Cisco Unified Contact Center Express CTI Protocol Developer Guide Release 11.0(1)

First Published: August 27, 2015

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Preface — **Preface** xi

Purpose xi

Audience xi

Organization xi

Related Documentation xii

Obtaining Documentation, Obtaining Support, and Submitting a Service Request xiii

Documentation Feedback xiii

PART I — **How to Use the Messages** 1

CHAPTER 1 — **Overview** 3

The Unified CCX CTI Protocol 3

Unified CCX 3

Unified CCX CTI Client/Server Architecture 3

Uses of the Unified CCX CTI Protocol 4

CHAPTER 2 — **Defining Unified CCX CTI Messages** 7

Message Structure and Syntax 8

Alignment of Data Elements 12

Pack and Unpack a Unified CCX CTI Message 12

Pack a CTI message 12

Unpack a CTI Message 13

Message Classification 13

Message Type Definitions 14

CHAPTER 3 — **Managing Sessions** 15

How a Client/Server Connection is Managed 15

How a Session is Initialized 16
Reasons for a Connection to Fail 16
How a Session is Maintained 17
How a Session is Ended 18
Two Client Modes for Connecting with Unified CCX 19
How the Client Selects the Types of Messages Wanted 20
Reasons for the Unified CCX to Send SYSTEM_EVENT Messages 21
The Unified CCX CTI Server in a High Availability Unified CCX System 21
Connect CTI Sessions in a Clustered Unified CCX 21

CHAPTER 4 Managing Configuration Data 23
Why Configuration Data Is Shared 23
Configuration Data That Can Be Shared 23
Configuration Message Summary Descriptions 24
Message List 24
Message Summary Descriptions 24
Share Configuration Data 26
How Configuration Consistency is Maintained 28
Share Agent Team Configuration Data 29
Error Handling 29

CHAPTER 5 Managing Agent States 31
What is an Agent State? 31
The Causes of Agent-State Transitions 33
Agent-State Message Summary Descriptions 34
Agent Authentication Without Agent State Change 35
About Forced Agent Logout and Forced Agent Login 35

CHAPTER 6 Communicating Call Events 37
Call Events 37
Call Context Data 38
Call-Event Message Summary Descriptions 38
The Unspecified Flow of Call-Event Messages 41
Fields Common to All Call-Event Messages 43
CallType Fields 43

CHAPTER 7 Managing Client Control of Calls 45

What is a Call-Control Message? 45

The Call-Control Message Flow 46

Call-Control Message Summary Descriptions 46

CHAPTER 8 Using the Outbound Feature 49

About the Outbound Feature 49

Outbound Expanded Call Context (ECC) Variables 49

Outbound Call Events 50

The BAResponse ECC Variable 51

PART II Reference Section 55

CHAPTER 9 All Message Types Organized by Class and Type 57

All Message Types Organized by ID Number 68

Session-Management Messages 73

Session Initialization Maintenance System Event and Termination Messages 73

OPEN_REQ 73

OPEN_CONF 75

HEARTBEAT_REQ 77

HEARTBEAT_CONF 77

SYSTEM_EVENT 77

CLOSE_REQ 79

CLOSE_CONF 79

Masks Used in the OPEN_REQ Message 79

CTI Service Masks 80

Unsolicited Call-Event Message Masks 80

Agent State Masks 82

Configuration-Information Masks 83

Failure Messages 83

FAILURE_CONF 83

FAILURE_EVENT 84

Configuration Messages 84

CONFIG_REQUEST_KEY_EVENT 85

CONFIG_KEY_EVENT 85
CONFIG_REQUEST_EVENT 86
CONFIG_BEGIN_EVENT 87
CONFIG_END_EVENT 88
CONFIG_CSQ_EVENT 89
CONFIG_APPLICATION_EVENT 90
CONFIG_AGENT_EVENT 92
CONFIG_DEVICE_EVENT 93
TEAM_CONFIG_EVENT 95
TEAM_CONFIG_REQ 96
TEAM_CONFIG_CONF 97
Agent-State Messages 97
AGENT_STATE_EVENT 98
QUERY_AGENT_STATE_REQ 100
QUERY_AGENT_STATE _CONF 101
SET_AGENT_STATE_REQ 102
SET_AGENT_STATE_CONF 103
Call-Event Messages 104
Primary.Actual Field Format 104
General Rules 105
BEGIN_CALL_EVENT 105
END_CALL_EVENT 107
CALL_DATA_UPDATE_EVENT 108
CALL_DELIVERED_EVENT 111
CALL_ESTABLISHED_EVENT 114
CALL_HELD_EVENT 116
CALL_RETRIEVED_EVENT 117
CALL_CLEARED_EVENT 118
CALL_CONNECTION_CLEARED_EVENT 119
CALL_ORIGINATED_EVENT 120
CALL_FAILED_EVENT 121
CALL_CONFERENCED_EVENT 123
CALL_TRANSFERRED_EVENT 125
CALL_DIVERTED_EVENT 128
CALL_SERVICE_INITIATED_EVENT 129

CALL_QUEUED_EVENT 131
CALL_DEQUEUED_EVENT 133
RTP_STARTED_EVENT (OPTIONAL) 134
RTP_STOPPED_EVENT (OPTIONAL) 136
Call-Control (Client-Control) Messages 137
CONTROL_FAILURE_CONF 139
ALTERNATE_CALL_REQ 139
ALTERNATE_CALL_CONF 141
ANSWER_CALL_REQ 141
ANSWER_CALL_CONF 142
CLEAR_CALL_REQ 142
CLEAR_CALL_CONF 143
CLEAR_CONNECTION_REQ 143
CLEAR_CONNECTION_CONF 144
CONFERENCE_CALL_REQ 144
CONFERENCE_CALL_CONF 146
CONSULT_CALL_REQ 148
CONSULT_CALL_CONF 150
HOLD_CALL_REQ 150
HOLD_CALL_CONF 151
MAKE_CALL_REQ 151
MAKE_CALL_CONF 153
RECONNECT_CALL_REQ 154
RECONNECT_CALL_CONF 155
RETRIEVE_CALL_REQ 155
RETRIEVE_CALL_CONF 156
TRANSFER_CALL_REQ 156
TRANSFER_CALL_CONF 158
SEND_DTMF_SIGNAL_REQ 159
SEND_DTMF_SIGNAL_CONF 160
SET_CALL_DATA_REQ 160
SET_CALL_DATA_CONF 162
SUPERVISE_CALL_REQ 162
SUPERVISE_CALL_CONF 164
SUPERVISOR_ASSIST_REQ 165

SUPERVISOR_ASSIST_CONF 166
SUPERVISOR_ASSIST_EVENT 167
BAD_CALL_REQ 167
BAD_CALL_CONF 168
Miscellaneous Messages 168
QUERY_QUEUE_STATISTICS_REQ 169
QUERY_QUEUE_STATISTICS_CONF 169
QUERY_AGENT_QUEUE_STATISTICS_REQ 172
QUERY_AGENT_QUEUE_STATISTICS_CONF 172
QUERY_DEVICE_INFO_REQ 173
QUERY_DEVICE_INFO_CONF 174
QUERY_SUMMARY_STATISTICS_REQ 175
QUERY_SUMMARY_STATISTICS_CONF 175
SNAPSHOT_CALL_REQ 177
SNAPSHOT_CALL_CONF 178
SNAPSHOT_DEVICE_REQ 181
SNAPSHOT_DEVICE_CONF 181

CHAPTER 10 **Data Types and Message Constants** 183
Message Field Data Types 184
Message Header Data Format 185
NAMEDVARIABLE Data Format 185
NAMEDARRAY Data Format 186
Floating FieldDataID Values 187
DeviceType Values 192
LocalConnectionState (LCS) Values 193
Call EventCause (CEC) Values 193
SystemEventID Values 195
ConnectionDeviceType Values 196
Audio Codec Type Values 196
CallType Values 197
Control Failure (CF) Values 198
Unified CCX Error Code Values 204
Special Values 208
Unified CCX Status Values 208

Disposition Values 209
Error (E) Status Codes 209
Reason Codes for Agent State Change 211

PART III System Level Information 213

CHAPTER 11 Client Application Development Guidelines 215
Configuring a Client Program on Unified CCX 215
Debugging a Unified CCX Client Program 215
Example Trace Log-File Excerpts 216
The System Impact of Client Programs 217
Handling Reserved Messages and Reserved Fields in Client Programs 217
Call-Event Message Flows as Seen in the Log Files 217
Example Message Flow for a Queued and Answered Call 218
Example Message Flow for a Transferred Call 225
Example Message Flow for a Conferenced Call 232
Example Message Flows for an Outbound Option Call 236
The Outbound Subsystem Picks an Agent to Place a Call to a Customer Contact 237
The Agent Skips the Contact 241
The Agent Rejects the Reservation Call 243

CHAPTER 12 Document History 245

Preface

- Purpose, page xi
- Audience, page xi
- Organization, page xi
- Related Documentation, page xii
- Obtaining Documentation, Obtaining Support, and Submitting a Service Request, page xiii
- Documentation Feedback, page xiii

Purpose

This guide shows how to use the CTI Protocol to access Cisco Unified CCX (Unified CCX) as a client.

The goal of this guide is twofold:

- First, to provide descriptions of how to use the Unified CCX CTI protocol messages.
- Second, to provide the Unified CCX CTI protocol message definitions.

Audience

This guide is for third-party client developers writing applications that need to use the Cisco Unified CCX CTI Protocol. The guide is intended to contain all of the information needed by Cisco Unified CCX CTI client developers.

Organization

Chapter	Describes
Part I: How To Use the Messages	
Overview, on page 3	The general environment in which a CTI client operates and a brief description of the Unified CCX CTI Protocol with its uses.

Chapter	Describes
Defining Unified CCX CTI Messages, on page 7	How to define a Unified CCX CTI message using the required message format, syntax, and structure.
Managing Sessions, on page 15	How a client/server CTI connection is initialized, maintained, and managed, and what can cause failures.
Managing Configuration Data, on page 23	The configuration messages and how the sever shares configuration data with a client and how the client can maintain that data.
Managing Agent States, on page 31	Agent states, agent state transitions, agent state messages, agent authentication, and how agent state information is shared and updated between a client and the server.
Communicating Call Events, on page 37	What call events are and how information about them is shared between the client and the server.
Managing Client Control of Calls, on page 45	How client applications through client-control ("call-control") messages can request changes or establish, answer, control, or terminate calls on behalf of a specified agent phone number, and manipulate the telephone features associated with the agent's IP phone.
Part II: Reference Section	
Message Type Definitions	Reference definitions of each Unified CCX CTI Protocol message, and the masks, values, and constants used in the messages.
Data Types and Message Constants, on page 183	Miscellaneous message constant values
Part III: System Level Information	
Client Application Development Guidelines, on page 215	Guidelines for developing a client application.
Document History, on page 245	Changes/ updates made to each chapter in the document

Related Documentation

Refer to the following documents for further information about Unified CCX applications and products:

- Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts
- Cisco Unified Contact Center Express Scripting and Development Series: Volume 2, Editor Step Reference

- Cisco Unified Contact Center Express Scripting and Development Series: Volume 3, Expression Language Reference
- Cisco Unified Contact Center Express Administration Guide
- Cisco Unified Contact Center Express Installation Guide
- Cisco Unified Contact Center Express Servicing and Troubleshooting Guide
- Cisco Unified Communications Manager Administration Guide
- Cisco Unified Communications Manager Extended Services Administrator Guide
- Cisco Unified Communications Manager System Guide
- Cisco IP Telephony Network Design Guide
- IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Standard 754-1985 (IEEE, New York)
- Cisco Unified Contact Center Express Gateway Deployment Guide ICM/IPCC Enterprise Edition Release 8.0(0), IPCC Express Release 4.0(0)
- Getting Started with Cisco Unified Contact Center Express, Release 4.1
- Getting Started with Cisco Unified IP IVR, Release 4.1

Obtaining Documentation, Obtaining Support, and Submitting a Service Request

For information on obtaining documentation, obtaining support, security guidelines, and also recommended aliases and general Cisco documents, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Documentation Feedback

You can provide comments about this document by sending an e-mail to the following address:

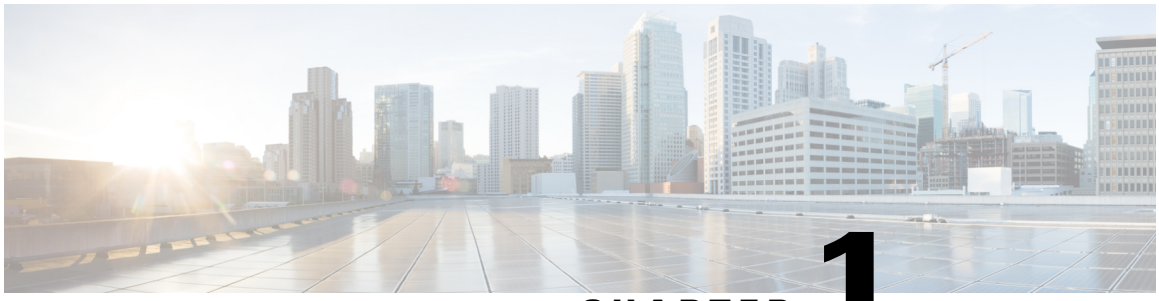
ccbu_docfeedback@cisco.com

We appreciate your comments.

PART I

How to Use the Messages

- Overview, page 3
- Defining Unified CCX CTI Messages, page 7
- Managing Sessions, page 15
- Managing Configuration Data, page 23
- Managing Agent States, page 31
- Communicating Call Events, page 37
- Managing Client Control of Calls, page 45
- Using the Outbound Feature, page 49



CHAPTER 1

Overview

This chapter includes the following topics.

- The Unified CCX CTI Protocol, page 3
- Unified CCX, page 3
- Unified CCX CTI Client/Server Architecture, page 3
- Uses of the Unified CCX CTI Protocol, page 4

The Unified CCX CTI Protocol

The Unified CCX Computer Telephony Integration (CTI) Protocol is a set of rules and message definitions for enabling a client to receive telephony information and control telephony functions through Unified CCX such as making and receiving voice calls and caller identification. The CTI protocol is supported for standard, enhanced, and premium license packages of Unified CCX.

Unified CCX

Unified CCX is an IP-based Automated Call Distribution (ACD) system that queues and distributes incoming calls destined for groups of Cisco Unified Communications Manager or Cisco Unified Communications Manager Express users.

You can use Unified CCX applications to route calls to specific agents. You can also integrate Unified CCX with Unified IP IVR to gather caller data and classify incoming calls. Unified CCX includes a web-based real-time and historical reporting system that you can use to monitor system, Contact Service Queue (CSQ), and resource performance. For more information about Unified CCX, see the Cisco Unified Contact Center Express Administration Guide and other documentation, found on the web at:

http://www.cisco.com/en/US/products/sw/custcosw/ps1846/tsd_products_support_series_home.html.

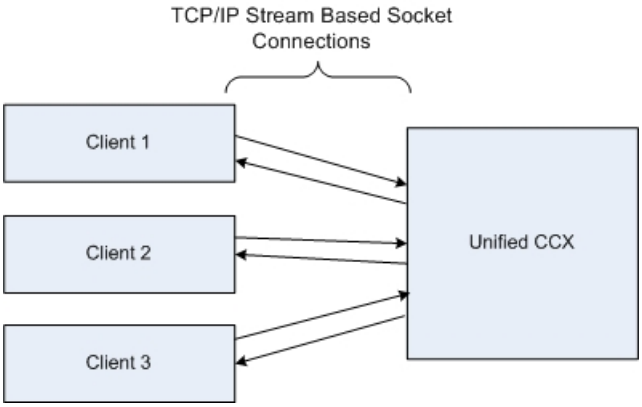
Unified CCX CTI Client/Server Architecture

Figure 1: General Architecture of a Cisco Unified CCX CTI Client-Server System, on page 4 shows the general architecture of a Unified CCX CTI client/server system. It shows one or more CTI clients exchanging

information with a Unified CCX system. Each client is its own separate program. CTI protocol messages are transported on top of TCP/IP protocol. They are stored in the data section of TCP/IP packets.

The Unified CCX CTI system listens on a configurable TCP/IP port for socket connections from CTI clients. After a TCP/IP connection is established between the Unified CCX and a CTI client, hand-shake messages are exchanged between the Unified CCX and the CTI client. If the handshake is successful, a session is opened. When the session is successfully opened between the CTI client and Unified CCX, the client may begin using the full set of Unified CCX CTI features. This is the subject of the rest of this guide.

Figure 1: General Architecture of a Cisco Unified CCX CTI Client-Server System



All Unified CCX CTI messages are received and sent through the CTI server module of the RMCM (Resource Manager- Contact Manager) Subsystem, used for resource distribution and the queuing of calls to call centers. See the *Cisco Unified Contact Center Express Administration Guide* for more information on this.

Uses of the Unified CCX CTI Protocol

The Unified CCX CTI protocol is a message based protocol that allows Unified CCX clients to send and receive information about:

- Current system configuration and future updates.
- Agents and their states
- Calls and their states
- Statistics for agents, calls, and queues
- Third-party call control

Note Not all Unified CCCX configurations support third party call control. For example, Cisco Unified Communications Manager Express does not support this feature.

- Device snapshots

Accordingly, you can create different applications that use the preceding protocol functions. For example:

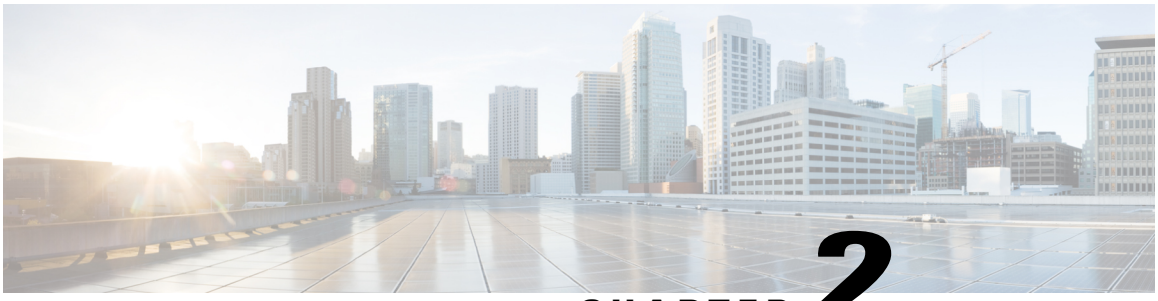
- A reporting application

If an application developer wants to write a reporting application, the developer may write an Unified CCX CTI client running in the bridge mode to monitor all events related to all agents and agent devices. The application may monitor these agent activities and device states and add business logic based on these activities and states. For example, if a device is down, the application may send a pager or e-mail to notify IT staff for further investigation. The application may also send a notification to a supervisor if an agent state change pattern is out of ordinary.

- An integration application

  Agents may use several applications on their desktop. To integrate these applications with Unified CCX, a developer may write a Unified CCX CTI client to get agent information. The integration application may manipulate agent states based on the agent information and information from other applications.

- An agent or supervisor desktop



CHAPTER 2

Defining Unified CCX CTI Messages

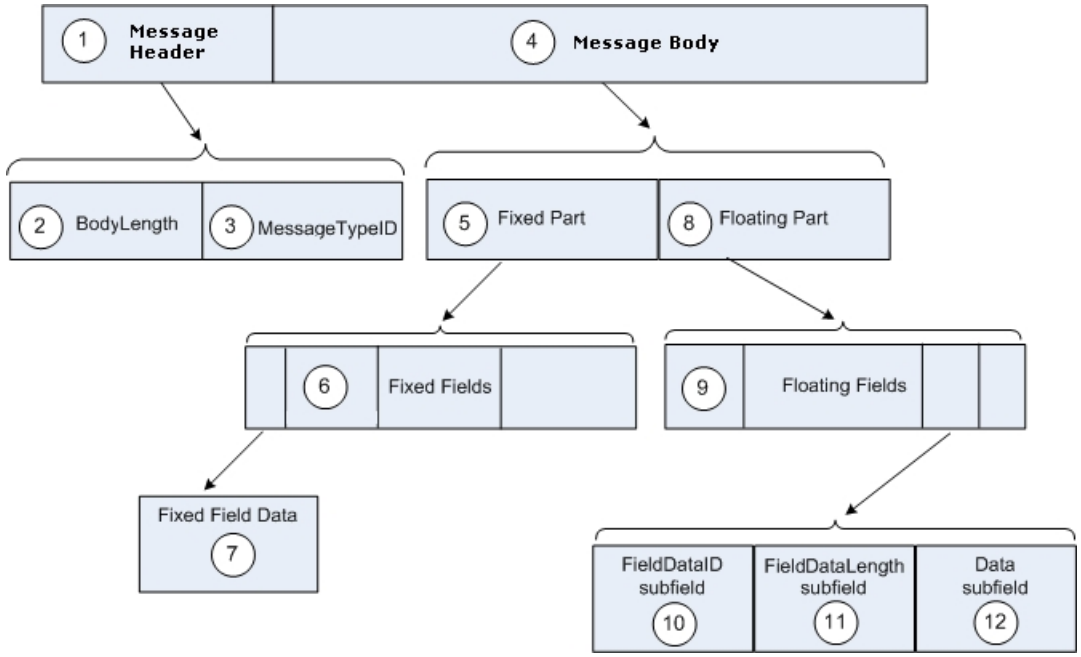
This chapter includes the following topics that you need to understand in order to correctly define Unified CCX CTI Messages:

- Message Structure and Syntax, page 8
- Alignment of Data Elements, page 12
- Pack and Unpack a Unified CCX CTI Message, page 12
- Message Classification, page 13
- Message Type Definitions, page 14

Message Structure and Syntax

A Unified CCX CTI message is a byte array of fields containing a Message Header and a Message Body. Figure 2: Diagram of a Unified CCX CTI Protocol Message, on page 8 shows its structure.

Figure 2: Diagram of a Unified CCX CTI Protocol Message



A Unified CCX CTI message is structured as follows. The numbered items refer to the numbers in Figure 2: Diagram of a Unified CCX CTI Protocol Message, on page 8:

1. Message Header

The message header is a required field. It contains a 4-byte MessageTypeID field and a 4-byte BodyLength field. The message header is in what is called the Message HeaDeR (MHDR) data format, which is a common format used for message headers that precede all messages exchanged between a client and a server. Message Header Data Format, on page 185 defines the MHDR format.

2. BodyLength

The BodyLength field contains the length of the MessageBody in bytes.

3. MessageTypeID

The MessageTypeID field identifies the type of message and has a unique numeric value used to determine the format of the remainder of the message. It also indicates if the message includes a floating part, and if so, what types of floating fields may appear within it.

The MessageTypeID field name usually indicates the purpose of the message. Example message types are OPEN_REQ and HEARTBEAT_REQ.

Table 10: Message Types Listed by Message Class and by Message TypeName, on page 58 defines all the message types in the message set with a unique MessageTypeID number that identifies each message type. To enter a message type into a message header, enter the MessageTypeID number specified for that message in Table 10: Message Types Listed by Message Class and by Message TypeName, on page 58.

4. Message Body

A message body consists of either a fixed part only or a fixed part and a floating part. The maximum size of a message body is 256 bytes.

5. Fixed Part

All the fields in the fixed part of a message are required and have both a fixed order and a fixed size.

6. Fixed Fields

Fixed fields begin after the message header. Every field defined in the fixed part of a message is required and must be defined in the order given with the defined size for the field. That is, all the field sizes in the fixed part of a message must be the size specified by the message definition.

For example, if there are 3 fields of 4 bytes in the fixed part of the message and the second field is a reserved one, you need to specify that the third field starts at byte 9, even though you are not using bytes 5 through 8.

A field in the fixed part of a message is identified by its location in the fixed part of the message.

Some messages have only fixed fields.

7. Fixed Field Data

Fixed Fields in messages are defined using a special set of data types. These are:

- CHAR
- UCHAR
- SHORT
- USHORT
- INT
- UINT
- BOOL
- TIME

For definitions of these data types, see Message Field Data Types, on page 184.

Note All numeric data longer than one byte is transmitted in order of the most significant byte to the least significant byte. This is the canonical network byte order defined by TCP/IP standards.

8. Floating Part (Variable in length and sometimes optional)

The message floating part contains floating fields that follow the fixed fields. The floating part is a required part of a message having floating fields.

9. Floating Fields

- The length of a floating field is variable. It can be up to or less than that maximum size specified in the message definition.
- Since these fields can be variable in length, the position of a floating field in a message can be said to float, depending on the length of the other floating fields in the message.
- Floating fields are packed together in the floating part of the message. The FieldDataID of one floating field immediately follows the data of the previous field. The message length (in the message header) indicates the end of the message.
- Some floating fields may be optional.
- In general, with one exception, floating fields may appear in any order in the floating part of a message, although, it is better to follow the order defined in the message. The exception is the situation in which a floating field can appear more than once, that is, when the field is a member of a list. In this case, a fixed field in the message indicates the number of list entries present.

