds server's trunk limit for specified object.
csa_kOtstndngRqstsLmtExcdd	44	Service exceeds server's outstanding requests limit for specified object.
Performance Errors		
csa_kPerformanceManagement_generic	51	Unspecific performance error.
csa_kPerformanceLimitExceeded	52	Performance limit exceeded.

Table A.2: csa_tStatus Return Codes (Continued)

Return Status	Value	Description
Security Errors		
csa_kUnspecified	60	Unspecific security error.
csa_kSequenceNumberViolated	61	Operation sequence number error.
csa_kTimeStampViolated	62	Operation time stamp error.
csa_kPACViolated	63	Operation PAC error.
csa_kSealViolated	64	Operation Seal error.

A.4: Service Response Return Codes

Table A.3 lists and describes the csa_tResponseStatus return codes for the Service Response functions.

Table A.3: csa_tResponseStatus Return Codes

Return Code	Value	Meaning
csa_kResponseSuccess	1	Response generated successfully.
csa_kResponseError	2	Unspecific error with response generation.

A.5: Service Event Return Codes

Table A.4 lists and describes the csa_tEventStatus return codes for the Service Event functions.

Table A.4: csa_tEventStatus Return Codes

Return Code	Value	Meaning
csa_kEventSuccess	1	Event generated successfully.
csa_kEventError	2	Unspecific error with event generation.
csa_kEventParamError	3	The event parameter value is not in range.
csa_kEventNotSup	4	The event requested is not supported.

A.6: CSTA Event Codes

Table A.5 lists and describes the csa_tEventCause return codes that report CSTA events.

Table A.5: csa_tEventCause Return Codes

Event Code	Value	Description
csa_kCauseNone	0	Unspecific cause code.
csa_kCauseActiveMonitor	1	Active Monitor invoked.
csa_kCauseAlternate	2	Call being interchanged with another call.
csa_kCauseBusy	3	Call encountered busy or unavailable device.
csa_kCauseCallBack	4	Call Back invoked.
csa_kCauseCallCancelled	5	Call terminated before device went on-hook.
csa_kCauseCallForwardAlways	6	Call redirected by Call Forward set for all conditions.
csa_kCauseCallForwardBusy	7	Call redirected by Call Forward set for endpoint busy.
csa_kCauseCallForwardNoAnswer	8	Call redirected by Call Forward set for endpoint not answering.
csa_kCauseCallForward	9	Call redirected by Call Forward for any reason.
csa_kCauseCallNotAnswered	10	Call not answered because timer elapsed.
csa_kCauseCallPickup	11	Call redirected by Call Pickup.
csa_kCauseCampOn	12	Camp On invoked.
csa_kCauseDestNotObtainable	13	Call could not reach destination.
csa_kCauseDoNotDisturb	14	Call encountered a Do Not Disturb.
csa_kCauseIncompatibleDestination	15	Call encountered an incompatible destination.
csa_kCauseInvalidAccountCode	16	Call has invalid account code.
csa_kCauseKeyOperation	17	Event occurred at a bridged or twin device.
csa_kCauseLockout	18	Call encountered inter-digit time-out while dialling.
csa_kCauseMaintenance	19	Call encountered a facility or endpoint in maintenance condition.
csa_kCauseNetworkCongestion	20	Call encountered congested network.

Table A.5: csa_tEventCause Return Codes (Continued)