10. FieldDataID

A floating field begins with a one-byte FieldDataID identifying the field type. Its data type is UCHAR.

Table 188: Floating FieldDataID Values Organized by the FieldDataID Name, on page 187 numerically lists all the available FieldDataID types.

Some floating fields are reserved.

Note In the Message Type Definitions, the square bracketed subscript number ending the Field Names for the data in the Floating Part of the message descriptions is the FieldDataID for that field. For example AgentID[194] means that 194 is the FieldDataID for the AgentID field.

Note A field FieldDataID is a global ID not specific to a message. More than one message can use the same DataID.

For example, the AGENT_STATE_EVENT and the SET_AGENT_STATE_REQ messages can both use the AGENT_ID FieldDataID.

11. FieldDataLength

Following the FieldDataID is a one-byte FieldDataLength field indicating the number of bytes, n, of data in the data subfield (excluding FieldDataID and FieldDataLength). Its data type is UCHAR.

12. Data

The data immediately follows the FieldLength subfield.

The maximum data size listed in each message definition for each floating field is the maximum number of data bytes allowed. This size, however, does not include the FieldDataID and the FieldLength bytes. For STRING data, the maximum size includes the null termination byte.

Any data type listed in Message Field Data Types, on page 184, (with the exception of the MHDR (Message Header) data type) can appear in the floating part of a message. However, the following four types of data appear only in the floating part:

- STRING[n]
- UNSPEC[n]
- NAMEDVAR
- NAMEDARRAY

For the definitions of all the message data types, see Message Field Data Types, on page 184.

Alignment of Data Elements

Note The messages described in this document are sent as a stream of bytes. If the client application uses data structures to represent the messages, take care that the data structures do not have padding inserted to align elements on particular boundaries, such as aligning 32-bit integers so that they are located on a 4-byte boundary.

Pack and Unpack a Unified CCX CTI Message

You may use data structures in your choice of programming languages to represent CTI messages. For example, in C++, you may use a class to represent a message type. The process of converting this data structure to a byte array to be sent to the Unified CCX server is called packing. The reverse process is called unpacking. The following are typical steps to do packing and unpacking.

Pack a CTI message

Procedure

Step 1 Allocate a continuous block of memory.

Step 2 Add the message header to the beginning of the memory block.
Reserve a 4-byte place for the BodyLength field to fill in later when you accurately know the message body length (when you have added all the Body fields). Convert the 4-byte MessageTypeID to network-byte order and add it to the memory block after the reserved BodyLength field.

Step 3 Convert each of the fixed message fields, one by one, to network-byte order, if needed, and add them to the memory block in the appropriate order according to the MessageTypeID.

Step 4 Convert the floating data fields, one by one, if needed, to the network-byte order. Add the fields in the floating part of the message into the memory block without any gap between the fields.

Step 5 Repeat this process to pack the rest of the floating fields until the end of the message.

Step 6 Remember to convert the BodyLength bytes from host-byte order to network-byte order. Then update the BodyLength field in the message header part of the memory block after all the floating fields are added to accurately reflect the message body length.

Unpack a CTI Message

Procedure

Step 1 Read the first 4 bytes as an unsigned integer. This value is the message BodyLength. Remember to convert the BodyLength bytes from network-byte order to host-byte order.

Step 2 Make sure all the BodyLength bytes are available from the network before further unpacking.

Step 3 Read the next 4 bytes and convert them to a host-byte-order unsigned integer. This value is the MessageTypeID.

Step 4 Based on the MessageTypeID, you can then use the message body definition to find out the fixed fields and their appropriate order in the fixed part of the message.

Step 5 Read each field, one by one, and convert it to host data-byte order, if needed.

Step 6 Based on the FieldDataID value and its definition, convert the floating data fields to the host-byte order, if necessary.

Step 7 Repeat this process to unpack the rest of floating fields until the end of the message. The end of the message is determined by the Message BodyLength.

Message Classification

The Unified CCX CTI Protocol messages can be classified in different ways:

- As Solicited (initiated by the client) or Unsolicited (initiated by Unified CCX).
  - Solicited Messages: Messages that the client initiates and for which the client expects a response (called a confirmation message) from Unified CCX.
  - Unsolicited Messages: Messages sent by Unified CCX that the client does not initiate and that do not require a confirmation.

  A bridge mode client can receive all unsolicited messages. In addition to unsolicited messages, a client receives confirmation (solicited) messages only for the requests it makes. A bridge mode client does not receive confirmation messages for other clients' requests.

  A client's request message triggers a solicited message response. In addition a client's request message may also trigger an unsolicited message. For example, the client's request could trigger an agent state event (an unsolicited message) as well as a confirmation message for the request (a solicited message).

- By function

  You can classify messages by function as:

  - Session Management Messages

    Messages for managing a session (for initializing, maintaining, and closing a session)

  - Configuration Messages

    Messages for sharing server configuration data (initial and updated) with the server's clients.

  - Agent State Messages

    Messages that allow the client to control the agent state, such as, for example, login and logout.

- Call Control Messages

  Messages that allow the client to control inbound and outbound calls.

- Call Events Messages

  Messages that describe what is happening to a call; for example, call established and call begun. Also messages that update call data in call context variables while the call is active.

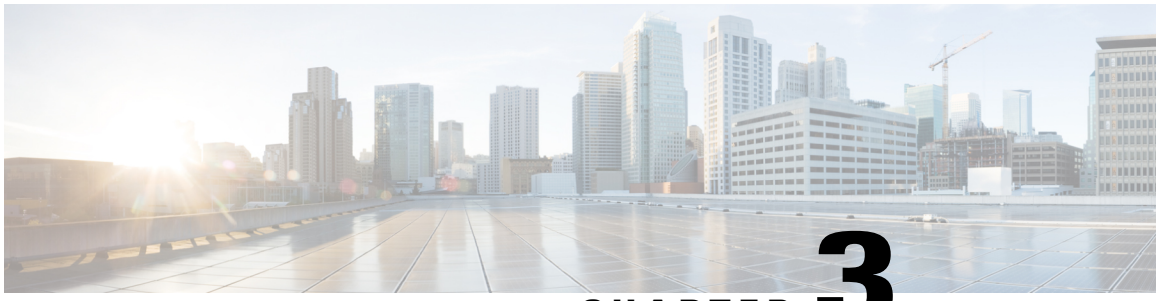
- Miscellaneous Service and Query Status Messages

  Miscellaneous unsolicited event messages and solicited messages available to all clients. And, messages that query the status of an agent, a queue, or a device.

Message Type Definitions

All the message types in the Unified CCX Protocol message set are listed and defined in Message Type Definitions When creating a message, you must use the definition specified for the type of message you are creating.

Note A version number next to a field name in a message type definition indicates that the field is used in the CTI protocol beginning with the specified version number.



CHAPTER 3

Managing Sessions

This chapter includes the following topics:

- How a Client/Server Connection is Managed, page 15
- How a Session is Initialized, page 16
- Reasons for a Connection to Fail, page 16
- How a Session is Maintained, page 17
- How a Session is Ended, page 18
- Two Client Modes for Connecting with Unified CCX, page 19
- How the Client Selects the Types of Messages Wanted, page 20
- Reasons for the Unified CCX to Send SYSTEM_EVENT Messages, page 21
- The Unified CCX CTI Server in a High Availability Unified CCX System, page 21
- Connect CTI Sessions in a Clustered Unified CCX, page 21

How a Client/Server Connection is Managed

The client uses TCP/IP protocols for network connectivity to the Unified CCX server.

TCP/IP transport services are used in a client/server arrangement. The clients are connected to specific addresses and ports on the server with a hostname/port number pair that is the IP address and the TCP port number of the server. See The Unified CCX CTI Server in a High Availability Unified CCX System, on page 21 for how to select an IP address and port number.

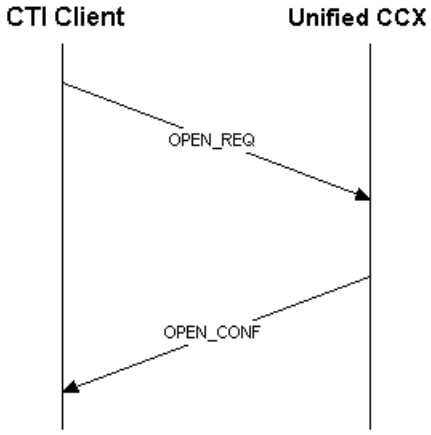
Clients attempt to connect to the server until a connection is established. Once a connection between the client and the server has been established, the connection remains in place until a failure occurs or the client closes the connection. If a failure occurs, the client may again attempt to establish a connection to the server until a new connection is established.

Connection failures may be detected by the TCP layer (see Reasons for a Connection to Fail, on page 16) or by the heartbeat mechanism (see How a Session is Maintained, on page 17).

How a Session is Initialized

Once a TCP connection has been established, a client attempts to initialize a communications session by sending an OPEN_REQ message, defined in OPEN_REQ, on page 73, to the server. The server responds with an OPEN_CONF message, defined in OPEN_CONF, on page 75, to confirm the successful establishment of a session. Figure 3: Session Initialization Message Flow, on page 16 depicts the message flow.

Figure 3: Session Initialization Message Flow



If the server rejects an OPEN_REQ message, the client must reset the TCP connection. The status code received in the rejection (see Error (E) Status Codes, on page 209) indicates the message data that must be corrected before retrying the attempt to establish a session. It is a good practice to wait for at least 10 seconds before retrying the OPEN_REQ message.

The OPEN_REQ message contains protocol version information. The minimum version number supported by Unified CCX is 10. In general, Unified CCX maintains backward compatibility in new product releases.

Reasons for a Connection to Fail

If for any reason the server determines that a new session must not be opened, it responds to the OPEN_REQ message with a FAILURE_CONF message.

The following are some of the reasons a new session might not be opened:

- If required floating data has not been provided, a FAILURE_CONF message, defined in FAILURE_CONF, on page 83, is returned with the status code set to E_CTI_REQUIRED_DATA_MISSING.
- If a client attempts to open a session for CLIENT EVENTS service (that is when the client is in Agent Mode — see Two Client Modes for Connecting with Unified CCX, on page 19) and the provided IP phone information items are not consistent with each other, a FAILURE_CONF message is returned with the status code set to E_CTI_INCONSISTENT_AGENT_DATA.
- If the indicated IP phone is already associated with a different client, the server refuses to open the new session and returns a FAILURE_CONF message with the status code set to E_CTI_DEVICE_IN_USE.

- If a connection is made to the non-master server, the server will reply with a FAILURE_CONF indicating error E_CTI_SERVER_NOT_MASTER.

Note That a session does not fail does not mean the client was granted everything requested. The server grants the client only what is available.

How a Session is Maintained

To detect and handle transmission failures, the client must send a HEARTBEAT_REQ message, see HEARTBEAT_REQ, on page 77, to the server whenever no messages have been sent for the heartbeat interval or sooner. On receipt of a HEARTBEAT_REQ message, the server immediately responds with a HEARTBEAT_CONF message, defined in HEARTBEAT_CONF, on page 77. If three heartbeats go unconfirmed, the client must declare a session failure and reset the TCP connection.

The heartbeat interval is solely determined by the client and represents a reasonable balance between the speed of failure detection and the network bandwidth consumed by heartbeat messages and their corresponding confirmations. A maximum value of 30 seconds is accepted by the server

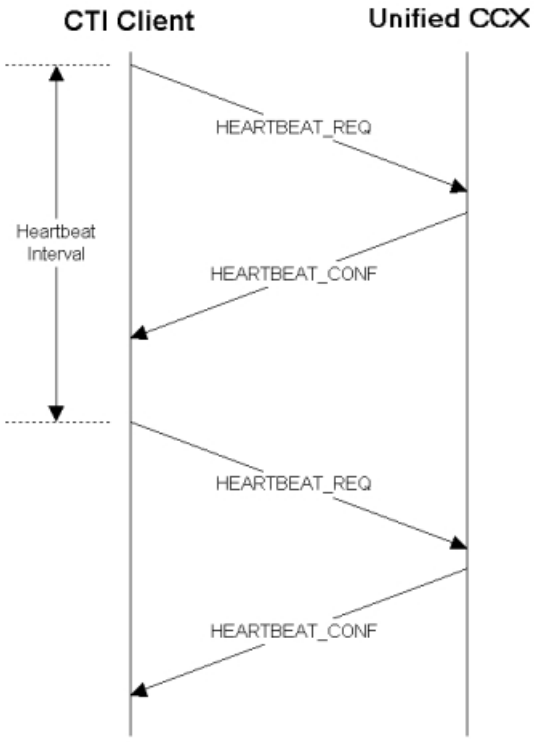
The server does not initiate HEARTBEAT_REQ messages. Failure detection on the server is accomplished using the IdleTimeout value from the OPEN_REQ message. When heartbeat messages are implemented, the client must set this value to four times the heartbeat interval. If the server does not receive any messages (including HEARTBEAT_REQ messages) for this period of time, the server declares a session failure and resets the TCP connection.

The server may respond to a HEARTBEAT_REQ message with a FAILURE_CONF message. This indicates to the client that the server is off-line, and the client must reset the TCP connection.

Note Heartbeat messages must be sent regardless of line activity or in-activity.

Figure 4: Heartbeat Message Flow, on page 18 depicts the heartbeat message flow.

Figure 4: Heartbeat Message Flow



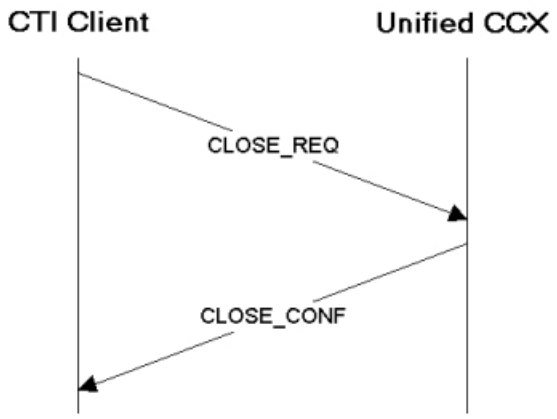
Note You need to synchronize the time of the client and the server and you need to make sure the intervals between the heartbeats are identical.

How a Session is Ended

The client may initiate the graceful termination of a communication session. The client sends a CLOSE_REQ message, defined in CLOSE_REQ, on page 79; the server responds with a CLOSE_CONF message, defined in CLOSE_CONF, on page 79. On receipt of the CLOSE_CONF message, the client resets the TCP connection. The client should wait up to 5 seconds for the CLOSE_CONF message before resetting the connection.

Figure 5: Session Termination Message Flow, on page 19 depicts the session termination message flow.

Figure 5: Session Termination Message Flow



Two Client Modes for Connecting with Unified CCX

Unified CCX provides support for two different ways of connecting to clients:

bridge mode (that is, all events mode)

A bridge mode client receives all agent-state and call events for all logged in agents in the system.

A bridge mode client receives all unsolicited messages and response messages that the client invoked by request messages. The client does not receive response messages invoked by other clients.

agent mode (that is, client mode)

An agent-mode client represents an agent's connection and receives only events that apply to the agent represented by the connection.

An agent mode client receives unsolicited messages that may affect the agent. For example, if an agent device is in a conference call, call events generated by any device in this conference call may be received by this client. In addition, the client receives all response messages that are a result of request messages that the client sent.

You set the mode in the OPEN_REQ message through specifying a mask:

- The CTI_SERVICE_ALL_EVENTS (0x00000010) selection in the Service Masque specifies the client is in bridge mode.
- The CTI_SERVICE_CLIENT_EVENTS (0x00000001) selection in the Service Masque specifies the client is in agent mode. A client connected in the agent mode uses the bit "CTI_SERVICE_CLIENT_EVENTS" in the Service Mask.

These two bits in the mask are mutually exclusive, which means that a client can either be connected in bridge mode or in agent mode.

Note Client connection modes are determined when a session is initialized.

Once you set a mode for a session, you cannot change it during that session. To change the mode, you need to create a new session.

How the Client Selects the Types of Messages Wanted

Unified CCX is capable of providing much more real-time data than the typical client needs. Message masks have been provided to avoid wasting network bandwidth by suppressing the transmission of unneeded data. Carefully consider the network impact of the expected number of simultaneously connected clients before deploying a client application that unmasks a large number of messages.

Within the OPEN_REQ message, there are four separate mask types for selecting the type of messages wanted and filtering out unwanted messages.

ServicesRequested

A bitwise combination of the Services that the client is requesting.

For example, if a client wants to use CALL_DATA_UPDATE, CLIENT_CONTROL and CLIENT_EVENTS, the Service Mask will be 0x7 which is a bit wise combination of 0x00000002 + 0x00000004 + 0x00000001. See CTI Service Masks, on page 80 for a list of all the service masks that you can use in an OPEN_REQUEST message.

In the OPEN_CONF message, the ServicesGranted field contains the services that Unified CCX provides to the client. The client must check this field and not assume that services requested are always granted.

CallMsgMask

A bitwise combination of the unsolicited call event message masks that the client wishes to receive.

For example, if a client wants to receive CALL_DELIVERED_EVENT and CALL_ESTABLISHED_EVENT, the CallMsgMask will be 0x5 which is a bit wise combination of 0x00000001 + 0x00000004. See Unsolicited Call-Event Message Masks, on page 80 for a list of all the unsolicited call event messages you can use in an OPEN_REQ message.

AgentStateMask

A bitwise combination of agent state masks that the client wishes to receive.

For example, if a client wants to receive all AgentStateEvents, the AgentStateMask will be 0x3FFF which is a bit wise combination of Agent State Mask. See Agent State Masks, on page 82 for a list of all the agent state masks that you can use in an OPEN_REQ message.

ConfigMsgMask

A bitwise combination of configuration events that the client wishes to receive.

For example, if a client wants to receive all type configuration update messages, the ConfigMsgMask will be 0x0000001F. See Configuration-Information Masks, on page 83 for a list of all the configuration event masks you can use in an OPEN_REQ message.

These masks represent the type of messages the client wishes to receive. Clients may also control what configuration events to receive in the configuration messages.

The masks specified in the OPEN_REQ are used to ensure that clients do not receive unexpected messages. See Masks Used in the OPEN_REQ Message, on page 79 for full descriptions of each message mask.

Note Whatever services are granted or not granted is reflected in the returned OPEN CONF message.

Reasons for the Unified CCX to Send SYSTEM_EVENT Messages

Unified CCX sends a SYSTEM_EVENT message (see SYSTEM_EVENT, on page 77) to all clients to indicate its status when:

- Unified CCX changes its status.
- The agent's device changes its status.

System Event messages are not controlled by bridge or agent mode and are sent to all clients.

The Unified CCX CTI Server in a High Availability Unified CCX System

High availability is a feature in CRS from release 4.0 onwards, except for the CRS 4.5 release.

A Unified CCX cluster consists of identically configured servers. Only one server is tagged as the master server. The master server makes decisions global to the cluster, like keeping track of calls, agent states, and maintaining socket connections to all CTI clients. At run-time only one of the servers can be the master server at a given time. The standby servers do not process any events on their own, with the possible exception of system events applicable to the local server.

Note While CRS 4.0 and 4.1 support up to 10 servers, CRS 4.5 supports only one server and CRS 5.0 supports 2 servers.

Since a standby server does not manage any CTI clients, it does not receive any CTI events. If an attempt is made to connect to the slave server with an OPEN_REQ message, it responds with a FAILURE_CONF message with a status code of E_CTI_SERVER_NOT_MASTER.

If the master server fails, one of the standby servers becomes the new master server. Since the servers are configured identically, there is no need for a fail back unless the new master server does not have the same capacity as the original one. This would be the case if a different type of hardware is used to install the Unified CCX software in the cluster. Agents are re-connected to the new master server in case of failure of the master server. The TCP port number is specified in the System Parameters web page in the UCCX Administration. The field is *RmCm TCP Port**.

Connect CTI Sessions in a Clustered Unified CCX

The following procedure connects a CTI session in a clustered Unified CCX with a given a set of IP addresses and ports which represent the nodes in the cluster:

Procedure

Step 1 Select the first CCX IP address and port.

Step 2 When the CTI client attempts to open a session at the specified IP address and port, there may be any of the following three scenarios:

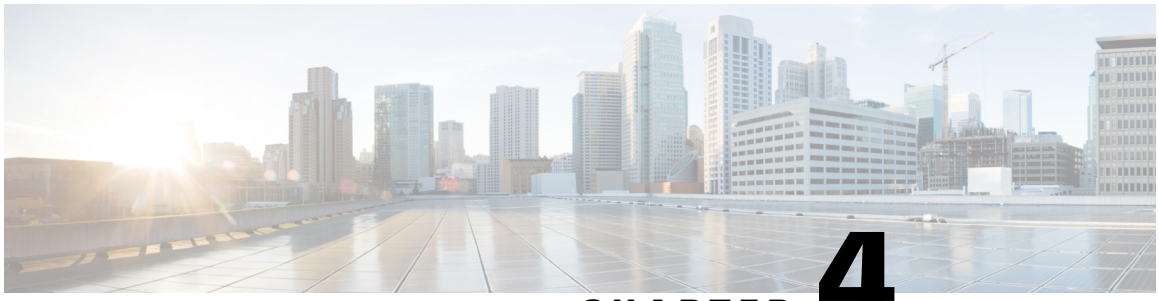
- Session opens and CTI client receives OPEN_CONF. Go to Step 4, on page 22.
- Socket error / no response.
- Receive FAILURE_CONF.

Step 3 Select next IP address and port. Go to Step 2, on page 22

Step 4 The CTI client should send periodic HEARTBEAT_REQs and should receive messages

Step 5 When the CTI client detects a socket error, or missing HEATBEAT_CONF, or receive a FAILURE_CONF/SYSTEM_EVENT, it should close the current session and go to Step 3, on page 22.

Note CCX will send FAILURE_CONF/SYSTEM_EVENT in various failure scenarios. The HEARTBEAT_CONF is unlikely to be missed, however if CTI client misses the HEARTBEAT_CONF, the CTI client should wait until 2-3 HEARTBEAT_CONFs are missed consecutively.



CHAPTER 4

Managing Configuration Data

This chapter includes the following topics:

- Why Configuration Data Is Shared, page 23
- Configuration Data That Can Be Shared, page 23
- Configuration Message Summary Descriptions, page 24
- Share Configuration Data, page 26
- How Configuration Consistency is Maintained, page 28
- Share Agent Team Configuration Data, page 29
- Error Handling, page 29

Why Configuration Data Is Shared

Clients can acquire configuration data from the CTI server and use configuration messages to synchronize that data between Unified CCX and themselves. This includes an initial bulk upload of configuration data and subsequent dynamic updates whenever there is a configuration change made through the RMCM subsystem in the Unified CCX Administration software on Unified CCX.

Configuration Data That Can Be Shared

Currently, the Unified CCX CTI protocol supports the CTI client retrieval of only the following configuration data.

CSQ (Contact Service Queue)

A CSQ controls incoming calls by determining where an incoming call is to be placed in the queue and to which agent the call is sent.

Unified CCX Application

Unified CCX applications interact with contacts and perform such functions as prompting callers for information, transferring calls, providing information to callers, and so on.

Agent

An agent is the person in the contact center who handles customer calls, e-mail, and other types of correspondence between the customer and the company.

Device (agent extensions, application triggers and CTI ports)

Call connection points in the computer system.

Team

Agent team information. A team may include agents, CSQs, a primary supervisor, and a secondary supervisor.

Configuration Message Summary Descriptions

Table 1: Unified CCX CTI Configuration Message List, on page 24 lists the configuration messages and Table 2: Configuration Message Descriptions, on page 25 describes the messages. The rest of this chapter explains how to use them. See Message Type Definitions for detailed descriptions of the message contents.

Message List

Table 1: Unified CCX CTI Configuration Message List

Request	Response
CONFIG_REQUEST_KEY_EVENT	CONFIG_KEY_EVENT
CONFIG_REQUEST_EVENT	CONFIG_BEGIN_EVENT CONFIG_END_EVENT CONFIG_CSQ_EVENT CONFIG_APPLICATION_EVENT CONFIG_AGENT_EVENT CONFIG_DEVICE_EVENT
TEAM_CONFIG_REQ	TEAM_CONFIG_EVENT TEAM_CONFIG_CONF

Message Summary Descriptions

Table 2: Configuration Message Descriptions, on page 25 briefly describes the configuration messages. For complete definitions of these messages, see the referenced definition link for each message.

Table 2: Configuration Message Descriptions

Message	Sent by	Description
Configuration initialization messages		
CONFIG_KEY_EVENT, on page 85	Client	Requests configuration data from the server. The type of update data received depends on the ConfigInfoMask used in this request. The ConfigInfoMasks used to indicate the type of data requested are: 1 = Application Information 2 = CSQ Information 4 = Agent Information 8 = Device Information 0 = No initial configuration information upload is wanted, but update configuration data is wanted. This is used when the client does not want the initial update messages but does want, at the time of this message, configuration update messages.
CONFIG_KEY_EVENT, on page 85	Unified CCX	Response to the client CONFIG_REQUEST_KEY_EVENT message. Contains the requested Unified CCX configuration keys at the time of the CONFIG_REQUEST_KEY_EVENT.
CONFIG_REQUEST_EVENT, on page 86	Client	Response to the server CONFIG_KEY_EVENT message. Tells the server if it wants the requested initial snapshot of the configuration or only updates.
Configuration data messages		
CONFIG_BEGIN_EVENT, on page 87	Unified CCX	Indicates the beginning of configuration data (all data of the same type of the same key) from the server.
CONFIG_END_EVENT, on page 88	Unified CCX	Indicates one of the following: • The end of a successful configuration upload. In this case, it follows configuration records preceded by a CONFIG_BEGIN_EVENT message. • No configuration for the items requested. • An error condition. This and the preceding message may be in response to a CONFIG_REQUEST_EVENT message.

Message	Sent by	Description
CONFIG_CSQ_EVENT, on page 89	Unified CCX	Sends CSQ configuration data to a client.
CONFIG_APPLICATION_EVENT, on page 90	Unified CCX	Sends application configuration data to a client.
CONFIG_AGENT_EVENT, on page 92	Unified CCX	Sends agent configuration data to a client.
CONFIG_DEVICE_EVENT, on page 93	Unified CCX	Sends device configuration data to a client.
Configuration team messages		
TEAM_CONFIG_REQ, on page 96	Client	Requests team configuration from the server.
TEAM_CONFIG_EVENT, on page 95	Unified CCX	Sends team configuration data to a client.
TEAM_CONFIG_CONF, on page 97	Unified CCX	Indicates the end of team configuration data or an error condition.

Share Configuration Data

Note The following process applies to application, CSQ, agent, and device configuration. For agent team data, see Share Agent Team Configuration Data, on page 29.

For a client to get configuration data:

Procedure

Step 1 The client must include the CTI_SERVICE_CONFIG_EVENTS bit (0x00040000) in the ServiceRequested field of an OPEN_REQ message if the client wants configuration data. The Unified CCX CTI server does not provide configuration data to the client unless that bit is set. In the OPEN_REQ message, the client must also include the configuration message mask indicating the type of configuration data that it wants.
The client cannot send a configuration request before the server is online. The status of the server is shown in the OPEN_CONF reply to the OPEN_REQ message or in a SYSTEM_EVENT message.

Step 2 The client sends a CONFIG_REQUEST_KEY_EVENT message to the server to request a set of current configuration keys (see CONFIG_REQUEST_KEY_EVENT, on page 85).

Step 3 In response to the CONFIG_REQUEST_KEY_EVENT message, the server replies with a CONFIG_KEY_EVENT message containing the requested keys (see CONFIG_KEY_EVENT, on page 85).

Step 4 Based on the keys returned in the CONFIG_KEY_EVENT message, the client decides if it needs an initial snapshot of the configuration.

Step 5 If the client needs an initial snapshot of the configuration, it sends the CONFIG_REQUEST_EVENT message with the requested configuration mask. Otherwise, the client sends this message with the configuration mask set to zero, indicating that the client configuration is in sync with the server configuration and that it only needs configuration updates.

Step 6 If in Step 5, the initial snapshot of the configuration is requested, the server sends the initial snapshot of the configuration data to the client in a CONFIG_BEGIN_EVENT and CONFIG_END_EVENT block of messages. In between these two messages, the server sends the appropriate message for the type of data requested (see CONFIG_BEGIN_EVENT, on page 87 and CONFIG_END_EVENT, on page 88).
If all the data does not fit in one message, the server sends multiple messages of the same type. In this case, each data message is called a record. The whole transaction from the begin event to the end event is called a data block.

The server sends configuration data messages in the following order:

CSQ data

Application data

Agent data

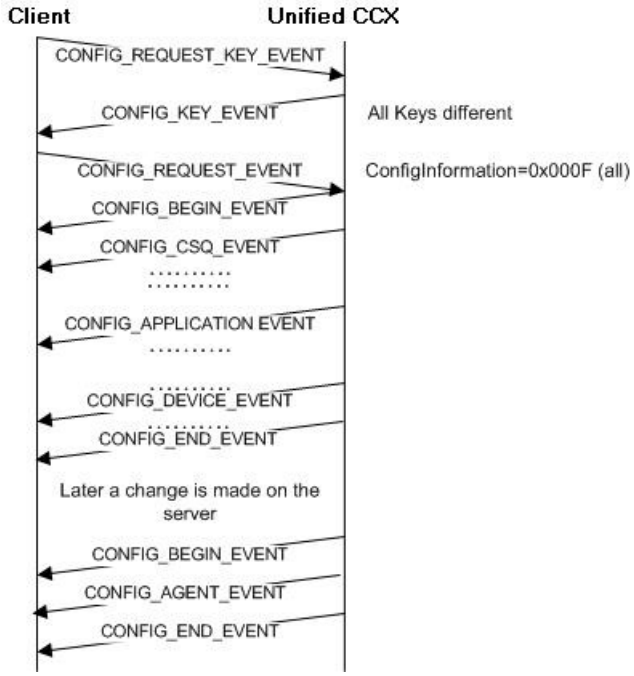
Device data

However, the order in which the client receives the data is irrelevant. The client may receive the data messages in any order depending on the timing and number of messages sent. What is important is that all the messages between a begin and end message contain consistent data. That is, for example, if a CSQ message is sent with an Agent message in the same data block, the agent referenced belongs to that CSQ.

Step 7 Configuration updates performed in the Unified CCX Administration web page are sent to clients in a CONFIG_BEGIN_EVENT and CONFIG_END_EVENT block of messages as well.

Figure 6: Configuration Message Flow, on page 28 illustrates the preceding steps

Figure 6: Configuration Message Flow



What to Do Next

Note During an initial configuration upload, if the server needs to break up the configuration into multiple CONFIG_BEGIN_EVENT/CONFIG_END_EVENT messages, a status 5 is listed in the CONFIG_END_EVENT message until all the data is sent. A status of 0 is included in the final CONFIG_END_EVENT message to indicate a successful data upload.

How Configuration Consistency is Maintained

The consistency of configuration data is managed through configuration keys. Unified CCX server always increments configuration keys when a configuration update happens in the Unified CCX Administration. The set of keys are sent to clients in the CONFIG_KEY_EVENT message and the CONFIG_BEGIN_EVENT message. Clients may store these keys.

The keys are 8-byte values and are stored as long data in Unified CCX. Configuration keys apply to the configuration events determined by the mask bit CTI_SERVICE_CONFIG_EVENTS of the ServicesRequested field of the OPEN_REQ message only.

If a client has the same set of keys as the server, it indicates that the client configuration is in sync with the server's configuration.

Note Agent Team configuration data is the exception and is not managed by these keys.

Share Agent Team Configuration Data

Procedure

Step 1 The client must include the CTI_SERVICE_SUPERVISOR bit (0x00000080) in the ServiceRequested field of an OPEN_REQ message if the client wants team configuration data. The CCX CTI server does not provide team update configuration data to the client unless that bit is set.

Step 2 When a client needs team configuration data, it sends a TEAM_CONFIG_REQ message to the Unified CCX server.
If the "configparam" field in the TEAM_CONFIG_REQ message is set to 0x1, it takes the initial configuration.

Note Clients must use the TEAM_CONFIG_REQ to request team configuration information. This information is not a part of the CONFIG_REQUEST_EVENT, as shown in Table 1: Unified CCX CTI Configuration Message List, on page 24, and must be requested explicitly.

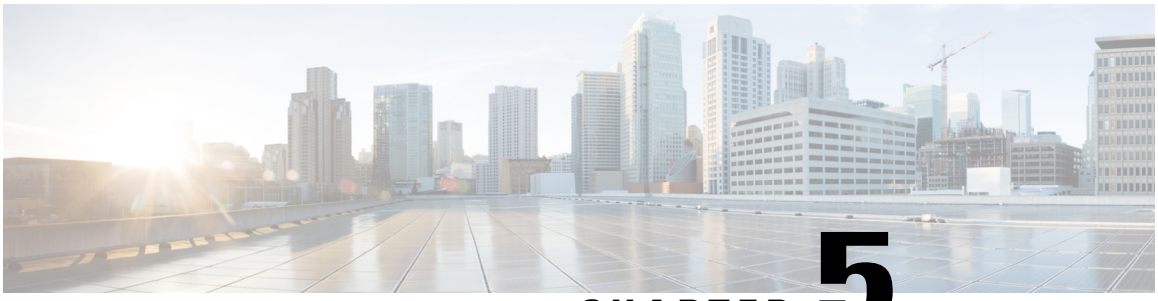
Step 3 The server responds with zero or more TEAM_CONFIG_EVENT messages, followed by a TEAM_CONFIG_CONF message.
The TEAM_CONFIG_CONF messages mark the end of the initial configuration upload. After that, the client receives updates only for the message session.

Step 4 After this request/confirmation exchange, the Unified CCX server sends TEAM_CONFIG_EVENT messages to the client whenever there is a team configuration change in the Unified CCX Administration web page.

Error Handling

As Unified CCX and clients exchange configuration messages, Unified CCX may encounter errors. These errors are conveyed to clients through configuration messages or a generic FAILURE_CONF message and a FAILURE_EVENT message. For example, the status code in the CONFIG_KEY_EVENT message and the CONFIG_END_EVENT message may indicate an error condition in Unified CCX. Client applications are expected to check these error code and error messages. When a client detects an error condition in Unified CCX, the client needs to start over the configuration data sharing process. It is required that the client waits for at least 10 seconds before starting the process.

Although the configuration data sharing process is designed to maintain configuration consistency using the configuration key, a client developer may want to design a mechanism to overwrite the configuration key mechanism in case of unforeseen conditions in which the Unified CCX configuration and client configuration are out of sync. For example, a client developer may create a button such that whenever that button is clicked, the client starts the configuration data sharing process. In this process, the result of key comparison is ignored and CONFIG_REQUEST_EVENT message is always sent.



CHAPTER 5

Managing Agent States

This chapter includes the following topics:

- What is an Agent State?, page 31
- The Causes of Agent-State Transitions, page 33
- Agent-State Message Summary Descriptions, page 34
- Agent Authentication Without Agent State Change, page 35
- About Forced Agent Logout and Forced Agent Login, page 35

What is an Agent State?

An agent is a person who has a phone on his desktop and receives calls, transfers calls, ends calls, and so on. The state of this agent and his phone are represented within the Unified CCX system by agent states and agent state events.

The agent state is what is described by the last received agent state event. Actions performed by the agent and his phone eventually translate into updated agent states. For example, when an agent hangs up his phone after working on a call, his state changes from talking to ready. The change in the state of this agent is communicated with the system through the AGENT_STATE_EVENT message.

The possible agent states are listed in Table 3: Agent States and Their Message Values, on page 32. The AGENT_STATE_EVENT, the QUERY_AGENT_STATE_CONF, and the SET_AGENT_STATE_REQ messages indicate an agent state by the values listed in Table 5: Example Values in a SET_AGENT_STATE_REQ Message, on page 35.

CCX maintains all agent states and may change an agent state based on various conditions such as the state of an agent phone.

Unified CCX CTI clients are also allowed to change agent states by a client using the SET_AGENT_STATE_REQ message. When Unified CCX receives this request, it processes the request based on the current state of the system. As a result, the SET_AGENT_STATE_REQ message may or may not actually change the agent state. For example, the client might request that an agent's state be changed from not ready to ready, but Unified CCX may not be able to change the agent's state if the agent has not yet logged in.

A Unified CCX CTI client must always monitor the AGENT_STATE_EVENT message or use the QUERY_AGENT_STATE_REQ message to obtain the current agent state.

Table 3: Agent States and Their Message Values

State name	Description	Value
AGENT_STATE_LOGIN	There is no specific Agent Login state in Unified CCX. However, this is a request value specified while trying to login an agent.	0
AGENT_STATE_LOGOUT	The agent has logged out of the ACD and cannot accept any additional calls.	1
AGENT_STATE_NOT_READY	The agent is not available to accept a routed call.	2
AGENT_STATE_READY	The agent is available to accept a routed call.	3
AGENT_STATE_TALKING	The agent is currently talking on a call with a customer or another agent (inbound, outbound, or inside). This state is automatically set for the agent by the ACD.	4
AGENT_STATE_WORK	The agent is completing work from a previous call and is unavailable to receive routed calls. There are two (mutually exclusive) ways to put an agent into AGENT_STATE_WORK: • Automatically, through CSQ configuration settings selected in the Unified CCX Administration web page. • Through the SET_AGENT_STATE_REQ message. In this case, when the agent is in the AGENT_STATE_TALKING state, the client can select the work state (by setting the AgentState field to AGENT_STATE_WORK). Then when the call ends, the agent will be moved by the Unified CCX server to the Work state.	5
BUSY_OTHER	This state signifies the agent state for a particular CSQ. This state is notified in the CSQ State field of QUERY_AGENT_STATE_CONF message only. The state of an agent for a CSQ will be BUSY_OTHER if that agent is handling calls from other CSQs.	7

State name	Description	Value
AGENT_STATE_RESERVED	The agent is reserved for a call that will arrive at the ACD shortly. The agent is temporarily set aside to receive a specific call. The agent's state is changed to the Talking state when the agent answers the call. If the agent fails to answer the call within a time limit specified by the system administrator, the ACD places the agent in a Not Ready state. The Reserved state is automatically set for the agent by the ACD. The agent can be in this state without the phone ringing (if the agent is waiting for it to ring).	8
AGENT_STATE_UNKNOWN	The associated agent state is unknown.	9

The Causes of Agent-State Transitions

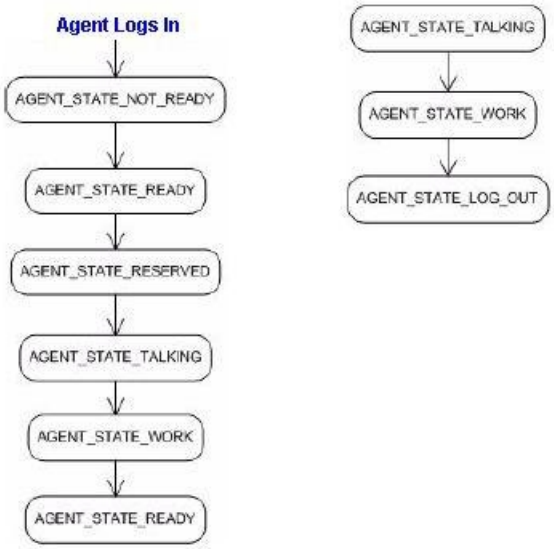
Call events and events from the Cisco Agent Desktop (CAD) and the agent IP phone change the agent state.

Figure 7: Example Agent State Transitions, on page 34 contains two example agent state transitions, one at the beginning of the day when the agent logs in and one when the agent logs out. In that figure:

1. The agent logs in and is immediately recorded as Not Ready.
2. When the agent is Ready, the client sets the agent to this state by using the SET_AGENT_STATE_REQ message.
3. When Unified CCX sends a call to the agent, the agent's state becomes Reserved.
4. As soon as the agent answers the call, the agent's state becomes Talking.
5. The agent indicates any after call work by using the SET_AGENT_STATE_REQ message.

6 The client sets the agent state to Ready by using the SET_AGENT_STATE_REQ message with the Ready state when the agent is available for the next call.

Figure 7: Example Agent State Transitions



Agent-State Message Summary Descriptions

Table 4: Agent-State Messages, on page 34 lists the agent state messages. See the referenced links for the message definitions. If Unified CCX fails to process the SET_AGENT_STATE_REQ message, it responds with a CONTROL_FAILURE_CONF message.

Table 4: Agent-State Messages

Message	Sent by	Purpose
AGENT_STATE_EVENT, on page 98	Server	Notification of new agent state. AGENT_STATE_EVENTs are sent to all bridge-mode sockets and the socket through which the agent specified in the event is connected.
QUERY_AGENT_STATE_REQ, on page 100	Client	Request to obtain the current state of an agent.
QUERY_AGENT_STATE _CONF, on page 101	Server	Response to a QUERY_AGENT_STATE request.
SET_AGENT_STATE_REQ, on page 102	Client	Request to alter the current state of an agent

Message	Sent by	Purpose
SET_AGENT_STATE_CONF, on page 103	Server	Response confirming a previous SET_AGENT_STATE_REQ request.

Agent Authentication Without Agent State Change

Typically, to authenticate an agent, the agent ID, password, and the agent's extension are required, and the agent state is changed.

However, if an agent has to be authenticated using just the agent ID and password without changing the agent state, the client must send a SET_AGENT_STATE_REQ with the value of the ForcedFlag field set to 2 and AgentInstrument set to Unknown. The SET_AGENT_STATE_CONF message confirms successful completion of the request.

Table 5: Example Values in a SET_AGENT_STATE_REQ Message, on page 35 shows the values that must be set in the SET_AGENT_STATE_REQ message to allow for agent authentication without providing an extension.

Table 5: Example Values in a SET_AGENT_STATE_REQ Message

Field	Value
reserved	1
AgentState	AGENT_STATE_LOGIN (0)
reserved	0
reserved	0
EventReasonCode	0
ForcedFlag	2
AgentExtension	"unknown"
AgentID	"agentid" // Enter the agent ID.
AgentPassword	"password" // Enter the agent password.

About Forced Agent Logout and Forced Agent Login

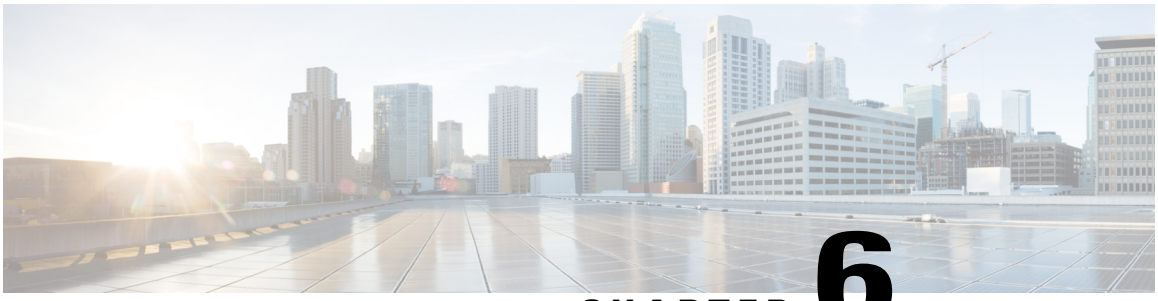
In general, if an agent is in the talking state, that agent state cannot be changed to the logout state.

However there are cases when an agent logout is mandatory. In these cases, the forced flag must be set to 1 in the SET_AGENT_STATE_REQ message.

For example:

- When a windows-based client closes, regardless of the current state of the agent, the agent state must be changed to Logout.
- In a contact center environment, some agents may share the same phone extension since they work on different shifts. In this case, if agent A is talking to a customer, then agent B with the same extension is not able to set his or her state to login. Instead, agent B must set the force flag to 1 in the SET_AGENT_STATE_REQ message in order to set his or her state to login. This message also automatically logs out Agent A, though it does not terminate Agent A's phone call.

Note Even though Agent A is automatically logged out, that agent can still finish talking with the customer. However, Agent B is now listed as the Agent with that extension number and the next call for that extension goes to Agent B.



CHAPTER 6

Communicating Call Events

This chapter includes the following topics:

- Call Events, page 37
- Call Context Data, page 38
- Call-Event Message Summary Descriptions, page 38
- The Unspecified Flow of Call-Event Messages, page 41
- Fields Common to All Call-Event Messages, page 43
- CallType Fields, page 43

Call Events

Any telephony device can generate a call. Unified CCX, however, knows only the call that is associated with a Unified CCX route point, a Unified CCX CTI port, and a Unified CCX agent phone device into which an agent is logged.

A call connection is the connection between both a caller and a receiver. If a conference call is made, then there are more than two people connected at the same time. Each phone that is part of the call makes up the connection.

Call events are generated by actions relating to calls and are done on the caller's or callee's telephony devices. For example, when a phone rings, a call delivered event is generated. Every action of the caller and callee can also generate a call event. For example the caller or callee can put a call on hold or transfer the call.

Unified CCX generates call events only for the telephony devices that it monitors. Unified CCX normally monitors triggers and agent devices into which agents are logged. In general, if a call contains a monitored device, call events that are relevant to that monitored device are generated.

Unified CCX does not control telephony devices directly. Currently Unified CCX supports two call control products: Cisco Unified Communications Manager and Cisco Unified Communications Manager Express. The procedure to enable an agent device to be monitored depends on these call control products. On Unified CCM, CTI capability must be enabled on agent devices and lines. Please refer to the Unified CCX documentation for details.

Note Unified CME is not supported from Unified CCX 9.0 and later.

Call events are unsolicited messages. A Unified CCX client can select what call events to receive by setting appropriate masks in the OPEN_REQ message. In general, a bridge-mode client receives call events related to all monitored devices while an agent-mode client can receive call events only related to calls that include the agent device.

The order of all call events is not guaranteed. However the CTI client can expect an order for some events. For example, the CALL_DELIVERED event is always before the CALL_ESTABLISHED event.

Call Context Data

Call context data is information about a call such as, for example, the caller ID or a caller account number. This data is maintained and updated through call variables in the messages.

Unified CCX maintains a set of 10 call variables for each call. Each variable is capable of storing a null terminated string of up to 39 characters (39 variable characters + null termination character = 40 bytes, STRING[40]). When a call is pre-routed by Unified CCX, these variables are initialized to null strings prior to executing the routing script.

The values of the call variables can be also used by Unified CCX to make routing decisions. In addition, the variables can contain added information about the caller, such as the result of a host database query. While routing a call, the routing script can update one or more of the call variables.

Call variables can also be set by a client that is associated with the call by using the SET_CALL_DATA_REQ, on page 160 message. For example, a client might want to update call data. Unified CCX changes call context data using steps in a Unified CCX script.

A passed call variable with a null string signifies no value. That is, if all the call variables are passed and call variable 2 contains simply a null for a zero length, it will not cause any change to that call variable.

To “blank out” a call variable, a client can enter a single space in the call variable.

Call-Event Message Summary Descriptions

Call-event messages are unsolicited messages generated by the CTI server when a call event occurs and go only from the server to its clients to inform them of the event.

There are no request or confirmation messages associated with unsolicited messages and the content of most of the call-event messages is event specific.

Table 6: Call-Event Messages , on page 39 lists the Unified CCX CTI protocol call-event messages. For each message definition, see Message Type Definitions

Table 6: Call-Event Messages

Message	Cause of Message and Purpose
BEGIN_CALL_EVENT, on page 105	Every call can be announced to the client with an unsolicited BEGIN_CALL_EVENT message, informing the client that it has just been associated with a new call and providing the initial call context data. Additional call and agent state events are then sent to the client as the call is handled. Finally, an END_CALL_EVENT message is sent to the client when its association with a call is dissolved.
END_CALL_EVENT, on page 107	An END_CALL_EVENT message is generated when the association between a call and the Unified CCX client is dissolved and no further call-event messages for the call are sent to this client. **Note** While this message informs the client that a call has been completely removed from Unified CCX, the message does not necessarily indicate that the subject call has been terminated. For example, the Cisco Unified Communications Manager might have forwarded the call to another call center.
CALL_DATA_UPDATE_EVENT, on page 108	Changes to the call context data generate a CALL_DATA_UPDATE_EVENT to the client that contains only the items that have changed. Each call has a set of data (call context data) associated with it. This data could be caller-entered digits or data in a call variable or a call context variable.
CALL_DELIVERED_EVENT, on page 111	The arrival of a call at a monitored device on the server generates a CALL_DELIVERED_EVENT message to the client. The call is delivered when the phone rings.
CALL_ESTABLISHED_EVENT, on page 114	The answering of a call at a device on the server creates a new call connection and generates a CALL_ESTABLISHED_EVENT message to the client.
CALL_HELD_EVENT, on page 116	Placing a call on hold generates a CALL_HELD_EVENT message to the client.
CALL_RETRIEVED_EVENT, on page 117	Resuming a call previously placed on hold generates a CALL_RETRIEVED_EVENT message to the client.
CALL_CLEARED_EVENT, on page 118	A CALL_CLEARED_EVENT message is sent to the client when a call is terminated, normally when the last device disconnects from a call.
CALL_CONNECTION_CLEARED_EVENT, on page 119	When a party drops from a call, a CALL_CONNECTION_CLEARED_EVENT message can be sent to the client.
CALL_ORIGINATED_EVENT, on page 120	The initiation of a call from the server at any monitored device generates a CALL_ORIGINATED_EVENT to the client.