Event Code	Value	Description
csa_kCauseNetworkNotObtainable	21	Call could not reach destination network.
csa_kCauseNewCall	22	Call not yet redirected.
csa_kCauseNoAvailableAgents	23	Call could not access any agent.
csa_kCauseOverride	24	Call resulted from use of Override feature.
csa_kCausePark	25	Call parked or retrieved from park.
csa_kCauseOverflow	26	Call overflowed a queue, group, or target.
csa_kCauseRecall	27	Call time-out due to failed feature or lack of expected user action.
csa_kCauseRedirected	28	Call redirected.
csa_kCauseReorderTone	29	Call encountered a re-order condition.
csa_kCauseResourcesNotAvailable	30	Resources not available.
csa_kCauseSilentMonitor	31	Third party listens in on call silently.
csa_kCauseTransfer	32	Transfer in progress or has occurred.
csa_kCauseTrunksBusy	33	Call encountered Trunks Busy.
csa_kCauseVoiceUnitInitiator	34	Action by automated unit rather than by human user.
csa_kCauseBlocked	35	One party disconnects from call leaving other party with local connection.
csa_kCauseCharCountReached	36	Specified number of characters entered.
csa_kCauseConsult	37	Consultation in progress or has occurred.
csa_kCauseDistributed	38	Call distributed by ACD or hunt group.
csa_kCauseDTMFToneDetected	39	DTMF tone detected during message playback or recording.
csa_kCauseDurExceeded	40	Maximum time for playback or recording exceeded.
csa_kCauseEndOfDataDetected	41	End of data detected during message play or review.
csa_kCauseEnteringDistribution	42	Call delivered to distribution mechanism (ACD).
csa_kCauseForcedPause	43	Agent enters required pause period.
csa_kCauseMakeCall	44	Make Call service initiated.

Table A.5: csa_tEventCause Return Codes (Continued)

Event Code	Value	Description
csa_kCauseMsgSizeExceeded	45	Maximum size of message exceeded during recording.
csa_kCauseNetworkSignal	45	Network signal (trunk supervision/call progress) occurred.
csa_kCauseNextmsg	47	Next message in sequence started.
csa_kCauseNormalClearing	48	Call or connection cleared normally.
csa_kCauseNoSpeechDetected	49	No speech detected during recording.
csa_kCauseNumberChanged	50	Called number has changed to a new number.
csa_kCauseSingleStepConf	51	Single Step Conference service initiated.
csa_kCauseSingleStepTransfer	52	Transfer occurred in a single step.
csa_kCauseSpeechDetected	53	Speech detected during recording.
csa_kCauseSwitchTerminated	54	Switching system terminated collection before other termination condition encountered.
csa_kCauseTermCharReceived	55	Termination character received.
csa_kCauseTimeout	56	Timeout occurred.

Appendix B: CSTA Services and CT-Connect Functions

This appendix lists the mapping of CSTA services to CT-Connect API functions.

B.1: CSTA Services Mappings

Table B.1 lists the CSTA service requests supported by the CSTA Services Interface and maps them to CT-Connect API functions.

All of the CSTA service events are returned by the CT-Connect API function ctcGetEvent. Note that CT-Connect only allows monitoring of devices represented by dialable numbers. Non-dialable device numbers may appear in events, but CT-Connect represents them only as trunk devices.

For more details, contact Dialogic.

Table B.1: CSTA Services and CT-Connect Functions

CSTA Service	CT-Connect API Function
Alternate Call	ctcSwapWithHeld
Answer Call	ctcAnswerCall
Associate Data	ctcAssociateData
Call Completion	ctcRespondToInactiveCall
Clear Call	Not supported.
Clear Connection	ctcHangupCall ctcCancelCall
Conference Call	ctcConferenceCall
Consultation Call	ctcConsultationCall
Deflect Call	ctcDeflectCall
Escape	ctcCstaEscape
Hold Call	ctcHoldCall

Table B.1: CSTA Services and CT-Connect Functions (Continued)

CSTA Service	CT-Connect API Function
Make Call	ctcMakeCall
Make Predictive Call	ctcMakePredictiveCall
Monitor Start	ctcSetMonitor
Monitor Stop	ctcSetMonitor
Pickup Call	ctcPickupCall
Pickup Group Call	ctcPickupCall
Query Device Agent State	ctcGetAgentState
Query Device Do-Not-Disturb	ctcGetDoNotDisturb
Query Device Forwarding	ctcGetCallForward
Query Device Information	ctcGetChannelInformation
Query Device Message Waiting	ctcGetMessageWaiting
Reconnect Call	ctcReconnectCall
Retrieve Call	ctcRetrieveCall
Route End	ctcRespondToRouteQuery
Route Select	ctcRespondToRouteQuery
Send DTMF	ctcSendDTMF
Set Agent State	ctcSetAgentStatus
Set Do-Not-Disturb	ctcSetDoNotDisturb
Set Forwarding	ctcSetCallForward
Set Message Waiting	ctcSetMessageWaiting
Single Step Transfer	ctcSingleStepTransfer
Snapshot Device	ctcSnapshot
Transfer Call	ctcTransferCall