Message	Cause of Message and Purpose
CALL_FAILED_EVENT, on page 121	A CALL_FAILED_EVENT message is usually sent to the client when a call cannot be completed.
CALL_CONFERENCED_EVENT, on page 123	The joining of calls into a conference call generates a CALL_CONFERENCED_EVENT to the client.
CALL_DIVERTED_EVENT, on page 128	The moving of a call from one device to another can generate a CALL_DIVERTED_EVENT to the client.
CALL_TRANSFERRED_EVENT, on page 125	The transfer of a call to another destination causes a CALL_TRANSFERRED_EVENT message.
CALL_SERVICE_INITIATED_EVENT, on page 129	The initiation of telecommunications service ("dial tone") at a device generates a CALL_SERVICE_INITIATED_EVENT to the client.
CALL_QUEUED_EVENT, on page 131	The placing of a call in a queue pending the availability of some resource generates a CALL_QUEUED_EVENT message to the client.
CALL_DEQUEUED_EVENT, on page 133	The explicit removal of a call from a queue generates a CALL_DEQUEUED_EVENT message to the client.
RTP_STARTED_EVENT (OPTIONAL), on page 134	The RTP_STARTED_EVENT message indicates that an RTP (Real-Time Protocol) media stream (voice) has been started. Since there are two media streams for audio media, there are also two RTP Started events, one indicating the input has started (that is, the phone is listening) and the other that the output has started (that is, the outgoing media from the agent phone has begun). Depending on the Unified CCX configuration, Unified CCX may or may not send this messages. For example, Cisco Unified Communications Manager sends this message, but Cisco Unified Communications Manager Express does not send the message.
RTP_STOPPED_EVENT (OPTIONAL), on page 136	The RTP_STOPPED_EVENT message indicates that an RTP media has been stopped. Since there are two media streams for audio media, there are also two RTP Stopped events, one indicating the input has stopped (that is, the phone is not listening) and the other that the output has stopped (that is, the outgoing media from the agent phone has stopped). Depending on the Unified CCX configuration, Unified CCX may or may not send this messages. For example, Cisco Unified Communications Manager sends this message, but Cisco Unified Communications Manager Express does not send the message.

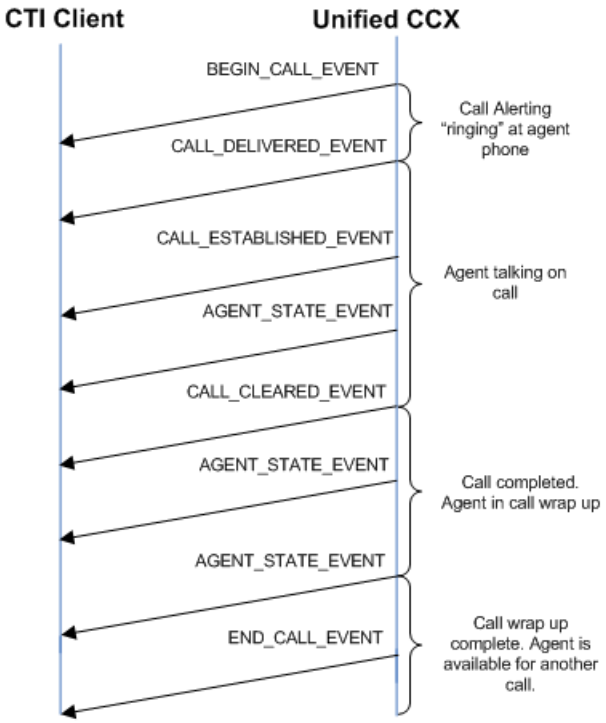
The Unspecified Flow of Call-Event Messages

The relative order of call-event messages and any corresponding agent-state-change event messages is not specified. An agent-state-event message indicating the agent is in the "talking" state, for example, might be sent before or after the corresponding call established event message.

When an event occurs that would conceptually result in two or more event messages being generated at the same time, the client must be prepared to handle the messages arriving in any order. For example, an agent answering an inbound call might generate both a CALL_ESTABLISHED_EVENT and an AGENT_STATE_EVENT message. These can be received by a client in either order, and other event messages can be sent to the client in between.

Figure 8: A Typical Unsolicited Call-Event Message Flow, on page 42 shows a typical unsolicited call-event message flow.

Figure 8: A Typical Unsolicited Call-Event Message Flow



ROUTE_REQUEST_EVENT

Fields Common to All Call-Event Messages

The following fields are in all call-event messages:

- Connection fields that identify a call connection:
  - A numeric **CallID** that identifies a call.
  - A string **ConnectionDeviceID** that identifies each connection of a device to a call.
  - A numeric **ConnectionDeviceType** value that identifies the type of device to which a call is connected.
- An EventCause field that can provide a reason for the occurrence of an event. However, in most cases, no EventCause information is supplied. This is represented by the value CEC_NONE. See Call EventCause (CEC) Values, on page 193, for a list of all the EventCause codes that can be reported.

CallType Fields

Some Call-Event messages contain a CallType field that provides the general classification of a call.

Call Types are classified by Unified CCX based on the following scenarios.

CALLTYPE_CONSULT

Whenever a call is originated by placing a consult (even if the call goes off-switch or hits an unmonitored device).

CALLTYPE_AGENT_INSIDE and CALLTYPE_OUT

Whenever an agent places a new call (non-consult), the default is CALLTYPE_AGENT_INSIDE and the call is reclassified to CALLTYPE_OUT if the calls goes off-switch.

CALLTYPE_ACD_IN

All inbound (queued) calls.

CALLTYPE_TRANSFERRED_IN and CALLTYPE_CONFERENCE

All ACD calls (CallType=CALLTYPE_ACD_IN) that are transferred or conferenced.

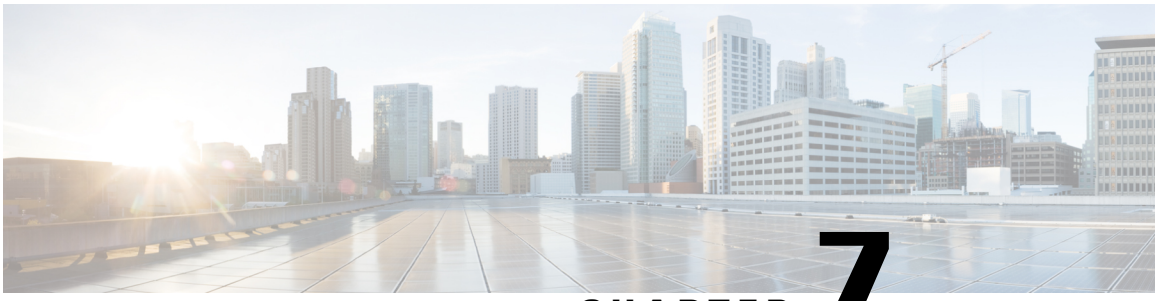
CALLTYPE_OTHER_IN

Non-ACD-Calls coming to an agent, from outside or from unmonitored devices.

CALLTYPE_ASSIST, CALLTYPE_SUPERVISOR_BARGE_IN, CALLTYPE_SUPERVISOR_INTERCEPT, and CALLTYPE_EMERGENCY

Supervisor Assists, Barge-ins, Intercepts, and Emergency calls.

See CallType Values, on page 197, for a list of all the CallTypes with their meanings.



CHAPTER 7

Managing Client Control of Calls

This chapter includes the following topics.

- What is a Call-Control Message?, page 45
- The Call-Control Message Flow, page 46
- Call-Control Message Summary Descriptions, page 46

What is a Call-Control Message?

Call-control (also called “client-control”) messages are from client applications requesting, establishing, answering, controlling, or terminating calls on behalf of a specified agent phone and manipulating the telephony features associated with the agent’s phone.

These messages invoke a message response from the server. Consequently the messages are paired request/response messages. A request message for the desired control action is sent, and the outcome of the request is indicated by the type of response message that is received.

Depending on the specifics of the request, as much as 10 or 15 seconds might elapse before the response message is returned. A successfully executed request is indicated by the corresponding control-action confirmation message, while an unsuccessful request is indicated by a CONTROL_FAILURE_CONF message.

You can use call-control messages to do the following:

- Initiate a new call.
- Answer a call.
- Drop a call.
- Put a call on hold.
- Take back a call from the hold state.
- Put one call on hold and answer another call.
- Drop the connection to a consult agent and reconnect to the original call.
- Initiate a call transfer operation.
- Initiate a conference operation.

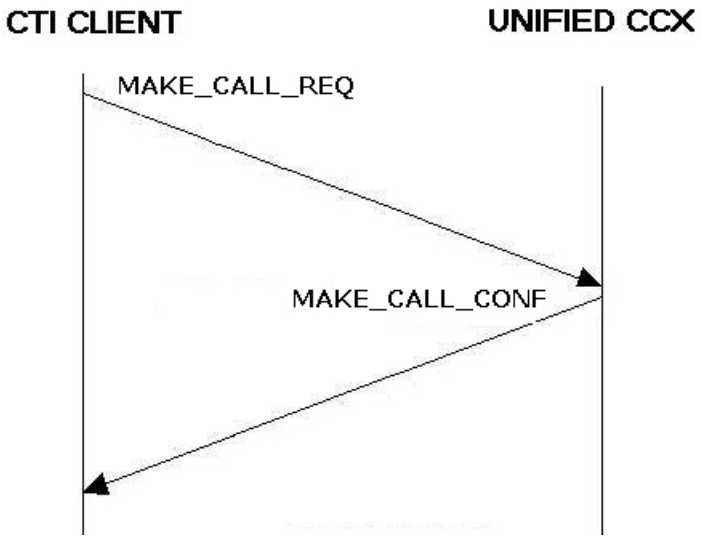
- Transmit Dual Tone Multi-Frequency (DTMF) tones, as for example, to enter a phone number or an account number.

The Call-Control Message Flow

There is no specified order to the flow of call-control messages.

Unsolicited call or agent event messages that are caused by a client request may be received before or after the request confirmation message. Figure 9: Call-Control Message Flow, on page 46 illustrates the general call-control message flow. In this example, the messages control the agent state.

Figure 9: Call-Control Message Flow



Call-Control Message Summary Descriptions

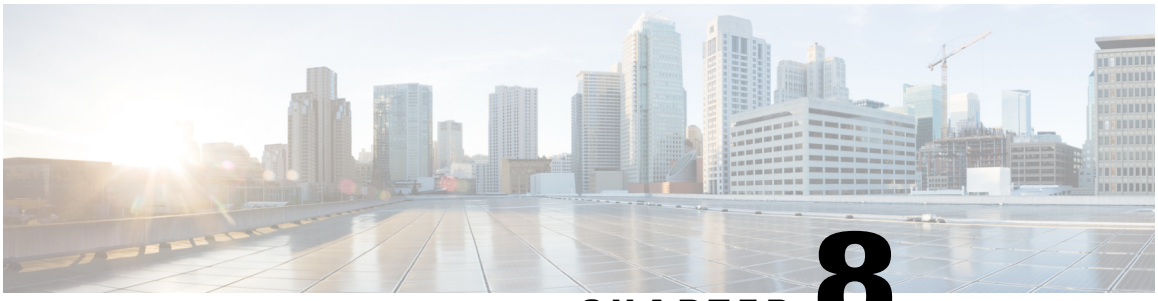
Figure 9: Call-Control Message Flow, on page 46 lists the call-control messages in the Unified CCX CTI protocol. See Message Type Definitions for each message definition.

Table 7: Call-Control Message List

Message	Purpose
CONTROL_FAILURE_CONF, on page 139	Indicates that the previously requested control-service function identified by the given invokeID was unsuccessful.
QUERY_DEVICE_INFO_REQ, on page 173	Allows a client to retrieve general information about a specified device.
QUERY_DEVICE_INFO_CONF, on page 174	Confirms the successful completion of the QUERY_DEVICE_INFO_REQ message.

Message	Purpose
ALTERNATE_CALL_REQ, on page 139	Allows a client to request the combined action of placing an active call on hold and then either retrieving a previously held call or answering an alerting call at the same device.
ALTERNATE_CALL_CONF, on page 141	Confirms successful completion of the ALTERNATE_CALL_REQ message.
ANSWER_CALL_REQ, on page 141	Allows a client to connect to an alerting call at the device that is alerting.
ANSWER_CALL_CONF, on page 142	Confirms the successful completion of the ANSWER_CALL_REQ message.
CLEAR_CALL_REQ, on page 142	Allows a client to release all devices from the specified call without regard to the number of other call parties.
CLEAR_CALL_CONF, on page 143	Confirms the successful completion of the CLEAR_CALL_REQ message.
CLEAR_CONNECTION_REQ, on page 143	Allows a client to release a specific device connection from the designated call.
CLEAR_CONNECTION_CONF, on page 144	Confirms the successful completion of the CLEAR_CONNECTION_REQ message.
CONFERENCE_CALL_REQ, on page 144	Allows a client to conference an existing held call with another active call.
CONFERENCE_CALL_CONF, on page 146	Confirms the successful completion of the CONFERENCE_CALL_REQ message.
CONSULT_CALL_REQ, on page 148	Allows the client to request the combined action of placing an active call on hold and then making a new call.
CONSULT_CALL_CONF, on page 150	Confirms the successful completion of the CONSULT_CALL_REQ message.
HOLD_CALL_REQ, on page 150	Allows a client to put an existing call connection into the held state.
HOLD_CALL_CONF, on page 151	Confirms the successful completion of the HOLD_CALL_REQ message.
MAKE_CALL_REQ, on page 151	Allows a client to initiate a call between two devices.
MAKE_CALL_CONF, on page 153	Confirms the successful completion of the MAKE_CALL_REQ message.
RETRIEVE_CALL_REQ, on page 155	Allows a client to retrieve an existing held call.

Message	Purpose
RETRIEVE_CALL_CONF, on page 156	Confirms the successful completion of the RETRIEVE_CALL_REQ message.
TRANSFER_CALL_REQ, on page 156	Allows a client to transfer a held call with an active call at the same device.
TRANSFER_CALL_CONF, on page 158	Confirms the successful completion of the TRANSFER_CALL_REQ message.
SNAPSHOT_CALL_REQ, on page 177	Allows a client to retrieve information on a specified call.
SNAPSHOT_CALL_CONF, on page 178	Confirms the successful completion of the SNAPSHOT_CALL_REQ message.
SNAPSHOT_DEVICE_REQ, on page 181	Allows a client to retrieve information on a specified device.
SNAPSHOT_DEVICE_CONF, on page 181	Confirms the successful completion of the SNAPSHOT_DEVICE_REQ message.
SEND_DTMF_SIGNAL_REQ, on page 159	Allows a client to have the server transmit a sequence of DTMF tones on behalf of a call party.
SEND_DTMF_SIGNAL_CONF, on page 160	Confirms the successful completion of the SEND_DTMF_SIGNAL_REQ message.
BAD_CALL_REQ, on page 167	Allows a client to notify the server of the bad quality of a call.
BAD_CALL_CONF, on page 168	Confirms the successful completion of the BAD_CALL_REQ message.



CHAPTER 8

Using the Outbound Feature

This chapter includes the following topics.

- About the Outbound Feature, page 49
- Outbound Expanded Call Context (ECC) Variables, page 49
- Outbound Call Events, page 50
- The BAResponse ECC Variable, page 51

About the Outbound Feature

CRS 5.0(1) includes an outbound feature. The outbound feature allows agents in a CSQ to handle outbound campaigns. When an agent handles an outbound call, the agent is presented with contact information about a customer. The agent has several options regarding this customer contact information. If the agent decides to call the customer, a call is made from the agent phone to the customer. For details about the outbound feature please see the Cisco Unified Contact Center Express Solutions Administration Guide.

Unified CCX CTI server supports this feature using the CTI protocol messages. Agents with outbound capability should specify this capability in the AgentCapacity field of the SET_AGENT_STATE_REQ message. When an agent with the outbound capability is logged in and the agent is part of an outbound campaign, the agent receives a special sequence of messages to handle an outbound call.

Outbound Expanded Call Context (ECC) Variables

Unified CCX uses pre-defined ECC variables to exchange data with CTI clients for the outbound feature. See the following table for a complete list of the pre-defined ECC variables that are used for the outbound feature.

Note These ECC variables should not be used by the CTI client for other purposes. They are reserved for the outbound feature.

Table 8: Outbound Expanded Call Context (ECC) Variables

ECC variable	Description
BACampaign	Indicates the name of the Outbound Dialer campaign to which the call belongs.
BAAccountNumber	Identifies a customer account number and can be used by CTI clients to perform a database lookup to obtain additional customer data. This ECC variable displays only if the data was available in the customer import file. **Note** The maximum character length of this ECC variable is 30 characters.
BAResponse	Multi-purpose placeholder that sends data from CTI client to the Outbound Dialer. This variable is used when the CTI client responds to the server's agent reservation request (for example: Accept, Reject, Skip, and so on). It is also used to schedule and cancel callbacks and make changes to the callback number.
BAStatus	Contains two characters indicating the mode and direction of the Outbound Dialer initiated call: • The first character identifies the call mode ("D" = Direct Preview reservation for Unified CCX, "C" = Connected to Customer, "Z" = Transferred/Conferenced) • The second character identifies the direction (always "O" = Outbound for Unified CCX). So a BAStatus of DO would indicate a Direct Preview Reservation call and a BAStatus of CO would indicate that this is an Outbound call that is connected to a customer. If an outbound call is transferred or conferenced, the BAStatus is set to ZO.
BADialedListID	A unique key identifying a specific customer record.
BATimeZone	Indicates the GMT offset, in minutes, for the customer's time zone and obtains the customer's local time.
BABuddyName	Contains the customer's first and last name separated by a comma, if provided in the contacts list imported for the campaign.
BACustomerNumber	Contains dialed customer phone number

Outbound Call Events

When Unified CCX needs to request an agent to make an outbound call, Unified CCX initiates a sequence of call events that is similar to those for making a call.

1 Before an outbound call is placed, an available agent is reserved.

2 Unified CCX sends two call events to the agent: BEGIN_CALL_EVENT and CALL_DELIVERED_EVENT.

- Although the call IDs used in these events are real, these call events are created by Unified CCX artificially. There is no actual phone call being made by any device. The call represented by these artificial call events is called a reservation call.
- The BAStatus ECC variable is set to “DO” for these events. This indicates that the call is an outbound call.

3 When a CTI client receives these events, it should inform the agent about the customer contact information in the outbound ECC variables. It does this through the BAResponse ECC variable.

At this point, the CTI client should send a SET_CALL_DATA_REQ message to Unified CCX with the BAResponse ECC variable set to one of the values in Table 9: The BAResponse ECC Variable Values, on page 51.

If the BAResponse value is Accept, Reject, Reject-Close, or Cancel Reservation, Unified CCX sends a CALL_CONNECTION_CLEARED_EVENT message to the CTI client with the reservation call ID.

4 If the customer contact is accepted by the agent, Unified CCX makes a call from the agent device to the customer phone. This call is termed the preview call. For preview calls, the BAStatus is set to CO.

Call Classification

For preview calls, a connected customer call is classified as Voice in Unified CCX by default. The agent has the option to change the call classification after accepting and while connected to the customer or when in the Work state.

The CTI client may change the call classification by sending the SET_CALL_DATA_REQ message to Unified CCX with the BAResponse ECC variable set to one of values in Table 9: The BAResponse ECC Variable Values, on page 51. In addition to reclassifying the call, the CTI client may also set the BAResponse ECC variable to schedule a customer callback, terminate the current call, and call the next available customer phone.

The BAResponse ECC Variable

Table 9: The BAResponse ECC Variable Values, on page 51 lists the values you can use with the BAResponse ECC variable.

Table 9: The BAResponse ECC Variable Values

Variable	Description
Values used to respond to the reservation call	
Accept	To accept the current customer contact. This will initiate the outbound call to the customer from the agent’s phone. Once the agent responds with Accept, the call becomes a preview call.
Reject	To reject the current customer contact. This will cancel the agent reservation and change her state to Ready. She can now handle either inbound or outbound calls.

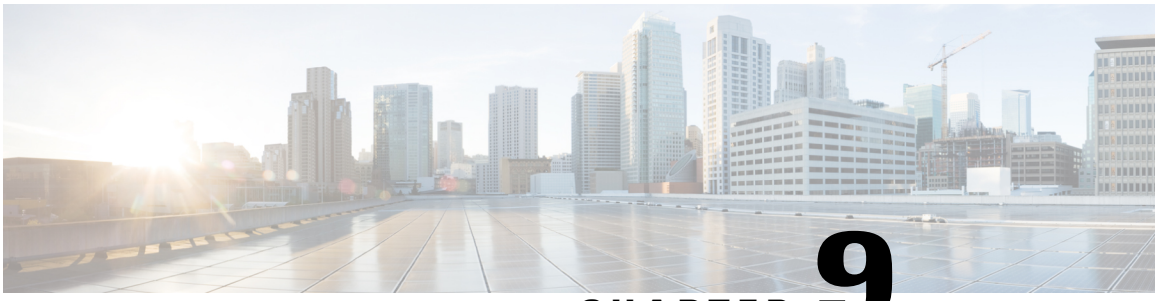
Variable	Description
Reject-Close	To reject the current customer contact and close the record so it will not be called again for this particular campaign. This will cancel the agent reservation and change her state to Ready. She can now handle either inbound or outbound calls.
Skip	To skip the current customer contact. The agent remains reserved to handle another outbound contact.
Skip-Close	To skip the current preview call and close the record so it will not be called again for this particular campaign. The agent remains reserved to handle another outbound contact.
Cancel Reservation	To cancel the agent reservation and to set the agent to Not Ready state. The record remains open in the database. Clicking Cancel Reservation has a similar effect to clicking Reject except that the agent goes to Not Ready instead of Ready.
Values used during a preview call	
Reclassify	Indicates the preview call is reclassified.
REX_VOICE	Indicates the preview call is reclassified as VOICE.
REX_ANS_MACHINE	Indicates the preview call is reclassified as ANSWERING MACHINE.
REX_FAX	Indicates the preview call is reclassified as FAX.
REX_INVALID	Indicates the preview call is reclassified as INVALID PHONE NUMBER.
DO_NOT_CALL	Indicates the preview call is reclassified as a phone number to be added on the DO NOT CALL list.
BUSY	Indicates the preview call is reclassified as BUSY.
Callback mmddyyyy hh:mm	Indicates the customer wants a call back at the specified time
P#<phone number> For example:P#5551212	Indicates the phone number that the customer wants to be called back with. This value may be sent to Unified CCX after the Callback is sent.
Callback Cancel	Indicates the customer wants to cancel a previous callback request.
SkipNext	Indicates an agent request to call the next available customer phone number.
SKIP WRONG_NUMBER	Informs the agent that the number called is a wrong number. After the call terminates, the system calls the next phone number for this customer.

Variable	Description
SKIP NOT_HOME	Informs the agent that the customer is not at home. After the call terminates, the system calls the next phone number for this customer.

PART II

Reference Section

- All Message Types Organized by Class and Type, page 57
- Data Types and Message Constants, page 183



CHAPTER 9

All Message Types Organized by Class and Type

Table 10: Message Types Listed by Message Class and by Message TypeName, on page 58 lists the message types by message class and by message type name. See the index for an alphabetical listing.

A “Solicit” message is a client message that makes a request of the server (Unified CCX). A “Solicited” message is a message Unified CCX sends to the client in response to a “solicit” message. For example: an OPEN_REQ and an OPEN_CONF message are a solicit and a solicited message respectively. An “unsolicited” message is one sent to the client that the client did not request; For example a SYSTEM_EVENT message.

Table 10: Message Types Listed by Message Class and by Message TypeName

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
Session messages	OPEN_REQ, on page 73	3	Client	Yes	Requests the start of a communications session with Unified CCX.
	OPEN_CONF, on page 75	4	Unified CCX	Yes	Confirms to the client the starting of a session.
	HEARTBEAT_REQ, on page 77	5	Client	Yes	Sends a HEARTBEAT_REQ message to Unified CCX whenever no messages have been sent for the heartbeat interval or sooner for the purpose of detecting and handling transmission failures.
	HEARTBEAT_CONF, on page 77	6	Unified CCX	Yes	Confirms that the HEARTBEAT_REQ message has been received.
	SYSTEM_EVENT, on page 77	31	Unified CCX	No	Indicates the Unified CCX server status or the agent's device status changes.
	CLOSE_REQ, on page 79	7	Client	Yes	Requests the termination of a session from a client.
	CLOSE_CONF, on page 79	8	Unified CCX	Yes	Confirms the termination of a session.
Error messages	FAILURE_CONF, on page 83	1	Unified CCX	Yes	Confirms a failure to process a client's request
	FAILURE_EVENT, on page 84	2	Unified CCX	No	Notifies the client, without it having been requested, of a failure or an error.

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
Configuration messages	CONFIG_REQUEST_KEY_EVENT, on page 85	230	Client	Yes	Requests the current configuration keys for various types of configuration data.
	CONFIG_KEY_EVENT, on page 85	231	Unified CCX	Yes	Provides the client with the requested configuration keys at the time of the client's key request.
	CONFIG_REQUEST_EVENT, on page 86	232	Client	Yes	Requests a complete set of configuration data.
	CONFIG_BEGIN_EVENT, on page 87	233	Unified CCX	Yes (for initial upload) No (for updates)	Signals the beginning of a consistent set of configuration data.
	CONFIG_END_EVENT, on page 88	234	Unified CCX	Yes (for initial upload) No (for updates)	Signals the end of a set of consistent configuration data.
	CONFIG_APPLICATION_EVENT, on page 90	235	Unified CCX	Yes (for initial upload) No (for updates)	Contains application configuration data.
	CONFIG_CSQ_EVENT, on page 89	236	Unified CCX	Yes (for initial upload) No (for updates)	Contains CSQ configuration data.
	CONFIG_AGENT_EVENT, on page 92	237	Unified CCX	Yes (for initial upload) No (for updates)	Contains agent configuration data.
	CONFIG_DEVICE_EVENT, on page 93	238	Unified CCX	Yes (for initial upload) No (for updates)	Contains route point, agent extension, CTI port, or call control group configuration data
	TEAM_CONFIG_REQ, on page 96	242	Client	Yes	Requests agent team configuration data.

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
	TEAM_CONFIG_EVENT, on page 95	243	Unified CCX	Yes (for initial upload) No (for updates)	Contains team configuration data.
	TEAM_CONFIG_CONF, on page 97	244	Unified CCX	Yes	Confirms the end of initial agent team configuration data.
Agent-State Messages	AGENT_STATE_EVENT, on page 98	30	Unified CCX	No	Notifies clients that an agent state has changed.
	QUERY_AGENT_STATE_REQ, on page 100	36	Client	Yes	Queries the current state of an agent.
	QUERY_AGENT_STATE _CONF, on page 101	37	Unified CCX	Yes	Provides the client with the agent state data that was requested.
	SET_AGENT_STATE_REQ, on page 102	38	Client	Yes	Requests a change in the current state of an agent.
	SET_AGENT_STATE_CONF, on page 103	39	Unified CCX	Yes	Confirms that Unified CCX has successfully processed the SET_AGENT_STATE_REQ message.

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
Call-Event Messages	BEGIN_CALL_EVENT, on page 105	23	Unified CCX	No	Notifies the client of a new call and provides the initial call context data.
	END_CALL_EVENT, on page 107	24	Unified CCX	No	Notifies the client that the association between a call and its Unified CCX client is dissolved.
	CALL_DATA_UPDATE_EVENT, on page 108	25	Unified CCX	No	Notifies the client of changes to a call's context data and contains only the data that has changed.
	CALL_DELIVERED_EVENT, on page 111	9	Unified CCX	No	Notifies the client that a call has arrived at a device. The call is delivered when the phone rings.
	CALL_ESTABLISHED_EVENT, on page 114	10	Unified CCX	No	Notifies the client that a call has been answered and connected at a device.
	CALL_HELD_EVENT, on page 116	11	Unified CCX	No	Notifies the client that a call has been placed on hold.
	CALL_RETRIEVED_EVENT, on page 117	12	Unified CCX	No	Notifies the client that a call previously placed on hold has been resumed.
	CALL_CLEARED_EVENT, on page 118	13	Unified CCX	No	Notifies the client that a call is terminated, normally when the last device disconnects from the call.
	CALL_CONNECTION_CLEARED_EVENT, on page 119	14	Unified CCX	No	Notifies the client that a party dropped from a call.
	CALL_ORIGINATED_EVENT, on page 120	15	Unified CCX	No	Notifies the client that a call is initiated at a monitored device.
	CALL_FAILED_EVENT, on page 121	16	Unified CCX	No	Notifies the client that a call encountered an error.
	CALL_CONFERENCED_EVENT, on page 123	17	Unified CCX	No	Notifies the client that a call has been joined with other(s) into a conference call.
	CALL_TRANSFERRED_EVENT, on page 125	18	Unified CCX	No	Notifies the client that a call has been transferred.

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
	CALL_DIVERTED_EVENT, on page 128	19	Unified CCX	No	Notifies the client that a call has been moved from one device to another.
	CALL_SERVICE_INITIATED_EVENT, on page 129	20	Unified CCX	No	Notifies the client of an initiation of telecommunications service (dial tone) at a device.
	CALL_QUEUED_EVENT, on page 131	21	Unified CCX	No	Notifies the client that a call has been placed in a queue pending the availability of an agent.
	CALL_DEQUEUED_EVENT, on page 133	86	Unified CCX	No	Notifies the client that a call has been explicitly removed from a queue.
	RTP_STARTED_EVENT (OPTIONAL), on page 134	116	Unified CCX	No	Notifies the client that an RTP (Real-Time Protocol) voice media stream has been started.
	RTP_STOPPED_EVENT (OPTIONAL), on page 136	117	Unified CCX	No	Notifies the client that an RTP media stream has been stopped.

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
Client-Control (Call-Control) messages	CONTROL_FAILURE_CONF, on page 139	35	Unified CCX	Yes	Confirms that the previously requested control-service function identified by the given invokeID was unsuccessful.
	ALTERNATE_CALL_REQ, on page 139	40	Client	Yes	Allows a client to request the combined action of placing an active call on hold and then either retrieving a previously held call or answering an alerting call at the same device.
	ALTERNATE_CALL_CONF, on page 141	41	Unified CCX	Yes	Confirms the processing completion of the ALTERNATE_CALL_REQ message.
	ANSWER_CALL_REQ, on page 141	42	Client	Yes	Allows a client to connect to an alerting call at the device which is alerting.
	ANSWER_CALL_CONF, on page 142	43	Unified CCX	Yes	Confirms the processing completion of the ANSWER_CALL_REQ message.
	CLEAR_CALL_REQ, on page 142	44	Client	Yes	Allows a client to release all devices from the specified call.
	CLEAR_CALL_CONF, on page 143	45	Unified CCX	Yes	Confirms the processing completion of the CLEAR_CALL_REQ message.
	CLEAR_CONNECTION_REQ, on page 143	46	Client	Yes	Allows a client to release a specific device connection from the specified call.
	CLEAR_CONNECTION_CONF, on page 144	47	Unified CCX	Yes	Confirms the processing completion of the CLEAR_CONNECTION_REQ message.
	CONFERENCE_CALL_REQ, on page 144	48	Client	Yes	Allows a client to complete a conference call by combining the consult call with the original call
	CONFERENCE_CALL_CONF, on page 146	49	Unified CCX	Yes	

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
					Confirms the processing completion of the CONFERENCE_CALL_REQ message.
	CONSULT_CALL_REQ, on page 148	50	Client	Yes	Allows the client to request the combined action of placing an active call on hold and then making a new call in order to make a conference call or to transfer the call.
	CONSULT_CALL_CONF, on page 150	51	Unified CCX	Yes	Confirms the processing completion of the CONSULT_CALL_REQ message.
	HOLD_CALL_REQ, on page 150	54	Client	Yes	Allows a client to place an existing call connection into the held state.
	HOLD_CALL_CONF, on page 151	55	Unified CCX	Yes	Confirms the processing completion of the HOLD_CALL_REQ message.
	MAKE_CALL_REQ, on page 151	56	Client	Yes	Allows a client to initiate a call.
	MAKE_CALL_CONF, on page 153	57	Unified CCX	Yes	Confirms the processing completion of the MAKE_CALL_REQ message.
	RECONNECT_CALL_REQ, on page 154	60	Client	Yes	Requests the combined action of clearing an active call and then retrieving an existing held call.
	RECONNECT_CALL_CONF, on page 155	61	Unified CCX	Yes	Confirms the processing completion of the RECONNECT_CALL_REQ message.
	RETRIEVE_CALL_REQ, on page 155	62	Client	Yes	Allows a client to retrieve an existing held call.
	RETRIEVE_CALL_CONF, on page 156	63	Unified CCX	Yes	Confirms the processing completion of the RETRIEVE_CALL_REQ message.

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
	TRANSFER_CALL_REQ, on page 156	64	Client	Yes	Allows a client to complete a transfer that was initiated by a consult.
	TRANSFER_CALL_CONF, on page 158	65	Unified CCX	Yes	Confirms the processing completion of the TRANSFER_CALL_REQ message.
	SEND_DTMF_SIGNAL_REQ, on page 159	91	Client	Yes	Allows a client to have the Unified CCX server transmit a sequence of DTMF tones on behalf of a call party.
	SEND_DTMF_SIGNAL_CONF, on page 160	92	Unified CCX	Yes	Confirms the processing completion of the SEND_DTMF_SIGNAL_REQ message.
	SET_CALL_DATA_REQ, on page 160	26	Client or Unified CCX	Yes	Requests the update of one or more call data.
	SET_CALL_DATA_CONF, on page 162	27	Unified CCX	Yes	Confirms the processing completion of a previous SET_CALL_DATA_REQ message.
	SUPERVISE_CALL_REQ	124	Client	Yes	Requests (from a supervisor) a monitoring or barge-in operation.
	SUPERVISE_CALL_CONF, on page 164	125	Unified CCX	Yes	Confirms the processing completion of a previous SUPERVISOR_CALL_REQ message.
	SUPERVISOR_ASSIST_REQ, on page 165	118	Client	Yes	Requests (from an agent) supervisor assistance.
	SUPERVISOR_ASSIST_CONF, on page 166	119	Unified CCX	Yes	Confirms the processing completion of a SUPERVISOR_ASSIST_REQ message.
	SUPERVISOR_ASSIST_EVENT, on page 167	120	Unified CCX	Yes	Notifies the client that a supervisor assist request was sent by a Unified CCX client.
	BAD_CALL_REQ, on page 167	139	Client	Yes	

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
					Allows a client to notify Unified CCX of the bad quality of a call.
	BAD_CALL_CONF, on page 168	140	Unified CCX	Yes	Confirms the processing completion of the BAD_CALL_REQ message.

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
Miscellaneous messages	QUERY_QUEUE_STATISTICS_REQ, on page 169	223	Client	Yes	Requests the current CSQ call handling statistics.
	QUERY_QUEUE_STATISTICS_CONF, on page 169	224	Unified CCX	Yes	Confirms the processing completion of a previous QUERY_AGENT_QUEUE_STATISTICS_REQ message.
	QUERY_AGENT_QUEUE_STATISTICS_REQ, on page 172	239	Client	Yes	Requests the current agent call handling statistics.
	QUERY_AGENT_QUEUE_STATISTICS_CONF, on page 172	240	Unified CCX	Yes	Confirms the processing completion of a QUERY_AGENT_QUEUE_STATISTICS_CONF message.
	QUERY_DEVICE_INFO_REQ, on page 173	78	Client	Yes	Allows a client to retrieve general information about a specified device.
	QUERY_DEVICE_INFO_CONF, on page 174	79	Unified CCX	Yes	Confirms the processing completion of a QUERY_DEVICE _INFO_REQ message.
	QUERY_SUMMARY_STATISTICS_REQ, on page 175	225	Client	Yes	Requests system summary statistics for Unified CCX. To avoid impacting system performance, clients should not request statistics too frequently.
	QUERY_SUMMARY_STATISTICS_CONF, on page 175	226	Unified CCX	Yes	Confirms the processing completion of a QUERY_SUMMARY_STATICS_REQ message.
	SNAPSHOT_CALL_REQ, on page 177	82	Client	Yes	Allows a client to retrieve information on a specified call.
	SNAPSHOT_CALL_CONF, on page 178	83	Unified CCX	Yes	Confirms the processing completion of a SNAPSHOT_CALL_REQ message.
	SNAPSHOT_DEVICE_REQ, on page 181	84	Client	Yes	Allows a client to retrieve information on a specified device.
		85	Unified CCX	Yes	

Message class	MessageTypeName	Type ID #	Sent by	Solicit or Solicited	Purpose
	SNAPSHOT_DEVICE_CONF, on page 181				Confirms the processing completion of a SNAPSHOT_DEVICE_REQ message.

- All Message Types Organized by ID Number, page 68
- Session-Management Messages, page 73
- Configuration Messages, page 84
- Agent-State Messages, page 97
- Call-Event Messages, page 104
- Call-Control (Client-Control) Messages, page 137
- Miscellaneous Messages, page 168

All Message Types Organized by ID Number

Table 11: Message Types Numerically Listed by ID Number , on page 68 lists the message types by ID number. See the index for an alphabetical listing.

Table 11: Message Types Numerically Listed by ID Number

MessageTypeID #	MessageTypeName
1	FAILURE_CONF
2	FAILURE_EVENT
3	OPEN_REQ
4	OPEN_CONF
5	HEARTBEAT_REQ
6	HEARTBEAT_CONF
7	CLOSE_REQ
8	CLOSE_CONF
9	CALL_DELIVERED_EVENT
10	CALL_ESTABLISHED_EVENT

MessageTypeID #	MessageTypeName
11	CALL_HELD_EVENT
12	CALL_RETRIEVED_EVENT
13	CALL_CLEARED_EVENT
14	CALL_CONNECTION_CLEARED_EVENT
15	CALL_ORIGINATED_EVENT
16	CALL_FAILED_EVENT
17	CALL_CONFERENCED_EVENT
18	CALL_TRANSFERRED_EVENT
19	CALL_DIVERTED_EVENT
20	CALL_SERVICE_INITIATED_EVENT
21	CALL_QUEUED_EVENT
23	BEGIN_CALL_EVENT
24	END_CALL_EVENT
25	CALL_DATA_UPDATE_EVENT
26	SET_CALL_DATA_REQ
27	SET_CALL_DATA_CONF
30	AGENT_STATE_EVENT
31	SYSTEM_EVENT
35	CONTROL_FAILURE_CONF
36	QUERY_AGENT_STATE_REQ
37	QUERY_AGENT_STATE _CONF
38	SET_AGENT_STATE_REQ
39	SET_AGENT_STATE_CONF
40	ALTERNATE_CALL_REQ

MessageTypeID #	MessageTypeName
41	ALTERNATE_CALL_CONF
42	ANSWER_CALL_REQ
43	ANSWER_CALL_CONF
44	CLEAR_CALL_REQ
45	CLEAR_CALL_CONF
46	CLEAR_CONNECTION_REQ
47	CLEAR_CONNECTION_CONF
48	CONFERENCE_CALL_REQ
49	CONFERENCE_CALL_CONF
50	CONSULT_CALL_REQ
51	CONSULT_CALL_CONF
54	HOLD_CALL_REQ
55	HOLD_CALL_CONF
56	MAKE_CALL_REQ
57	MAKE_CALL_CONF
60	RECONNECT_CALL_REQ
61	RECONNECT_CALL_CONF
62	RETRIEVE_CALL_REQ
63	RETRIEVE_CALL_CONF
64	TRANSFER_CALL_REQ
65	TRANSFER_CALL_CONF
68	reserved
69	reserved
78	QUERY_DEVICE_INFO_REQ

MessageTypeID #	MessageTypeName
79	QUERY_DEVICE_INFO_CONF
82	SNAPSHOT_CALL_REQ
83	SNAPSHOT_CALL_CONF
84	SNAPSHOT_DEVICE_REQ
85	SNAPSHOT_DEVICE_CONF
86	CALL_DEQUEUED_EVENT
91	SEND_DTMF_SIGNAL_REQ
92	SEND_DTMF_SIGNAL_CONF
112	reserved
113	reserved
114	reserved
115	reserved
116	RTP_STARTED_EVENT (OPTIONAL)
117	RTP_STOPPED_EVENT (OPTIONAL)
118	SUPERVISOR_ASSIST_REQ
119	SUPERVISOR_ASSIST_CONF
120	SUPERVISOR_ASSIST_EVENT
121	EMERGENCY_CALL_REQ
122	EMERGENCY_CALL_CONF
123	EMERGENCY_CALL_EVENT
124	SUPERVISE_CALL_REQ
125	SUPERVISE_CALL_CONF
126	reserved
127	reserved

MessageTypeID #	MessageTypeName
128	reserved
134	reserved
139	BAD_CALL_REQ
140	BAD_CALL_CONF
211	reserved
212	reserved
213	reserved
214	reserved
215	reserved
223	QUERY_QUEUE_STATISTICS_REQ
224	QUERY_QUEUE_STATISTICS_CONF
225	QUERY_SUMMARY_STATISTICS_REQ
226	QUERY_SUMMARY_STATISTICS_CONF
230	CONFIG_REQUEST_KEY_EVENT
231	CONFIG_KEY_EVENT
232	CONFIG_REQUEST_EVENT
233	CONFIG_BEGIN_EVENT
234	CONFIG_END_EVENT
235	CONFIG_APPLICATION_EVENT
236	CONFIG_CSQ_EVENT
237	CONFIG_AGENT_EVENT
238	CONFIG_DEVICE_EVENT
239	QUERY_AGENT_QUEUE_STATISTICS_REQ
240	QUERY_AGENT_QUEUE_STATISTICS_CONF

MessageTypeID #	MessageTypeName
242	TEAM_CONFIG_REQ
243	TEAM_CONFIG_EVENT
244	TEAM_CONFIG_CONF

Session-Management Messages

This section covers the following topics:

- Session Initialization Maintenance System Event and Termination Messages, on page 73
- Masks Used in the OPEN_REQ Message, on page 79
- Failure Messages, on page 83

Session Initialization Maintenance System Event and Termination Messages

This section includes the following messages:

- OPEN_REQ, on page 73
- OPEN_CONF, on page 75
- HEARTBEAT_REQ, on page 77
- HEARTBEAT_CONF, on page 77
- SYSTEM_EVENT, on page 77
- CLOSE_REQ, on page 79
- CLOSE_CONF, on page 79

OPEN_REQ

The OPEN_REQ and the OPEN_CONF, on page 75 messages depict the session initialization message flow. Once a TCP connection has been established, a client attempts to initialize a communications session by sending to Unified CCX an OPEN_REQ message, defined in this section.

Note The ServiceRequested mask determines if a client is in bridge mode or in agent mode. These two modes are mutually exclusive. Do not set agent-mode fields for a bridge-mode client. See Two Client Modes for Connecting with Unified CCX, on page 19, for an explanation of agent and bridge modes.

The square bracketed subscript number ending the field names for the data in the floating part of the message descriptions is the FieldDataID for that field. For example $AgentID_{[194]}$ means that 194 is the FieldDataID for the AgentID field.

Table 12: OPEN_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
VersionNumber	The version number of the interface requested by the client. This defines the version of all messages in the message set.	UINT	4
IdleTimeout	The session idle timer, expressed in seconds. If the session is idle (no messages received) for this length of time, Unified CCX resets the TCP connection and awaits the establishment of a new session. This value must be at least 2 times the heartbeat interval but less than 120 seconds.	UINT	4
reserved	Set this field to zero.	UINT	4
ServicesRequested	A bitwise combination of the CTI Services listed in Table 22: CTI Service Masks , on page 80 that the client is to receive.	UINT	4
CallMsgMask	A bitwise combination of the Unsolicited Call Event Message Masks listed in Table 23: Unsolicited Call Event Message Masks , on page 80 that the client is to receive.	UINT	4
AgentStateMask	A bitwise combination of Agent State Masks listed in Table 24: Agent State Masks , on page 82 that the client is to receive.	UINT	4
ConfigMsgMask	A bitwise combination of the Configuration Event masks listed in Table 25: Configuration Information Masks , on page 83 that the client is to receive.	UINT	4
reserved	Set this field to zero.	UINT	4
reserved	Set this field to zero.	UINT	4
reserved	Set this field to zero.	UINT	4

Table 13: OPEN_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ClientID (required)[1]	The user ID of the client.	STRING	64
ClientPassword[2] (required)	The password of the user identified by clientID. ClientID and ClientPassword are optionally used to authenticate the client making the session open request. This field must be present even if authentication is not being used (they can be of zero length).	UNSPEC	64
ClientSignature[3] (optional)	A character string that can be used to identify a client.	STRING	64
AgentExtension[4] (required for Agent mode)	The agent's ACD IP phone extension. For Agent mode, at least one of AgentExtension, AgentID, or AgentInstrument must be provided by the client.	STRING	16
AgentInstrument[6] (required for Agent mode)	The agent's IP phone number. For Agent mode, at least one of AgentExtension, AgentID, or AgentInstrument must be provided by the client.	STRING	64
AgentID[194] (required for Agent mode)	The agent's Unified CCX login.	STRING	32

[1] **Note** Unified CCX does not use the Client ID and Client Password (they can be of zero length). Even though the fields are required, Unified CCX Computer Telephony Integration ignores them.

OPEN_CONF

The OPEN_CONF message, defined in the following tables, confirms the completion of the processing requested by the OPEN_REQ message.

Table 14: OPEN_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Maximum size
InvokeID	The InvokeID from the corresponding OPEN_REQ message.	UINT	4
ServicesGranted	A bitwise combination of the CTI Services listed in CTI Service Masks, on page 80 that the client has been granted. Services granted can be less than those requested.	UINT	4
reserved	Zero.	UINT	4

Fixed part Field name	Value	Data type	Maximum size
reserved	Zero.	UINT	4
reserved	Zero.	TIME	4
Unified CCX Online	The current Unified CCX on-line status when client EVENTS service has been granted. 1: online 0: offline	BOOL	2
reserved	Zero.	USHORT	2
AgentState	One of the values from What is an Agent State?, on page 31 representing the current state of the associated agent phone. This field is required for Agent mode.	USHORT	2

Table 15: OPEN_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
AgentExtension[4]	The agent's IP phone extension, when client EVENTS service has been granted and the agent is currently logged into Unified CCX. This field is required for Agent mode.	STRING	16
reserved[5]	Ignore this value.	STRING	12
AgentInstrument[6]	The agent's IP phone number, when client EVENTS service has been granted and the agent is currently logged into Unified CCX. This field is required for Agent mode.	STRING	64
AgentID[194]	The agent's Unified CCX login. This field is required for Agent mode.	STRING	32
NumPeripherals[228] (Version 14 and later)	The number of FltPeripheralID and MultilineAgentControl pairs specified in the floating portion of the message. For Unified CCX, this is always 1.	USHORT	2
FltPeripheralID[208] (Version 14 and later)	The Peripheral ID for the MultilineAgentControl field. For Unified CCX, this is the peripheral ID given by the OPEN_REQ.	UINT	4

Floating part Field name	Value	Data type	Maximum size
MultilineAgentControl[224] (Version 14 and later)	Specifies if multi-line agent control is available on the above peripheral. For Unified CCX, this is always 1.	USHORT	2

HEARTBEAT_REQ

The HEARTBEAT_REQ and the HEARTBEAT_CONF, on page 77 messages depict the heartbeat message flow. The following table defines the HEARTBEAT_REQ message.

Table 16: HEARTBEAT_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4

HEARTBEAT_CONF

On receipt of a HEARTBEAT_REQ message, Unified CCX immediately responds with a HEARTBEAT_CONF message, defined in the following table.

Table 17: HEARTBEAT_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	The InvokeID from the corresponding HEARTBEAT_REQ message.	UINT	4

SYSTEM_EVENT

When the Unified CCX status or the agent's device status changes, Unified CCX sends a SYSTEM_EVENT message, defined in the following table, to all clients to indicate that status (for example, on_line or off_line).

Table 18: SYSTEM_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
CCXStatus	The current operational status of Unified CCX. Any non-zero value is indicative of a component failure or communication outage that prevents normal CTI operations (see also Unified CCX Status Values, on page 208).	UINT	4
CCXTime	The current Unified CCX date and time.	TIME	4
SystemEventID	A value that enumerates the specific system event that occurred. See SystemEventID Values, on page 195 for system event ID values.	UINT	4
SystemEventArg1	An argument value that is specific to the system event being reported.	UINT	4
SystemEventArg2	A second argument value that is specific to the system event being reported.	UINT	4
SystemEventArg3	A third argument value that is specific to the system event being reported.	UINT	4
EventDeviceType	Indicates the type of the device ID supplied in the EventDeviceID floating field. See DeviceType Values, on page 192 for device ID type values. This is set to DEVID_NONE if no floating field is provided.	USHORT	2

Table 19: SYSTEM_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
Text[7] (optional)	A text message associated with the provided SystemEventID.	STRING	255
EventDeviceID[206]	A text value of the device ID if reported. Initially only used by Unified CCX for a SYS_DEVICE_IN_SERVICE, and a SYS_DEVICE_OUT_OF_SERVICE message.	STRING	64

CLOSE_REQ

The CLOSE_REQ message, defined in the following table, requests from Unified CCX a communication session termination.