Index

A

Abstract Syntax Notation 1
    See ASN.1
Account code
    adding to call, 5-6
ACSE
    error reports, 2-6
    errors, 4-2
Active call
    clear, 5-44
    hold, 5-2
    transfer, 5-61
Agent logged on, 5-53
Agent state, 5-34
    list, 5-54
    setting, 5-53
Allocation state, 5-25
Answer call, 5-4
Answer ringing call, 5-30
Application
    and required DLL files, 2-10
    background information, 2-1
    building from sample files, 1-3
    events, 7-1
    function calling convention, 2-5
    guidelines, 2-5
    how to develop, 2-1
    identifier parameters, 2-2
    initialising CSTA Services Interface, 3-3
    requests, 5-1
    required functionality, 2-9
    response functions, 6-1
    shutdown CSTA Services Interface, 3-5
    verifying operation, 2-11
Application data
    data structure, 5-7
ASN.1, 1-1
    error reports, 2-6
    errors, 4-2
    translation, 1-1
Associate data, 5-6
Asynchonous operation, 2-6
    illustrated, 2-7
authorisation code
    adding to call, 5-6

C

Call
    allocation state types, 5-26
    completion, 5-8
    completion request types, 5-9
    conference, 5-14
    consultation, 5-16
    forwarding types, 5-58
    putting on hold, 5-16
    states listed, 6-43
    transfer, 5-61
Call backs
    data structure, 3-4
    location, 2-5
Call reference identifier, 2-3
Call state list
    data structure, 6-42
Calling convention
    for functions, 2-5
Causes
    of events, A-6
Clear call, 5-10, 5-44
Clear connection, 5-12
Communications
    parameters, 2-5
    parameters data structure, 3-3
    port, 2-6
Completion type, 5-8
Computer Supported Telecommunications Applications
    See CSTA

Computer Telephony Integration
See CTI
Conference call, 5-14
Connection ID, 2-2
data structure, 5-2
list data structure, 6-9
Connection states
listed, 6-43
Consultation call, 5-16
CSTA
and CT-Connect, 1-2, B-1
applicability to switching systems, 1-1
ASN.1 message flow, 1-2
environment, 1-1
errors, A-1
event causes, A-6
events, 7-1, A-1
integrating switching system, 1-1
messages, 1-3
protocol, 1-1
reporting events, 5-27
requests, 5-1
requirements, 1-3
responses, 6-1
return codes, 2-9
scenarios definition, 2-11
CSTA Services Interface
and CSTA_SERVICES.DLL, 2-5
and CT-Connect API, B-1
as a protocol converter, 1-2
ASN.1 support, 1-1
asynchronous operation, 2-6
asynchronous operation illustrated, 2-7
contents of kit, 1-1, 1-2, 2-10
environment defined, 2-2
environment illustrated, 1-4
error reports, 4-2
event generation, 2-2
event reporting, 4-1
example C files, 1-2
files described, 2-10
flow of messages through, 1-4
functionality provided, 1-2
functions provided, 2-1
initialisation and management, 2-5, 3-1
major components, 1-3
purpose, 1-1
reporting events to, 5-27
synchronous operation, 2-6
synchronous operation illustrated, 2-7
CSTA_SERVICES.DLL
and your application, 2-10, 2-11
function calling, 2-5
processing blocked, 2-6
CT-Connect, 1-2
API mapped to CSTA services, B-1
CTI, 1-1
server, 1-4
server connection, 2-6, 4-4
server links, 2-6, 2-7, 3-5
server maximum links, 2-4

D

Data structures
csa_tCallbacks, 3-4
csa_tCallStateList, 6-42
csa_tCommsSettings, 3-3
csa_tConnectionID, 5-2
csa_tConnIDList, 6-9
csa_tDeviceID, 5-17
csa_tForwardParam, 6-27
csa_tOctetStr, 5-7
csa_tPrivDataArray, 5-3
Deflect call, 5-18
Destination busy, 5-8
Device
class and type, 5-40
classes listed, 6-30
number types, 5-2
obtaining information on, 5-40
status information, 5-63
types listed, 6-30
Device ID, 2-2
data structure, 5-17
Dialable numbers
See DN
DLL
files listed, 2-10
files required by application, 2-10