The CLOSE_REQ and CLOSE_CONF messages depict the session termination message flow.

Table 20: CLOSE_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
Status	A status code indicating the reason for closing the session. See Table 201: Error Status Codes , on page 209.	UINT	4

CLOSE_CONF

Unified CCX confirms the termination of the communication session with the CLOSE_CONF message, defined in the following table.

Table 21: CLOSE_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding `CLOSE_REQ` message.	UINT	4

Masks Used in the OPEN_REQ Message

Unified CCX can provide much more real-time data than the typical client needs. Apply message masks to suppress (mask) the transmission of unneeded data and thereby avoid wasting network bandwidth.

The network impact of the expected number of simultaneously connected clients must be carefully considered before deploying a client application that unmasks a large number of messages.

Within the OPEN_REQ message, there are four separate mask types for selecting the type of messages wanted and filtering out (masking) unwanted messages:

- CTI Service Masks, on page 80
- Unsolicited Call-Event Message Masks, on page 80
- Agent State Masks, on page 82
- Configuration-Information Masks, on page 83

CTI Service Masks

CTI service masks specify the CTI services that the client requests.

Table 22: CTI Service Masks

Mask name	Description	Bytes
CTI_SERVICE_CLIENT_EVENTS	Client receives call and agent state change events associated with a specific phone. This bit is used to indicate the client is in agent mode. This bit cannot be set if CTI_SERVICE_ALL_EVENT is set.	0x00000001
CTI_SERVICE_CALL_DATA_UPDATE	Client can modify call context data.	0x00000002
CTI_SERVICE_CLIENT_CONTROL	Client can control calls and agent states associated with a specific phone.	0x00000004
CTI_SERVICE_ALL_EVENTS	Client receives all call and agent-state change events (associated with any phone). This is the bit to indicate that the client is in bridge mode. This bit cannot be set when CTI_SERVICE_CLIENT_EVENTS is set.	0x00000010
CTI_SERVICE_SUPERVISOR	Client can perform agent supervisory functions.	0x00000080
CTI_AGENT_STATE_CONTROL_ONLY (Version 12)	Client can perform agent state control functions.	0x00002000
CTI_SERVICE_CONFIG_EVENTS	Requests that this client receive configuration events.	0x00040000

Unsolicited Call-Event Message Masks

The Unsolicited Call-Event Message masks specify unsolicited call-event messages that the client requests.

Table 23: Unsolicited Call Event Message Masks

Mask name	Description	Value
CALL_DELIVERED_MASK	Set when client wishes to receive CALL_DELIVERED_EVENT messages.	0x00000001
CALL_QUEUED_MASK	Set when client wishes to receive CALL_QUEUED_EVENT messages.	0x00000002

Mask name	Description	Value
CALL_ESTABLISHED_MASK	Set when client wishes to receive CALL_ESTABLISHED_EVENT messages.	0x00000004
CALL_HELD_MASK	Set when client wishes to receive CALL_HELD_EVENT messages.	0x00000008
CALL_RETRIEVED_MASK	Set when client wishes to receive CALL_RETRIEVED_EVENT messages.	0x00000010
CALL_CLEARED_MASK	Set when client wishes to receive CALL_CLEARED_EVENT messages.	0x00000020
CALL_CONNECTION_CLEARED_MASK	Set when client wishes to receive CALL_CONNECTION_CLEARED_EVENT messages.	0x00000040
CALL_ORIGINATED_MASK	Set when client wishes to receive CALL_ORIGINATED_EVENT messages.	0x00000080
CALL_CONFERENCED_MASK	Set when client wishes to receive CALL_CONFERENCED_EVENT messages.	0x00000100
CALL_TRANSFERRED_MASK	Set when client wishes to receive CALL_TRANSFERRED_EVENT.	0x00000200
CALL_DIVERTED_MASK	Set when client wishes to receive CALL_DIVERTED_EVENT messages.	0x00000400
CALL_SERVICE_INITIATED_MASK	Set when client wishes to receive CALL_SERVICE_INITIATED_EVENT messages.	0x00000800
BEGIN_CALL_MASK	Set when client wishes to receive BEGIN_CALL_EVENT messages.	0x00002000
END_CALL_MASK	Set when client wishes to receive END_CALL_EVENT messages.	0x00004000
CALL_DATA_UPDATE_MASK	Set when client wishes to receive CALL_DATA_UPDATE_EVENT messages.	0x00008000
CALL_FAILED_MASK	Set when client wishes to receive CALL_FAILED_EVENT messages.	0x00010000
CALL_DEQUEUED_MASK	Set when client wishes to receive CALL_DEQUEUED_EVENT messages.	0x00040000
RTP_STARTED_MASK	Set when client wishes to receive RTP_STARTED_EVENT messages.	0x00200000

Mask name	Description	Value
RTP_STOPPED_MASK	Set when client wishes to receive RTP_STOPPED _EVENT messages.	0x00400000

Agent State Masks

The Agent state masks specify the agent state messages that the client requests.

Table 24: Agent State Masks

Mask name	Description	Value
AGENT_LOGIN_MASK	Set when client wishes to receive “login” AGENT_STATE_EVENT messages.	0x00000001
AGENT_LOGOUT_MASK	Set when client wishes to receive “logout” AGENT_STATE_EVENT messages.	0x00000002
AGENT_NOT_READY_MASK	Set when client wishes to receive “not ready” AGENT_STATE_EVENT messages.	0x00000004
AGENT_AVAILABLE_MASK	Set when client wishes to receive “available” AGENT_STATE_EVENT messages.	0x00000008
AGENT_TALKING_MASK	Set when client wishes to receive “talking” AGENT_STATE_EVENT messages.	0x00000010
AGENT_WORK_MASK	Set when client wishes to receive “work” AGENT_STATE_EVENT messages.	0x00000020
AGENT_TALKING_PENDING_WORK_MASK	Set when client wishes to receive "talking pending work" AGENT_STATE_EVENT messages.	0x00000040
AGENT_TALKING_PENDING_NOT_READY_MASK	Set when client wishes to receive “talking pending not ready” AGENT_STATE_EVENT messages.	0x00000080
AGENT_RESERVED_MASK	Set when client wishes to receive “reserved” AGENT_STATE_EVENT messages.	0x00000100

Configuration-Information Masks

The Configuration-Information masks specify the configuration event messages that the client requests.

Table 25: Configuration Information Masks

Mask name	Description	Value
CONFIG_AGENT_MASK	Set when client wishes to receive agent configuration update messages.	0x00000001
CONFIG_CSQ_MASK	Set when client wishes to receive skill group configuration update messages.	0x00000002
CONFIG_APPLICATION_MASK	Set when client wishes to receive service configuration update messages.	0x00000004
CONFIG_DEVICE_MASK	Set when client wishes to receive device configuration update messages.	0x00000008

Failure Messages

Unified CCX can indicate errors to the client using the FAILURE_CONF, on page 83 and FAILURE_EVENT, on page 84 messages.

FAILURE_CONF

Unified CCX may use the FAILURE_CONF message, defined in the following table, in response to any request message from the client. Unified CCX sends the FAILURE_CONF message in place of the positive confirmation message specific to the request.

Note In a high availability environment, if a connection is made to a standby server, the server can reply with a FAILURE_CONF indicating error E_CTI_SERVER_NOT_MASTER.

Table 26: FAILURE_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
Status	A status code indicating the cause of the failure. The possible status codes are defined in Table 201: Error Status Codes , on page 209.	UINT	4

Table 27: FAILURE_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
Text[7]	A text description of the reason for the failure reason, if one is known.	STRING	255

FAILURE_EVENT

Unified CCX may use the FAILURE_EVENT message, defined in the following table, to asynchronously indicate a failure or error condition to the client.

Table 28: FAILURE_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
Status	A status code indicating the cause of the failure. The possible status codes are defined in Table 201: Error Status Codes , on page 209.	UINT	4

Table 29: FAILURE_EVENT Floating PatField Name Message Body Format

Floating PatField name	Value	Data type	Maximum size
Text[7]	Text field indication a text description of the failure reason (if any).	STRING	255

Configuration Messages

This section includes the following configuration message definitions:

- CONFIG_REQUEST_KEY_EVENT, on page 85
- CONFIG_KEY_EVENT, on page 85
- CONFIG_REQUEST_EVENT, on page 86
- CONFIG_BEGIN_EVENT, on page 87
- CONFIG_END_EVENT, on page 88
- CONFIG_CSQ_EVENT, on page 89
- CONFIG_APPLICATION_EVENT, on page 90
- CONFIG_AGENT_EVENT, on page 92

- CONFIG_DEVICE_EVENT, on page 93
- TEAM_CONFIG_EVENT, on page 95
- TEAM_CONFIG_REQ, on page 96
- TEAM_CONFIG_CONF, on page 97

CONFIG_REQUEST_KEY_EVENT

The CONFIG_REQUEST_KEY_EVENT can be sent by the client to request the current configuration keys for different types of data.

Table 30: CONFIG_REQUEST_KEY_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved (Only available in CTI Protocol version 11 and later.)	Set this field to 0.	UINT	4

Table 31: CONFIG_REQUEST_KEY_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$reserved_{[131]}$	Set this field to 0.	UINT	4

CONFIG_KEY_EVENT

The CONFIG_KEY_EVENT message is sent by Unified CCX in response to CONFIG_REQUEST_KEY_EVENT message. It contains the Unified CCX keys at the time of the request. Although the data type for the keys is UNSPEC, the keys should be interpreted as 8-byte integer.

Returning any key of all binary 0's tells the client that the specified configuration needs to be uploaded.

Table 32: CONFIG_KEY_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
Status	Status value of operation. See Table 34: CONFIG_KEY_EVENT Status Values , on page 86.	UINT	4

Table 33: CONFIG_KEY_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ApplicationConfigKey[177]	The Unified CCX configuration key of the customer (or all customers) for applications.	UNSPEC (8)	8
CSQConfigKey[178]	The Unified CCX configuration key of the customer (or all customers) for CSQs.	UNSPEC (8)	8
AgentConfigKey[179]	The Unified CCX configuration key of the customer (or all customers) for agents.	UNSPEC (8)	8
DeviceConfigKey[180]	The Unified CCX configuration key of the customer (or all customers) for device information.	UNSPEC (8)	8

Table 34: CONFIG_KEY_EVENT Status Values

Status Value	Value	Meaning
CONFIG_SUCCESS	0	Successful upload of configuration data.
CONFIG_SERVICE_PROVIDER	1	No data was sent because the configuration service is not requested in the service request of the OPEN_REQ message.
CONFIG_NO_KEY_SUPPORT	2	Unified CCX is unable to provide configuration key support due to an internal error.
CONFIG_UNKNOWN_DATA	3	Invalid reserved[131] field.

CONFIG_REQUEST_EVENT

The CONFIG_REQUEST_EVENT message may be sent by the client after it receives the CONFIG_KEY_EVENT message.

Unified CCX responds by sending a CONFIG_BEGIN_EVENT, CONFIG_xxx records (where xxx refers to the type of configuration records such as CSQ, Application, Agent, or Device), and then a CONFIG_END block containing all the records for that configuration item. After the client gets this new configuration data, the client should clear the existing configuration and use the new configuration set.

Table 35: CONFIG_REQUEST_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
ConfigurationMask	Bit Mask indicating what type of information is requested: 1: Agent Information 2: CSQ Information 4: Application Information 8: Device Information 0: the client is not requesting an initial configuration upload. This is used to tell Unified CCX that it is now permitted to send configuration update messages when the client does not want the initial update. What updates are received depend on the ConfigurationMask. **Note** The configuration message that can be sent out is also restricted by the configuration mask (CallMsgMask, AgentState Maks, or ConfigMsgMask) value in the OPEN_REQ message.	UINT	4
reserved(Version 11)	Set this value to 0.	UINT	4

Table 36: CONFIG_REQUEST_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$reserved_{[131]}$(required)	Set this value to 0.	UINT	4

CONFIG_BEGIN_EVENT

The CONFIG_BEGIN_EVENT signifies the beginning of configuration data from Unified CCX.

Table 37: CONFIG_BEGIN_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
ConfigType	1: Initial Configuration 2: Update	USHORT	2

Fixed part Field name	Value	Data type	Byte size
ConfigurationMask	Bit Mask indicating the type of information included: 1: Application Information 2: CSQ Information 4: Agent Information 8: Device Information	UINT	4

Table 38: CONFIG_BEGIN_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ApplicationConfigKey[177]	The Unified CCX configuration key of the customer (or all customers) for applications (if any).	UNSPEC (8)	8
CSQueueConfigKey[178]	The Unified CCX configuration key of the customer (or all customers) for CSQs (if any).	UNSPEC (8)	8
AgentConfigKey[179]	The Unified CCX configuration key of the customer (or all customers) for agents (if any).	UNSPEC (8)	8
DeviceConfigKey[180]	The Unified CCX configuration key of the customer (or all customers) for device information (if any).	UNSPEC (8)	8

CONFIG_END_EVENT

The CONFIG_END_EVENT message is sent by Unified CCX to indicate the end of a successful configuration upload or an error condition. It most likely will follow configuration records preceded by a CONFIG_BEGIN_EVENT message or it can be a response to a CONFIG_REQUEST_EVENT message indicating either an error or that there is no configuration for the items requested.

Table 39: CONFIG_END_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
Status	Indicates the status of the configuration block. See Table 41: CONFIG_END_EVENT Status Values, on page 89 for status values and descriptions.	UINT	4

Table 40: CONFIG_END_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
Text[7]	Optional Text describing Errors or other information.	STRING	255

Table 41: CONFIG_END_EVENT Status Values

Status	Value	Meaning
CONFIG_SUCCESS	0	Successful upload of configuration data.
CONFIG_SERVICE_PROVIDER	1	No data was sent because the configuration service is not requested in the service request of the OPEN_REQ message.
CONFIG_UNKNOWN_DATA	2	An invalid reserved[131] field is in the CONFIG_REQUEST_EVENT message.
CONFIG_ERROR	3	An error occurred and an invalid or partial configuration was sent.
CONFIG_EMPTY	4	No configuration data exists on Unified CCX.
CONFIG_PARTIAL	5	Partial configuration data.

CONFIG_CSQ_EVENT

The CONFIG_CSQ_EVENT message is sent to indicate a CSQ configuration update.

Table 42: CONFIG_CSQ_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
NumRecords	The number of records included in the floating part of this message. The maximum number of records is 10.	USHORT	2

Table 43: CONFIG_CSQ_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
RecordType[183]	0: Add 1: Change 2: Delete	USHORT	2
reserved[184] (Version 11)	Set to 0.	UINT	4
CSQID[184]	The Contact Service Queue ID. This ID is used to identify a CSQ internally in Unified CCX.	UINT	4
reserved[213] (Version 11)	Set to 0.	UINT	4
reserved[64]	Set to 0.	USHORT	2
reserved[192]	Set to 0.	UINT	4
AutoWork[185]	TRUE if the agent goes into work mode after handling a call from this CSQ. FALSE if not present.	BOOL	4
reserved[173]	An empty string (" ").	STRING	16
CSQName[133]	Name of the CSQ.	STRING	64
Description[134]	An empty string (" ").	STRING	128
MR Domain ID[216] (Version 11) (see Table note 1, on page 90)	0= Voice 1= Email (see Table note 2, on page 90).	UINT	4
reserved[176]	An empty string (" ").	STRING	255

Table Notes

1 A version number next to a field name in a message type definition indicates that field is used in the CTI protocol beginning with the specified version number.

2 “1”=Email will only be available in UCCX 7.0(1) and later. In earlier versions, this field will always be "0".

CONFIG_APPLICATION_EVENT

The CONFIG_APPLICATION_EVENT message is sent by Unified CCX to provide information about an Application.

Table 44: CONFIG_APPLICATION_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
NumRecords	The number of records contained in the floating part of this message. The maximum number of records that can be put in the message is 10.	USHORT	2

Table 45: CONFIG_APPLICATION_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Max Size
RecordType$_{[183]}$	0: Add 1: Change 2: Delete	USHORT	2
reserved$_{[208]}$ (Version 11)	0.	UINT	4
ApplicationID$_{[184]}$	The application ID.	UINT	4
reserved$_{[213]}$ (Version 11)	0.	UINT	4
MaxQueued$_{[129]}$	The maximum number of sessions associated with this application.	UINT	4
reserved$_{[173]}$ (optional)	An empty string (" ").	STRING	16
PreviousApplicationID$_{[132]}$	If the application ID is changed, this value stores the previous application ID.	UINT	4
ApplicationName$_{[133]}$	Name of the application.	STRING	64
Description$_{[134]}$ (optional)	Text description of the application.	STRING	128
reserved$_{[174]}$	0.	UINT	4
reserved$_{[175]}$	0.	UINT	4
reserved$_{[176]}$	An empty string (" ").	STRING	255
reserved$_{[216]}$ (Version 11)	0.	UINT	4

CONFIG_AGENT_EVENT

The CONFIG_AGENT_EVENT message is sent by Unified CCX to provide information about an agent.

Note The LoginID field is considered unique for all records. Two records sent with matching LoginID's are considered to be the same record.

Table 46: CONFIG_AGENT_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
NumRecords	The number of records contained in the floating portion of this message. The maximum number of records that can be put in the message is 10.	USHORT	2

Table 47: CONFIG_AGENT_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
RecordType$_{[183]}$	0: Add 1: Change 2: Delete	USHORT	2
reserved$_{[208]}$ (Version 11)	0	UINT	4
AgentType$_{[189]}$	1: Agent 2: Supervisor	USHORT	2
LoginID$_{[190]}$	The agent's Unified CCX Login ID.	STRING	64
reserved$_{[214]}$ (Version 11)	" "	STRING	64
reserved$_{[207]}$ (Version 11)	" "	STRING	32
LastName$_{[138]}$	The agent's last name.	STRING	32
FirstName$_{[137]}$	The agent's first name.	STRING	32
Extension$_{[173]}$	The agent's phone extension number.	STRING	16
Description$_{[134]}$	" "	STRING	128

Floating part Field name	Value	Data type	Maximum size
reserved[141]	0	UINT	4
NumCSQ[191]	The Number of the CSQ to which this agent belongs. A pair consists of the next two fields with a maximum of 30. For example: If the Number of CSQs for the agent is 2, then four fields follow this one (CSQID1 and Reserved and CSQID2 and Reserved).	USHORT	2
CSQID[62]	The ID of the CSQ in which the agent is a member.	UINT	4
reserved[64]	0	USHORT	2

CONFIG_DEVICE_EVENT

The CONFIG_DEVICE_EVENT message is sent by Unified CCX to provide information about a device's configuration. Unified CCX sends CONFIG_DEVICE_EVENT messages for all route points, agents, and CTI ports.

Table 48: CONFIG_DEVICE_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
NumRecords	The Number of records included in the floating part of this message. The maximum Number of records allowed is 10.	USHORT	2

Table 49: CONFIG_DEVICE_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
RecordType [183]	0: Add1: Change2: Delete	USHORT	2
reserved [208] (Version 11)	0	UINT	4

Floating part Field name	Value	Data type	Maximum size
DeviceType [195]	See Table 50: Message Floating Field Data for Each Type of Data, on page 94.	USHORT	2
MaxQueued [129]		UINT	4
DeviceField0 [184]		UINT	4
DeviceField1 [193]		UINT	4
DeviceField2 [11]		STRING	40
reserved [10]	Varies.	STRING	32
DeviceField4 [173]	See Table 50: Message Floating Field Data for Each Type of Data, on page 94 .	STRING	16
Description [134]	Text description of the entity.	STRING	128

Table 50: Message Floating Field Data for Each Type of Data, on page 94 specifies the floating data fields in the CONFIG_DEVICE_EVENT message according to the device type. For example:

- If the CONFIG_DEVICE_EVENT message is for agent data:
  - In that message's DeviceType field, enter 3.
  - In DeviceField0, enter the AgentID.
  - In DeviceField2, enter the AgentLoginID, and so on.
- If the message is for route points:
  - In the DeviceType field, enter 5.
  - In DeviceField1, enter the CallControlGroupID, and so on.

Table 50: Message Floating Field Data for Each Type of Data

Message fields	For agent data	For route point data	For CTI port data	For call control group data
DeviceType	3	5	6	7
DeviceField0	AgentRecordID (Unified CCX internal ID for this agent record.)	CallControlGroupID	CallControlGroupID	CallControlGroupID
DeviceField1	Not used	ApplicationID	Not used	Not used

Message fields	For agent data	For route point data	For CTI port data	For call control group data
DeviceField2	LoginID	RoutePoint	CTIPortID	Not used
MaxQueued	Not used	SessionLimit	Not used	CTIPortCount
DeviceField3	Not used	Not used	Not used	Not used
DeviceField4	AgentExtension	Not used	Not used	Not used

TEAM_CONFIG_EVENT

The TEAM_CONFIG_EVENT message informs the clients of Unified CCX that there is a change to the agent team configuration. A team may include agents, one primary supervisor, secondary supervisors, and CSQs.

Table 51: TEAM_CONFIG_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
TeamID	The team ID.	UINT	4
NumRecords	The number of AgentID, AgentFlag, Recordtype, and MemberType fields present in the floating part of the message, up to a maximum of 64.	USHORT	2
Operation	The type of agent team configuration change to perform. One of the following values: 1: Add Team 2: Remove Team (No optional floating part is present. The client is responsible for removing all team related data for this team). 3: Change Team (includes TeamName change, addition/deletion of team members, addition/deletion of primary supervisors, addition/deletion of secondary supervisors)	USHORT	2
reserved	0	UNSPEC (8)	8

Table 52: TEAM_CONFIG_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
TeamName[204] (required)	The assigned team name. This field must be the first floating field.	STRING	50
AgentID[194] (optional)	The AgentID of an agent or supervisor if the MemberType is agent/supervisor member or the CSQID of a member of the team if the MemberType is CSQ member.	STRING	32
AgentFlags[87] (optional)	A set of flags indicating the attributes of the preceding AgentID. Possible values are: 0x0001: Primary Supervisor 0x0002: Temporary Agent 0x0004: Supervisor (0 flag is for regular agent)	USHORT	2
RecordType[183] (optional)	The type of agent change. It can have following values: 0: Add 2: Delete	USHORT	2
MemberType[205]	The team member type: 1: Agent or Supervisor member 2: CSQ member	USHORT	2

TEAM_CONFIG_REQ

The TEAM_CONFIG_REQ message requests team configuration data from Unified CCX. If initial configuration is requested, the initial configuration is sent followed by the TEAM_CONFIG_CONF message.

Table 53: TEAM_CONFIG_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4

Fixed part Field name	Value	Data type	Byte size
ConfigParam	This parameter indicates if the client wants initial bulk upload and/or a team configuration update. The values are: 0x1: initial configuration requested. 0x2: updates only.	USHORT	2
reserved	Set this value to 0.	UNSPEC	8

TEAM_CONFIG_CONF

The TEAM_CONFIG_CONF message is an acknowledgement from Unified CCX to the client that it received the request for team configuration data. This message also indicates to clients that they should expect to receive team configuration updates.

Table 54: TEAM_CONFIG_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	The same value as the InvokeID from the corresponding request message.	UINT	4
Status	The response to the TEAM_CONFIG_REQ. message. The values are: 0: Success 1: Unified CCX Internal Error	UINT	4

Table 55: TEAM_CONFIG_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$Text_{[7]}$ (optional)	Detailed text describing the Status field value.	STRING	128

Agent-State Messages

This section includes the following message definitions:

- AGENT_STATE_EVENT, on page 98
- QUERY_AGENT_STATE_REQ, on page 100

- QUERY_AGENT_STATE _CONF, on page 101
- SET_AGENT_STATE_REQ, on page 102
- SET_AGENT_STATE_CONF, on page 103

AGENT_STATE_EVENT

An agent-state change (such as logging on, becoming available to handle incoming calls, and so on) sends to the client an AGENT_STATE_EVENT message, defined in the following tables.

Table 56: AGENT_STATE_EVENT Fixed Part Message Body Format

Fixed part Field name	Description	Data type	Byte size
reserved	A value of 0.	UINT	4
reserved	A value of 1.	UINT	4
reserved	A value of 0.	UINT	4
reserved	A value of 21.	USHORT	2
CSQState	One of the values representing the current state of the agent (see Table 3: Agent States and Their Message Values, on page 32). If only one event is sent for the agent (Not one for each CSQ to which the agent belongs) this is set to 0.	USHORT	2
StateDuration	The number of seconds since the agent entered this state (typically 0).	UINT	4
CSQID	The Customer Service Queue ID affected by the state change, as known by CCX.	UINT	4
reserved	A value of 0XFFFFFFFF.	UINT	4
reserved	A value of 0.	USHORT	2
AgentState	The value representing the current state of the associated agent (see Table 3: Agent States and Their Message Values, on page 32).	USHORT	2
EventReasonCode	A Unified CCX code indicating the reason for the state change. (see Table 1)	USHORT	2
reserved	A value of 1.	INT	4
reserved	A value of 0.	UINT	4

Fixed part Field name	Description	Data type	Byte size
reserved	A value of 0.	USHORT	2
reserved	A value of 0.	UINT	4
reserved	A value of 0.	INT	4
reserved	A value of 1.	INT	4
numCSQs	If information for more than one CSQ is passed, this must be non-zero and indicates the number of records (CSQID, and CSQState, $Reserved_{[63]}$, and $Reserved_{[64]}$ fields) present in the floating part of the message. There can be up to 99 records. If more than 99 are required, then another AGENT_STATE_EVENT message must be sent. If there are 0 records in the floating part of the message, then a single record (of CSQID and CSQState) is specified in the fixed part of the message.	USHORT	2

Table 57: AGENT_STATE_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$CTIClientSignature_{[23]}$ (optional)	The signature of the client that is associated with this agent.	STRING	64
$reserved_{[5]}$	Ignore this value.	STRING	12
$AgentExtension_{[4]}$	The agent's IP phone extension.	STRING	16
$AgentInstrument_{[6]}$ (optional)	The agent's IP phone number.	STRING	64
$AgentID_{[194]}$ (optional)	The agent's Unified CCX login.	STRING	32
Duration $_{[150]}$ (optional)	If present specifies in seconds the anticipated time in the state specified. This is useful for work states to estimate the time before going ready or not ready.	UINT	4
$NextAgentState_{[123]}$	The next agent state (if known).	USHORT	2

Floating part Field name	Value	Data type	Maximum size
CSQID[62]	The ID of the CSQ affected by the state change. If a particular CSQ is specified, the state in all other CSQs is implicitly made BUSY_OTHER.	UINT	4
reserved[63]	Set this value to 0XFFFFFFFF.	UINT	4
reserved[64]	A value of 0.	USHORT	2
CSQState[65]	One of the values from Table 3: Agent States and Their Message Values, on page 32 representing the current state of the associated agent with respect to the CSQ. There can be more than one CSQ field in the message (see numCSQs above).	USHORT	2

QUERY_AGENT_STATE_REQ

The QUERY_AGENT_STATE_REQ message, defined in the following tables, allows a client to retrieve the current state of an agent at a specified device.

Table 58: QUERY_AGENT_STATE_REQ Field Name Message Body Format

Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
reserved	Set this value to 1.	INT	4
reserved	Set this value to 0	INT	4

Table 59: QUERY_AGENT_STATE_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
AgentExtension[4]	The agent's IP phone number.	STRING	16
AgentInstrument[6]	The agent's IP phone number. At least one of AgentExtension, AgentID, or AgentInstrument must be provided.	STRING	64

Floating part Field name	Value	Data type	Maximum size
AgentID$_{[194]}$	The agent's Unified CCX login ID.	STRING	32

QUERY_AGENT_STATE _CONF

The QUERY_AGENT_STATE _CONF message, defined in the following tables, is confirmation of the receipt of the QUERY_AGENT_STATE_REQ message.

Table 60: QUERY_AGENT_STATE _CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
AgentState	One of the values from representing the current state of the associated agent (see Table 3: Agent States and Their Message Values, on page 32).	USHORT	2
numCSQs	The count of CSQID, reserved$_{[63]}$, reserved$_{[64]}$, and CSQState as a group in the floating part. The maximum count allowed is 20.	USHORT	2
reserved	This value is set to 1.	INT	4
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 0.	USHORT	2
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4

Table 61: QUERY_AGENT_STATE _CONF FloatingField Name Message Body Format

FloatingField name	Value	Data type	Maximum size
reserved$_{[5]}$ (optional)	Ignore this value.	STRING	12
AgentExtension$_{[4]}$ (optional)	The agent's Unified CCX IP phone number, if the agent is logged on.	STRING	16

FloatingField name	Value	Data type	Maximum size
AgentInstrument[6] (optional)	The agent's IP phone number, if the agent is logged on.	STRING	64
AgentID[194]	The agent's Unified CCX login.	STRING	32
CSQID[62]	The ID of the CSQ affected by the state change. If a particular CSQ is specified, the state in all other CSQs is implicitly made BUSY_OTHER.	UINT	4
reserved[63]	This value is set to 0.	UINT	4
reserved[64]	This value is set to 0.	USHORT	2
CSQState[65]	One of the values from Table 3: Agent States and Their Message Values, on page 32 representing the current state of the associated agent with respect to the CSQ identified by CSQID.	USHORT	2

SET_AGENT_STATE_REQ

The SET_AGENT_STATE_REQ message, defined in the following tables, allows a client to change an ACD agent's state.

Table 62: SET_AGENT_STATE_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
AgentState	One of the values from representing the desired state of the associated agent (see Table 3: Agent States and Their Message Values, on page 32).	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
EventReasonCode	A Unified CCX code indicating the reason for the agent state change (see Table 3: Agent States and Their Message Values, on page 32).	USHORT	2

Fixed part Field name	Value	Data type	Byte size
ForcedFlag	Unified CCX is requested to force this state change regardless of its validity. Used only with AGENT_STATE_LOGIN or AGENT_STATE_LOGOFF: 0: FALSE 1: TRUE 2: Agent authentication only. No agent state change. Use with AGENT_STATE_LOGIN (Version 12)	UCHAR	1
AgentCapacity[Version 12]	Specifies the agent capacity. 0x1: Outbound feature enabled	UINT	4

Table 63: SET_AGENT_STATE_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$AgentInstrument_{[6]}$ (required)	The agent's IP phone number.	STRING	64
$AgentPassword_{[49]}$ (optional)	The password that allows an agent to log into a CSQ. This field is required when AgentState is AGENT_STATE_LOGIN.	STRING	64
$AgentID_{[194]}$	The agent's Unified CCX login ID. This field is required when the AgentState is AGENT_STATE_LOGIN or AGENT_STATE_LOGOFF.	STRING	32

SET_AGENT_STATE_CONF

The SET_AGENT_STATE_CONF message, defined in the following table, confirms the successful completion of a request for setting the agent state.

Table 64: SET_AGENT_STATE_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

Call-Event Messages

Call-Event messages are unsolicited messages sent to clients when Unified CCX reports that a call-event has occurred. There are no request or confirmation messages associated with these unsolicited event messages. Every call can be announced to the client with an unsolicited BEGIN_CALL_EVENT message, informing the client that it has just been associated with a new call and providing the initial call context data.

Additional call events are then sent to the client as the call is handled, depending upon the type of ACD involved and the treatment that the call receives. Finally, an END_CALL_EVENT message is sent to the client when its association with a call is dissolved.

This section includes the following message definitions:

- BEGIN_CALL_EVENT, on page 105
- END_CALL_EVENT, on page 107
- CALL_DATA_UPDATE_EVENT, on page 108
- CALL_DELIVERED_EVENT, on page 111
- CALL_ESTABLISHED_EVENT, on page 114
- CALL_HELD_EVENT, on page 116
- CALL_RETRIEVED_EVENT, on page 117
- CALL_CLEARED_EVENT, on page 118
- CALL_CONNECTION_CLEARED_EVENT, on page 119
- CALL_ORIGINATED_EVENT, on page 120
- CALL_FAILED_EVENT, on page 121
- CALL_CONFERENCED_EVENT, on page 123
- CALL_TRANSFERRED_EVENT, on page 125
- CALL_DIVERTED_EVENT, on page 128
- CALL_SERVICE_INITIATED_EVENT, on page 129
- CALL_QUEUED_EVENT, on page 131
- CALL_DEQUEUED_EVENT, on page 1331
- RTP_STARTED_EVENT (OPTIONAL), on page 134
- RTP_STOPPED_EVENT (OPTIONAL), on page 136

Primary.Actual Field Format

From Cisco Unified CCX 8.0(1) release onwards, Join Across Line (JAL) feature is supported. If an agent uses this feature while handling calls between an IPCC extension and non-ACD extension (non-IPCC extension), and the call on his non-ACD extension survives, then Unified CCX will treat this as a business call, which moved to secondary extension of the agent.

In this scenario, if a CTI client like CAD uses the Unified CCX CTI protocol version 14 for fetching call events, it will receive the call events in the new Primary.Actual field format. In this format, all connection

device and subject device id fields in many of the Unified CCX CTI call events sent to the client will be changed to include the primary ICD extension concatenated with a "dot" and the actual extension used (for example, 1000.1001).

General Rules

The general rules for the Primary.Actual field format in the Unified CCX CTI Protocol Version 14 are mentioned below:

- This format applies only to non-ICD secondary lines for logged on agents.
- This format is backward compatible with Unified CCX CTI clients that use versions prior to 14. For backward compatibility, Unified CCX removes the primary ID field and returns only the actual extension ID.
- All instances of ConnectionDeviceID, subject DeviceID, ANI, DNIS, and DialedNumbers fields are affected. These fields exist in many _CONF and _EVENT messages. Clients can also provide the new format to _REQ messages (not required). Unified CCX will ignore the primary part and utilize only the actual extension to be used in the third party call control methods.
- The device type shall also change to reflect the field type. A new field DEVID_NON_ACD_DEVICE is introduced to label “Primary.Actual” fields.
- Since the device ID is able to hold two IDs, the maximum length of the ID is essentially cut in half. With this new format, the maximum length for the device ID fields is cut from 64 bytes to 31 bytes (1 byte for the dot) assuming that both the primary and the actual IDs are of the same length.

BEGIN_CALL_EVENT

The first association between a call and a client (a route point, a CTI port, or an agent phone to which the agent has logged in) generates a BEGIN_CALL_EVENT message to the client, providing the call ID and the initial call context data. This message is optional.

The CallID identifies the call, and the ConnectionDeviceType and ConnectionDeviceID uniquely identify the client’s local call connection, if any, or another valid call connection.

This message always precedes any other event messages for that call. Subsequent changes to the call context data (if any) result in CALL_DATA_UPDATE_EVENT messages containing the changed call data being sent to the client.

Note There can be multiple calls with the same CallID value.

The following tables define the format of the BEGIN_CALL_EVENT message.

Table 65: BEGIN_CALL_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	Set this value to 0.	UINT	4
reserved	Set this value to 1.	UINT	4

Fixed part Field name	Value	Data type	Byte size
reserved	Set this value to 21.	USHORT	2
NumCTIClients	The number of clients previously associated with this call. This value also indicates the number of client signatures and timestamps that are present in the floating part of the message.	USHORT	2
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2
CallType	The general classification of the call type. See CallType Values, on page 197.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See Table 193: ConnectionDevice Type Values, on page 196	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
CalledPartyDisposition	Indicates the disposition of the called party. See Disposition Values, on page 209.	USHORT	2

Table 66: BEGIN_CALL_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	MaxSize
ConnectionDeviceID$_{[25]}$ (required)	The ID of the connection between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
ANI$_{[8]}$ (optional)	ANI (Automatic Number Identification). The telephone number of the person making the call. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	40
UserToUserInfo$_{[9]}$ (optional)	The user-to-user information element.	UNSPEC	131

Floating part Field name	Value	Data type	MaxSize
DNIS[10] (optional)	The DNIS (Dialed Number Identification Service) provided with the call; that is, the number associated with a call on a switch. This can be different, though often it is not, from the number the caller dialed. This is different if the call is transferred. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	32
DialedNumber[11] (optional)	The telephone number dialed. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	40
CallerEnteredDigits[12] (optional)	The digits entered by the caller in response to IVR prompting.	STRING	40
CallVariable1[13] (optional)	Call-related variable data.	STRING	40
CallVariable2[14]. (optional)throughCallVariable9[21] (optional)	…	…	…
CallVariable10[22] (optional)	Call-related variable data.	STRING	40
CallWrapupData[46] (optional)	Call-related wrapup data.	STRING	40
NamedVariable[82] (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray[83] (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252

END_CALL_EVENT

An END_CALL_EVENT message is generated when the association between a call and the client is dissolved and no further call-event messages for the call are sent to this client. This message does not necessarily indicate that the subject call has been terminated.

This message is optional.

The following tables define the format of the END_CALL_EVENT message.

Table 67: END_CALL_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	Set this value to 0.	UINT	4
reserved	Set this value to 1.	UINT	4
reserved	Set this value to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4

Table 68: END_CALL_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The ID of the connection between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_DATA_UPDATE_EVENT

Changes to the call context data generate a CALL_DATA_UPDATE_EVENT to the client that contains only the items that have changed. This message is optional. The initial call context is provided in the BEGIN_CALL_EVENT message.

Note This event MUST be sent if Unified CCX changes the call ID with no other notifying events such as CALL_CONFERENCED_EVENT or CALL_TRANSFERRED_EVENT. One circumstance in Unified CCX where this can happen is when a non-ACD call is transferred through a route point into the ACD system. In the 1-2-1 transfer model, when the call is transferred, call 1 is the resultant call but is unknown to monitoring systems.

The CALL_DATA_UPDATE_EVENT message is defined in the following tables.

Table 69: CALL_DATA_UPDATE_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	Set this value to 0.	UINT	4
reserved	Set this value to 1.	UINT	4
reserved	Set this value to 21.	USHORT	2
NumCTIClients	The number of clients associated with this call. This value also indicates the number of client signatures and timestamps that are present in the floating part of the message.	USHORT	2
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2
CallType	The general classification of the call type. See CallType Values, on page 197.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
NewConnectionDeviceType	The type of device ID supplied in the NewConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
NewCallID	The new Call ID value assigned to this call by Unified CCX.	UINT	4
CalledPartyDisposition	The disposition of the called party. See Disposition Values, on page 209.	USHORT	2
CampaignID	The Campaign ID value. Set this field to zero if not used.	UINT	4
reserved	Set this value to zero.	UINT	4

Table 70: CALL_DATA_UPDATE_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The previous ID of the call connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
NewConnectionDeviceID[47] (required)	The new ID of the call connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
ANI[8] (optional)	ANI (automatic Number Identification). The telephone number of the person making the call. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	40
CallerEnteredDigits[12] (optional)	The digits entered by the caller in response to the IVR prompting.	STRING	40
CallVariable1[13] (optional)	Call-related variable data.	STRING	40
CallVariable2[14]. (optional)throughCallVariable9[21] (optional)	…	…	…
CallVariable10[22] (optional)	Call-related variable data.	STRING	40
CallWrapupData[46] (optional)	Call-related wrapup data.	STRING	40
NamedVariable[82] (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray[83] (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252
CustomerPhoneNumber (optional)	Customer phone number.	STRING	20
CustomerAccountNumber[96] (optional)	Customer Account Number.	STRING	32

CALL_DELIVERED_EVENT

The arrival of a call at any device on Unified CCX generates a CALL_DELIVERED_EVENT message to the client. This event generally indicates the called device is ringing. This is defined in the following tables.

The LocalConnectionState field can be used to distinguish between those cases of the call arriving at the switch and those cases of the call arriving at an agent IP phone.

Table 71: CALL_DELIVERED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2
ApplicationID	The Application ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
CSQID	The Contact Service Queue ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
reserved	This value is set to 0.	USHORT	2
AlertingDeviceType	Indicates the device ID type supplied in the **Alerting**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CallingDeviceType	Indicates the device ID type supplied in the **Calling**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CalledDeviceType	Indicates the device ID type supplied in the **Called**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2

Fixed part Field name	Value	Data type	Byte size
LastRedirectDeviceType	Indicates the type of the device ID supplied in the **LastRedirect**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2

Table 72: CALL_DELIVERED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID [25] (required)	The ID of the connection between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
AlertingDeviceID[26] (required)	The device ID of the device that is alerting. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CallingDeviceID[27] (optional)	The device ID of the calling device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CalledDeviceID[28] (required)	The device ID of the originally called device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

Floating part Field name	Value	Data type	Maximum size
LastRedirectDeviceID[29] (optional)	The device ID of the previously alerted device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
SecondaryCallID[202] (optional)	The ID of the consultation Call that Unified CCX placed from the CTI port to the agent IP phone.	UINT	4
ANI[8] (optional)	ANI (Automatic Number Identification). The telephone number of the person making the call. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	40
UserToUserInfo[9] (optional)	The user-to-user information element.	UNSPEC	131
DNIS[10] (optional)	The DNIS (Dialed Number Identification Service) provided with the call; that is, the number associated with a call on a switch. This can be different, though often it is not, from the number the caller dialed. This is different if the call is transferred. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	32
DialedNumber[11] (optional)	The telephone number dialed. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	40
CallerEnteredDigits[12] (optional)	The digits entered by the caller in response to the IVR prompting.	STRING	40
CallVariable1[13] (optional)	Call-related variable data.	STRING	40
CallVariable2[14]. (optional) through CallVariable9[21] (optional)	…	…	…
CallVariable10[22] (optional)	Call-related variable data.	STRING	40
CallWrapupData[46] (optional)	Call-related wrapup data.	STRING	40

Floating part Field name	Value	Data type	Maximum size
NamedVariable[82] (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray[83] (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252

CALL_ESTABLISHED_EVENT

The answering of a call on an IP phone on Unified CCX creates a new call connection and generates a CALL_ESTABLISHED_EVENT message to the client. This is defined in the following tables.

The CallID identifies the call, and the ConnectionDeviceType and ConnectionDeviceID uniquely identify the new call connection that was created. When more than one call with the same CallID value exists, the connection being created by this CALL_ESTABLISHED_EVENT shall apply to the call that does not yet have a destination connection established.

Table 73: CALL_ESTABLISHED_EVENT Fixed Field Name Message Body Format

Fixed Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2
ApplicationID	The Application ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
CSQID	The Contact Service Queue ID of the call.	UINT	4

Fixed Field name	Value	Data type	Byte size
reserved	This value is set to 0xffffffff.	UINT	4
reserved	This value is set to 0.	USHORT	2
AnsweringDeviceType	Indicates the type of the device ID supplied in the AnsweringDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CallingDeviceType	Indicates the device ID type supplied in the **Calling**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CalledDeviceType	Indicates the device ID type supplied in the **Called**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LastRedirectDeviceType	Indicates the type of the device ID supplied in the **LastRedirect**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 74: CALL_ESTABLISHED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The ID of the connection between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
AnsweringDeviceID[30] (required)	The device ID of the device that answered the call. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CallingDeviceID[27](optional)	The device ID of the calling device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

Floating part Field name	Value	Data type	Maximum size
CalledDeviceID[28](required)	The device ID of the originally called device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
LastRedirectDeviceID[29] (optional)	The device ID of the previously alerted device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_HELD_EVENT

Placing a call on hold sends to the client a CALL_HELD_EVENT message, defined in the following tables.

Table 75: CALL_HELD_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
HoldingDeviceType	Indicates the device ID type supplied in the HoldingDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 76: CALL_HELD_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The ID of the connection between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
HoldingDeviceID[31] (required)	The device ID of the device that activated the hold. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_RETRIEVED_EVENT

Resuming a call previously placed on hold sends to the client a CALL_RETRIEVED_EVENT message, defined in the following tables.

Table 77: CALL_RETRIEVED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
RetrievingDeviceType	Indicates the type of the device ID supplied in the RetrievingDeviceID floating field.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 78: CALL_RETRIEVED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The device ID between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
RetrievingDeviceID[32] (optional)	The device ID of the device that de-activated hold. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_CLEARED_EVENT

A CALL_CLEARED_EVENT message, defined in the following tables, is sent to the client when a call is terminated, normally when the last device disconnects from a call.

Table 79: CALL_CLEARED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
reserved	This value is set to 0.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 80: CALL_CLEARED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
reserved	This value is left blank.	STRING	64

CALL_CONNECTION_CLEARED_EVENT

When a party drops from a conference call, a CALL_CONNECTION_CLEARED_EVENT message, defined in the following tables, can be sent to the client.

Table 81: CALL_CONNECTION_CLEARED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
ReleasingDeviceType	Indicates the device ID type supplied in the ReleasingDeviceID floating field.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 82: CALL_CONNECTION_CLEARED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The ID of the cleared connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
ReleasingDeviceID[33] (required)	The device ID of the device that cleared the connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_ORIGINATED_EVENT

The initiation of a call at any monitored device sends to the client a CALL_ORIGINATED_EVENT, defined in the following tables.

Table 83: CALL_ORIGINATED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 7.	USHORT	2
ApplicationID	The Application ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
CSQID	The Contact Service Queue ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	USHORT	2
CallingDeviceType	Indicates the device ID type supplied in the **Calling**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CalledDeviceType	Indicates the device ID type supplied in the **Called**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 84: CALL_ORIGINATED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The device ID between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CallingDeviceID[27] (required)	The device ID of the calling device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CalledDeviceID[28] (required)	The device ID of the called device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_FAILED_EVENT

A CALL_FAILED_EVENT message, defined in the following tables, can be sent to the client when a call cannot be completed.

Table 85: CALL_FAILED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
FailingDeviceType	Indicates the type of the device ID supplied in the FailingDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CalledDeviceType	Indicates the type of the device ID supplied in the CalledDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 86: CALL_FAILED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The device ID between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
FailingDeviceID$_{[34]}$ (optional)	The device ID of the failing device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

Floating part Field name	Value	Data type	Maximum size
CalledDeviceID[28] (optional)	The device ID of the called device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_CONFERENCED_EVENT

The joining of calls into a conference call causes a CALL_CONFERENCED_EVENT message, defined in the following tables, to be sent to the client. Unified CCX MUST ensure that the CALL_CONFERENCED_EVENT is sent after any DELIVERED events for the secondary call, even in the case of a blind conference call.

Table 87: CALL_CONFERENCED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
PrimaryDeviceType	Indicates the connection type supplied in the PrimaryDeviceID floating field.	USHORT	2
PrimaryCallID	The Call ID value that Unified CCX assigned to the primary call.	UINT	4
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2
reserved	This value is set to 0xffffffff.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
reserved	This value is set to 0.	USHORT	2
NumParties	The number of active connections associated with this conference call, up to a maximum of 16. This value also indicates the number of ConnectedPartyCallID, ConnectedPartyDeviceType, and ConnectedPartyDeviceID floating fields present in the floating part of the message.	USHORT	2

Fixed part Field name	Value	Data type	Byte size
SecondaryDeviceType	Indicates the connection type of the ID supplied in the SecondaryDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
SecondaryCallID	The Call ID value that Unified CCX assigns to the secondary call.	UINT	4
ControllerDeviceType	Indicates the device ID type supplied in the ControllerDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
AddedPartyDeviceType	Indicates device ID type supplied in the AddedPartyDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 88: CALL_CONFERENCED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
PrimaryDeviceID$_{[35]}$ (required)	The device ID of the primary call connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
SecondaryDeviceID$_{[36]}$ (required)	The device ID of the secondary call connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
ControllerDeviceID$_{[37]}$ (required)	The device ID of the conference controller device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

Floating part Field name	Value	Data type	Maximum size
AddedPartyDeviceID[38] (optional)	The device ID of the device added to the call. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
ConnectedPartyCallID[39] (optional)	The Call ID value assigned to one of the conference call parties. There can be more than one ConnectedPartyCallID field in the message (see the preceding TypeParties field description).	UINT	4
ConnectedPartyDeviceType[40] (optional)	The device ID type supplied in the following ConnectedPartyDeviceID floating field. See ConnectionDeviceType Values, on page 196. There can be more than one ConnectedPartyDeviceType field in the message (see the preceding TypeParties field description). This field always immediately follows the corresponding ConnectedPartyCallID field.	USHORT	2
ConnectedPartyDeviceID[41] (optional)	The device ID of one of the conference call parties. See DeviceType Values, on page 192. There can be more than one ConnectedPartyDeviceID field in the message (see the preceding TypeParties field description). This field always immediately follows the corresponding ConnectedPartyDeviceType field. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_TRANSFERRED_EVENT

The transfer of a call to another destination causes a CALL_TRANSFERRED_EVENT message, defined in the following tables, to be sent to the client.

A CALL_TRANSFERRED_EVENT must always be sent AFTER any corresponding CALL_DELIVERED_EVENT for the secondary call.