DN
dialable number, 2-4
specifying, 5-2
Do-not-disturb
setting, 5-55
status, 5-36
DTMF tones
generating, 5-51
Dynamic Link Library
See DLL

E

Errors
ACSE, 2-6, 4-2
ASN.1, 2-6, 4-2
internal, 4-2
listed and defined, A-1
reported, 2-6, 4-2
Events, 7-1
and Monitor ID, 2-4
causes listed, A-6
generation, 2-2
listed and defined, A-1
reporting, 5-27
services, 2-2

F

Forwarding
destination, 5-57
status, 5-38
type, 5-57
Forwarding parameters list
data structure, 6-27
Function prototypes
management operations, 2-2
service events, 2-2
service responses, 2-2
Functions
and multithreaded code, 2-7
blocking, 2-6
call back locations, 2-5
events, 7-1
failed, 2-9
management operations, 2-5, 3-1
management status, 4-1
requests, 5-1
responses, 6-1
return status codes, 2-6, 2-9, A-1
successful, 2-9

G

Generate DTMF tones, 5-51
Group pickup call, 5-32

H

Hold call, 5-2, 5-16, 5-21
retrieve, 5-2, 5-44, 5-46
transfer, 5-65

I

Identifier parameters
defined, 2-2
Initialisation
and management, 2-5, 3-1
process, 2-5
Internal and parse errors, 4-2
Invoke ID, 2-2

L

Links
and CTI server, 2-6
and more than one CTI server, 2-7
and threads, 2-7
communication parameters, 2-5
disconnecting CTI server, 3-5
maximum number, 2-4
state changes, 2-4
to switching system, 3-2

M

Make call, 5-23, 5-25
Management
and initialisation, 2-5

operations described, 2-1
status reports described, 2-1
Manufacturer ID
and private data, 2-3
Message waiting, 5-59
status, 5-42
Messages
typical flow, 1-4
Monitor ID, 2-2
Monitoring
CT-Connect and device type, B-1
start, 5-27
stop, 5-29

N

Non-dialable numbers, 2-4
specifying, 5-2

O

Operation
asynchronous, 2-6
synchronous, 2-6
Operations
management, 2-1

P

Pickup call, 5-30
group, 5-32
Predictive call, 5-25
Private data, 5-20
data structure, 2-3, 5-3
required content, 2-3
Programming
See application

Q

Query
device agent state, 5-34
device information, 5-40
do-not-disturb status, 5-36
forwarding status, 5-38
message waiting, 5-42

R

Reconnect call, 5-44
Redirect call, 5-18
Release device, 5-12
Reporting events, 5-27, 5-29
Requests
services, 2-1
Response time
and asynchronous operation, 2-7, 2-8
Responses, 6-1
services, 2-1
Retrieve call, 5-2, 5-44, 5-46
Return status codes, 2-9, A-1
CSTA events, A-6
Route end, 5-46, 5-48
Route select, 5-49
Routing, 5-46, 5-48
algorithms, 7-46
destination, 5-49

S

Scenarios
and operational verification, 2-11
Send DTMF tones, 5-51
Services
events described, 2-2
requests, 5-1
requests described, 2-1
responses described, 2-1
Set
agent state, 5-53
do-not-disturb, 5-55
forwarding, 5-57
message waiting, 5-59
Single step transfer, 5-61
Snapshot, 5-63
Start monitoring, 5-27
States
agent, 5-54
call, 6-43
connection, 6-43

Status code returns, 2-6, A-1
Status information
device, 5-63
Status reports
management, 2-1, 4-1
stdcall
function calling convention, 2-5
Stop monitoring, 5-29
Stub functions
defined, 1-2
described, 2-1
illustrated, 2-8
management status reports, 2-2
services requests, 2-2
Switching system
defined, 1-1
integrating into CSTA environment, 1-1
links, 3-2
private data, 5-20
types of, 1-1
Synchronous operation, 2-6
illustrated, 2-7

T

Transfer call, 5-61, 5-65
Types
allocation states, 5-26
completion request, 5-9
device classes, 6-30
device numbers, 5-2
devices, 6-30
forwarding, 5-58
internal and parse errors, 4-2
routing algorithms, 7-46

V

Verifying application operation, 2-11