Table 89: CALL_TRANSFERRED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
PrimaryDeviceType	Indicates the connection type supplied in the PrimaryDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
PrimaryCallID	The Call ID value that Unified CCX assigned to the primary call.	UINT	4
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2
reserved	This value is set to 0xffffffff.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
reserved	This value is set to 0.	USHORT	2
NumParties	The number of active connections associated with this conference call, up to a maximum of 16. This value also indicates the number of the ConnectedPartyCallID, the ConnectedPartyDeviceType, and the ConnectedPartyDeviceID floating fields present in the floating part of the message.	USHORT	2
SecondaryDeviceType	Indicates the connection type supplied in the SecondaryDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
SecondaryCallID	The Call ID value that Unified CCX assigned to the secondary call.	UINT	4
TransferringDeviceType	Indicates the device ID type supplied in the TransferringDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2

Fixed part Field name	Value	Data type	Byte size
TransferredDeviceType	Indicates the device ID type supplied in the TransferredDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 90: CALL_TRANSFERRED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
PrimaryDeviceID$_{[35]}$ (required)	The device ID of the primary call connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
SecondaryDeviceID$_{[36]}$ (required)	The device ID of the secondary call connection. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
TransferringDeviceID$_{[42]}$ (required)	The device ID of the device that transferred the call. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
TransferredDeviceID$_{[43]}$ (required)	The device ID of the device to which the call was transferred. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
ConnectedPartyCallID$_{[39]}$ (optional)	The Call ID value assigned to one of the call parties. There can be more than one ConnectedPartyCallID field in the message (see the preceding NumParties field description).	UINT	4

Floating part Field name	Value	Data type	Maximum size
ConnectedPartyDeviceType[40] (optional)	Indicates the type of the device ID supplied in the following ConnectedPartyDeviceID floating field. There can be more than one ConnectedPartyDeviceType field in the message (see the preceding TypeParties field description). This field always immediately follows the corresponding ConnectedPartyCallID field.	USHORT	2
ConnectedPartyDeviceID[41] (optional)	The device ID of one of the call parties. There can be more than one ConnectedPartyDeviceID field in the message (see the preceding NumParties field description). This field always immediately follows the corresponding ConnectedPartyDeviceType field. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_DIVERTED_EVENT

The moving of a call from one device or target to another can cause a CALL_DIVERTED_EVENT message, defined in the following tables, to be sent to the client. Examples of this are a call ringing at one agent and getting diverted to another. Another example is a call getting diverted out of queue to an agent.

Table 91: CALL_DIVERTED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
Application ID	The Application ID of the call.	UNIT	4
reserved	This value is set to 0xffffffff.	UNIT	4

Fixed part Field name	Value	Data type	Byte size
DivertingDeviceType	The type of the device ID supplied in the DivertingDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CalledDeviceType	The type of the device ID supplied in the CalledDeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	A reason or explanation for the occurrence of the event. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 92: CALL_DIVERTED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The device ID between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
DivertingDeviceID[44] (required)	The device ID of the device from which the call was diverted. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CalledDeviceID[28] (required)	The device ID of the device to which the call was diverted. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_SERVICE_INITIATED_EVENT

The initiation of telecommunications service (“dial tone”) at a device causes a CALL_SERVICE_INITIATED_EVENT message, defined in the following tables, to be sent to the client.

Table 93: CALL_SERVICE_INITIATED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2
ApplicationID	The Application ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
CSQID	The Contact Service Queue ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
reserved	This value is set to 0.	USHORT	2
CallingDeviceType	Indicates the device ID type supplied in the **Calling**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 94: CALL_SERVICE_INITIATED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The device ID between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CallingDeviceID[27] (required)	The device ID of the calling device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

CALL_QUEUED_EVENT

The placing of a call in a queue pending the availability of some resource causes a CALL_QUEUED_EVENT message, defined in the following tables, to be sent to the client.

Clients with Client Events Service can receive this message when an outbound call is queued waiting for a resource. Clients with All Events Service can also receive this message when inbound calls are queued.

Table 95: CALL_QUEUED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
ApplicationID[77]	The Application ID of the call.	UINT	4
reserved	This value is set to 0xffffffff.	UINT	4
QueueDeviceType	Indicates the device ID type supplied in the **Queue** DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2

Fixed part Field name	Value	Data type	Byte size
CallingDeviceType	Indicates the device ID type supplied in the **Calling**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
CalledDeviceType	Indicates the device ID type supplied in the **Called**DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
LastRedirectDeviceType	Indicates the device ID type supplied in the **LastRedirect** DeviceID floating field. See DeviceType Values, on page 192.	USHORT	2
reserved	This value is set to zero.	USHORT	2
NumCSQs	The number of CSQs to which the call has queued, up to a maximum of 20.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 96: CALL_QUEUED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The device ID between the call and the device. This is the CTI Port that the call is on. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
QueuedDeviceID [45] (required)	The ID of the queuing device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CallingDeviceID[27] (optional)	The calling line ID (if any). In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64

Floating part Field name	Value	Data type	Maximum size
CalledDeviceID[28] (required)	The DN of the Route point that was called. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
LastRedirectDeviceID[29] (optional)	The device ID of the redirecting device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format, on page 104 will apply to this field.	STRING	64
CSQID[62] (required)	The Contact Service Queue ID of the call.	UINT	4
reserved[63]	This value is set to 0xffffffff.	UINT	4
reserved[64]	This value is set to 0.	USHORT	2

CALL_DEQUEUED_EVENT

The explicit removal of a call from a queue causes a CALL_DEQUEUED_EVENT message, defined in the following tables, to be sent to the client.

Note that this event is not reported when calls leave a queue “normally” (that is, due to resource availability, call termination, and so on). In those cases, a DIVERTED event is expected. It is not currently anticipated that Unified CCX will need to send this event. One possible exception would be if a call is queued to multiple queues and then a script or some mechanism on Unified CCX dequeues the call from less than all the queues.

Table 97: CALL_DEQUEUED_EVENT Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
reserved	This value is set to 21.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4
ApplicationID	The Application ID of the call.	UINT	4

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0xffffffff.	UINT	4
QueueDeviceType	Indicates the type of device supplied in the QueueDeviceID floating field.	USHORT	2
NumQueued	The number of calls remaining in the queue for this service.	USHORT	2
NumCSQs	The number of CSQs that the call has been removed from, up to a maximum of 20. A zero value indicates that the call has been implicitly removed from all queues.	USHORT	2
LocalConnectionState	The local end state of the connection. For more information, see LocalConnectionState (LCS) Values, on page 193.	USHORT	2
EventCause	Indicates a reason or explanation for the event occurrence. See Call EventCause (CEC) Values, on page 193.	USHORT	2

Table 98: CALL_DEQUEUED_EVENT Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The device ID between the call and the device.	STRING	64
QueueDeviceID$_{[45]}$ (required)	The device ID of the queuing device. For Unified CCX this is “”.	STRING	64
CSQID$_{[62]}$ (required)	The Contact Service Queue ID of the call.	UINT	4
reserved$_{[63]}$	This value is set to 0xffffffff.	UINT	4
reserved$_{[64]}$	This value is set to 0.	USHORT	2

RTP_STARTED_EVENT (OPTIONAL)

The RTP_STARTED_EVENT message indicates that an RTP (Real Time Protocol) media stream has been started. There are two media streams for audio media (one for the caller and one for the receiver). So there are two RTP Started events, one indicating the input has started (that is, the phone is listening) and the other that the output has started (that is, the outgoing media from the agent phone has begun).

The RTP Started event generally comes up at the same time as the established event. It also occurs when a call is retrieved from being on hold, and when the transfer or conference operations are completed.

There is no guarantee of the order of the RTP started events in relationship to the established and retrieved events. The RTP started events can occur before or after the established event.

Tthe following tables define the format of the RTP_STARTED_EVENT message.

Table 99: RTP_STARTED_EVENT (OPTIONAL) Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
clientPort	The TCP/IP port number of the client connection.	UINT	4
Direction	The direction of the event. One of the following values: 0: Input1: Output2: Bi-directional.	USHORT	2
RTPType	The type of the RTP event. One of the following values: 0: Audio1: Video2: Data	USHORT	2
BitRate	The media bit rate, used for a G.723 payload only: RTPBitRateR5_3: 1RTPBitRateR6_4: 2	UINT	4
EchoCancellation	0: off1: on	USHORT	2
PacketSize	The packet size in milliseconds	UINT	4
PayloadType	The audio codec type. See Audio Codec Type Values, on page 196.	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4

Table 100: RTP_STARTED_EVENT (OPTIONAL) Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The device ID between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format will apply to this field.	STRING	64
clientAddress [81] (required)	The IP address of the client.	STRING	16
AgentExtension[4] (optional)	The agent's IP phone extension	STRING	16
AgentInstrument[6] (optional)	The agent's IP phone number	STRING	64
SendingAddress[125] (optional)	The IP Address to which the client is sending the RTP stream.	STRING	16
SendingPort[126] (optional)	The UDP (User Datagram Protocol) port number to which the client is sending the RTP Stream.	UINT	4
AgentID[194]	The agent's Unified CCX login.	STRING	32

RTP_STOPPED_EVENT (OPTIONAL)

The RTP_STOPPED_EVENT message indicates that an RTP media has been stopped. There are two media streams for audio media so there are two RTP Stopped events, one indicating the input has stopped (that is, the phone is not listening) and the other that the output has stopped (that is, the outgoing media from the agent phone has stopped).

The RTP Stopped Event is received when the call is placed on hold and when the call disconnects.

The following tables define the format of the RTP_STOPPED_EVENT message.

Table 101: RTP_STOPPED_EVENT (OPTIONAL) Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 1.	UINT	4
clientPort	The TCP/IP port number of the client connection that was closed.	UINT	4

Fixed part Field name	Value	Data type	Byte size
Direction	The direction of the event. One of the following values: 0: Input 1: Output2: Bi-directional	USHORT	2
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to this call by Unified CCX.	UINT	4

Table 102: RTP_STOPPED_EVENT (OPTIONAL) Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The device ID between the call and the device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format will apply to this field.	STRING	64
ClientAddress$_{[81]}$ (required)	The IP address of the client.	STRING	16
AgentExtension$_{[4]}$ (optional)	The agent's ACD IP phone extension.	STRING	16
AgentInstrument$_{[6]}$ (optional)	The agent's IP phone number.	STRING	64
SendingAddress$_{[125]}$ (optional)	The IP Address to which the client was sending the RTP stream.	STRING	16
SendingPort$_{[126]}$ (optional)	The UDP (User Datagram Protocol) port number to which the client was sending the RTP Stream.	UINT	4
AgentID$_{[194]}$	The agent's Unified CCX login.	STRING	32

Call-Control (Client-Control) Messages

The call-control messages are from client applications requesting changes to agent states or establishing, answering, controlling, or terminating calls on behalf of a specified agent phone number, or manipulating the telephone features associated with an agent's IP phone.

These messages evoke a message response from Unified CCX. Consequently the messages are paired request/response messages. A request message for the desired control action is sent, and the outcome of the request is indicated by the type of response message that is received. Depending on the specifics of the request, as much as 10 or 15 seconds might elapse before the response message is returned. A successfully executed request is indicated by the corresponding control-action confirmation message, while an unsuccessful request is indicated by a CONTROL_FAILURE_CONF message.

This section includes the following message definitions:

- CONTROL_FAILURE_CONF, on page 139
- ALTERNATE_CALL_REQ, on page 139
- ALTERNATE_CALL_CONF, on page 141
- ANSWER_CALL_REQ, on page 141
- ANSWER_CALL_CONF, on page 142
- CLEAR_CALL_REQ, on page 142
- CLEAR_CALL_CONF, on page 143
- CLEAR_CONNECTION_REQ, on page 143
- CLEAR_CONNECTION_CONF, on page 144
- CONFERENCE_CALL_REQ, on page 144
- CONFERENCE_CALL_CONF, on page 146
- CONSULT_CALL_REQ, on page 148
- CONSULT_CALL_CONF, on page 150
- HOLD_CALL_REQ, on page 150
- HOLD_CALL_CONF, on page 151
- MAKE_CALL_REQ, on page 151
- MAKE_CALL_CONF, on page 153
- RECONNECT_CALL_REQ, on page 154
- RECONNECT_CALL_CONF, on page 155
- RETRIEVE_CALL_REQ, on page 155
- RETRIEVE_CALL_CONF, on page 156
- TRANSFER_CALL_REQ, on page 156
- TRANSFER_CALL_CONF, on page 158
- SEND_DTMF_SIGNAL_REQ, on page 159
- SEND_DTMF_SIGNAL_CONF, on page 160
- SET_CALL_DATA_REQ, on page 160
- SET_CALL_DATA_CONF, on page 162
- SUPERVISE_CALL_REQ

- SUPERVISE_CALL_CONF, on page 164
- SUPERVISOR_ASSIST_REQ, on page 165
- SUPERVISOR_ASSIST_CONF, on page 166
- SUPERVISOR_ASSIST_EVENT, on page 167
- BAD_CALL_REQ, on page 167
- BAD_CALL_CONF, on page 168

CONTROL_FAILURE_CONF

The CONTROL_FAILURE_CONF message, defined in the following tables, confirms that the previously requested control-service function identified by the given invokeID was unsuccessful. This message is sent in place of the corresponding confirmation message for the requested control-service function.

Table 103: CONTROL_FAILURE_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
FailureCode	One of the values specifying the reason that the request failed. See Control Failure (CF) Values, on page 198.	USHORT	2
Unified CCX ErrorCode	Unified CCX detailed error data, if available. Otherwise, 0. See Unified CCX ErrorCode Values.	UINT	4

Table 104: CONTROL_FAILURE_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$Text_{[7]}$	Text describing the error.	STRING	255

ALTERNATE_CALL_REQ

The ALTERNATE_CALL_REQ message, defined in the following tables, requests the double action of placing an active call on hold and then either retrieving a previously held call or answering an alerting call at the same device.

Note When specifying an alerting call, since there is no formal connection between a call and an alerting device, the ConnectionDeviceID of the calling connection is used here (as given in the CALL_DELIVERED_EVENT message).

Table 105: ALTERNATE_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
ActiveCallID	The Call ID value assigned to the currently active call by Unified CCX.	UINT	4
OtherCallID	The Call ID value assigned to the other call by Unified CCX.	UINT	4
ActiveConnectionDeviceType	The device ID type supplied in the ActiveConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
OtherConnectionDeviceType	The device ID type supplied in the OtherConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 106: ALTERNATE_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ActiveConnectionDeviceID[50] (required)	The ID of the currently active call connection.	STRING	64
OtherConnectionDeviceID[52] (required)	The ID of the other call connection.	STRING	64
AgentInstrument[6] (optional)	The agent's IP phone number.	STRING	64

ALTERNATE_CALL_CONF

The ALTERNATE_CALL_CONF message, defined in the following table, confirms the processing completion of the Alternate Call request.

Table 107: ALTERNATE_CALL_CONF Fixed Part Message Body Format

Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

ANSWER_CALL_REQ

The ANSWER_CALL_REQ message, defined in the following tables, allows a client to connect an alerting call at the device which is alerting.

Note Since there is no formal connection between a call and an alerting device, the ConnectionDeviceID of the calling connection is used here (as given in the CALL_DELIVERED_EVENT message).

Table 108: ANSWER_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
CallID	The Call ID value assigned to the call by Unified CCX. Can contain the special value 0xffffffff when alerting callID is not provided.	UINT	4
ConnectionDeviceType	The device ID type supplied in the ConnectionDeviceID floating field. Set this field to CONNECTION_ID_NONE when the alerting callID is not provided. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 109: ANSWER_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (optional)	The ID of the connection between the call and the device. Either ConnectionDeviceID or AgentInstrument must be specified.	STRING	64
AgentInstrument$_{[6]}$ (optional)	The IP phone number that answers the call. Either ConnectionDeviceID or AgentInstrument must be specified.	STRING	64

ANSWER_CALL_CONF

The ANSWER_CALL_CONF message, defined in the following table, confirms successful completion of the Answer Call request.

Table 110: ANSWER_CALL_CONF Fixed Part Message Body Format

Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

CLEAR_CALL_REQ

The CLEAR_CALL_REQ message, defined in the following tables, allows a client to release all devices from the specified call without regard to the number of other call parties.

Note Most applications use the CLEAR_CONNECTION_REQ message to avoid inadvertent clearing of all conference parties when dropping from a conference call.

Table 111: CLEAR_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4

Fixed part Field name	Value	Data type	Byte size
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 112: CLEAR_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The ID of the connection between the call and the device.	STRING	64
AgentInstrument[6] (optional)	The agent's IP phone number.	STRING	64

CLEAR_CALL_CONF

The CLEAR_CALL_CONF message, defined in the following table, confirms the processing completion of the Clear Call request.

Table 113: CLEAR_CALL_CONF Field Name Message Body Format

Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

CLEAR_CONNECTION_REQ

The CLEAR_CONNECTION_REQ message, defined in the following tables, allows a client to release a specific device connection from a designated call.

If only one party remains connected to the call following this request, the remaining connection is cleared and the call is terminated.

Table 114: CLEAR_CONNECTION_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	An ID for this request message that is returned in the corresponding confirm message.	UINT	4
reserved	Set this value to 1.	UINT	4
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 115: CLEAR_CONNECTION_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The ID of the device connection that is to be released.	STRING	64
AgentInstrument$_{[6]}$ (optional)	The phone number of the agent's IP phone whose connection is to be released.	STRING	64

CLEAR_CONNECTION_CONF

The CLEAR_CONNECTION_CONF message, defined in the following table, confirms the processing completion of the Clear Connection request.

Table 116: CLEAR_CONNECTION_CONF Field Name Message Body Format

Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

CONFERENCE_CALL_REQ

The CONFERENCE_CALL_REQ message, defined in the following tables, allows a client to conference an existing held call with another active call.

The two calls are merged and the two connections at the conferencing device are in the connected state.

Table 117: CONFERENCE_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	An ID for this request message that is returned in the corresponding confirm message.	UINT	4
reserved	Set this value to 1.	UINT	4
HeldCallID	The Call ID value assigned to the call held by Unified CCX.	UINT	4
ActiveCallID	The Call ID value assigned to the active call by Unified CCX.	UINT	4
HeldConnectionDeviceType	The type device ID supplied in the HeldConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
ActiveConnectionDeviceType	The type device ID supplied in the ActiveConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	BOOL	2
reserved	Set this value to 0.	BOOL	2
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2

Table 118: CONFERENCE_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ActiveConnectionDeviceID$_{[50]}$ (required)	The ID of the active call connection.	STRING	64
HeldConnectionDeviceID$_{[53]}$ (optional)	The ID of the held call connection. Either a HeldConnectionDeviceID or DialedNumber is required.	STRING	64
AgentInstrument$_{[6]}$ (required)	The agent's IP phone number.	STRING	64
DialedNumber$_{[11]}$ (optional)	The number to be dialed. Either a HeldConnectionDeviceID or DialedNumber is required.	STRING	40
CallVariable1$_{[13]}$ (optional)	Call-related variable data.	STRING	40
CallVariable2$_{[14]}$. through CallVariable9$_{[21]}$ (optional)	…	…	…
CallVariable10$_{[22]}$ (optional)	Call-related variable data.	STRING	40
CallWrapupData$_{[46]}$ (optional)	Call-related wrapup data.	STRING	40
NamedVariable$_{[82]}$ (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray$_{[83]}$ (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252
AccountCode$_{[78]}$ (optional)	Account code used by CTI applications.	STRING	40

CONFERENCE_CALL_CONF

The CONFERENCE_CALL_CONF message, defined in the following tables, confirms successful completion of the Conference Call request.

Table 119: CONFERENCE_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
NewCallID	The Call ID value assigned to the resulting conference call by Unified CCX.	UINT	4
NewConnectionDeviceType	The connection type of the device supplied in the NewConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
NumParties	The number of active connections associated with this conference call, up to a maximum of 16. This value also indicates the number of ConnectedPartyCallID, ConnectedPartyDeviceType and ConnectedPartyDeviceID floating fields present in the floating part of the message.	USHORT	2
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2

Table 120: CONFERENCE_CALL_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
NewConnectionDeviceID[47] (required)	The ID of the connection between the call and the device.	STRING	64
ConnectedPartyCallID[39] (optional)	The Call ID value assigned to one of the conference call parties. There can be more than one ConnectedPartyCallID field in the message (see NumParties, above).	UINT	4
ConnectedPartyDeviceType[40] (optional)	The device ID type of the device supplied in the following ConnectedPartyDeviceID floating field. There can be more than one ConnectedPartyDeviceType field in the message (see NumParties, above).	USHORT	2
ConnectedPartyDeviceID[41] (optional)	The device ID of one of the conference call parties. There can be more than one ConnectedPartyDeviceID field in the message (see NumParties, above).	STRING	64

CONSULT_CALL_REQ

The CONSULT_CALL_REQ message, defined in the following tables, requests the combined action of placing an active call on hold and then making a new call.

By default, the call context data of the active call is used to initialize the context data of the consultation call. A client can override some or all of this original call context in the consultation call by providing the desired values in this request.

Table 121: CONSULT_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
ActiveCallID	The Call ID value assigned to the active call by Unified CCX.	UINT	4
ActiveConnectionDeviceType	The connection type of the device supplied in the ActiveConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
ConsultType	Use following values: 0: Unspecified 1: Transfer 2: Conference	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	BOOL	2
reserved	Set this value to 0.	BOOL	2

Fixed part Field name	Value	Data type	Byte size
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2

Table 122: CONSULT_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ActiveConnectionDeviceID$_{[50]}$ (required)	The device ID of the active call connection.	STRING	64
DialedNumber $_{[11]}$ (required)	The number to be dialed to establish the new call.	STRING	40
AgentInstrument$_{[6]}$ (optional)	The phone number of the agent's IP phone that initiates the new call.	STRING	64
CallVariable1$_{[13]}$ (optional)	Call-related variable data used in place of the corresponding variable from the active call.	STRING	40
CallVariable2$_{[14]}$.(optional) through CallVariable9$_{[21]}$ (optional)	…	…	…
CallVariable10$_{[22]}$ (optional)	Call-related variable data that used in place of the corresponding variable from the active call.	STRING	40
CallWrapupData$_{[46]}$ (optional)	Call-related wrapup data used in place of the corresponding data from the active call.	STRING	40
NamedVariable$_{[82]}$ (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray$_{[83]}$ (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252
AccountCode$_{[78]}$ (optional)	Account code used by CTI applications.	STRING	40

CONSULT_CALL_CONF

The CONSULT_CALL_CONF message, defined in the following tables, confirms successful completion of the Consult Call request:

Table 123: CONSULT_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
NewCallID	The Call ID value assigned to the resulting new call by Unified CCX.	UINT	4
NewConnectionDeviceType	The type of device ID supplied in the NewConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2

Table 124: CONSULT_CALL_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
NewConnectionDeviceID[47] (required)	The ID of the device connection associated with the new call.	STRING	64

HOLD_CALL_REQ

The HOLD_CALL_REQ message, defined in the following tables, allows a client to place an existing call connection into the held state.

Table 125: HOLD_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4

Fixed part Field name	Value	Data type	Byte size
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4
ConnectionDeviceType	The connection type of the device supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	Set this value to 0.	BOOL	2

Table 126: HOLD_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The ID of the connection between the call and the device.	STRING	64
AgentInstrument$_{[6]}$ (optional)	The agent's IP phone number.	STRING	64

HOLD_CALL_CONF

The HOLD_CALL_CONF message, defined in the following table, confirms successful completion of the Hold Call request.

Table 127: HOLD_CALL_CONF Field Part Message Body Format

Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

MAKE_CALL_REQ

The MAKE_CALL_REQ message, defined in the following tables, allows a client to initiate a call between two devices.

This request attempts to create a new call and establish a connection between the calling device (originator) and the called device (destination).

Table 128: MAKE_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	BOOL	2
reserved	Set this value to 0.	BOOL	2
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2
reserved (Version 14 and later)	Set this value to NULL_CSQ (0xFFFFFFFF).	UINT	4

Table 129: MAKE_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
AgentInstrument$_{[6]}$ (required)	The agent's IP phone number	STRING	64
DialedNumber $_{[11]}$ (required)	The number to be dialed to establish the new call.	STRING	40
CallVariable1$_{[13]}$ (optional)	Call-related variable data.	STRING	40

Floating part Field name	Value	Data type	Maximum size
CallVariable2[14]. (optional) through CallVariable9[21] (optional)	…	…	…
CallVariable10[22] (optional)	Call-related variable data.	STRING	40
CallWrapupData[46] (optional)	Call-related wrapup data.	STRING	40
NamedVariable[82] (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray[83] (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252
AccountCode[78] (optional)	Account code used by CTI applications.	STRING	40

MAKE_CALL_CONF

The MAKE_CALL_CONF message, defined in the following tables, confirms the processing completion of the Make Call request.

Table 130: MAKE_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
NewCallID	The Call ID value assigned to the call by Unified CCX.	UINT	4
NewConnectionDeviceType	The connection type of the device supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2

Table 131: MAKE_CALL_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
NewConnectionDeviceID[47] (required)	The device ID of the connection between the call and the device.	STRING	64

RECONNECT_CALL_REQ

The RECONNECT_CALL_REQ message, defined in the following tables, requests the combined action of clearing an active call and then retrieving an existing held call.

Table 132: RECONNECT_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
ActiveCallID	The Call ID value assigned to the currently active call by Unified CCX.	UINT	4
HeldCallID	The Call ID value assigned to the held call by Unified CCX.	UINT	4
ActiveConnectionDeviceType	The type of device ID supplied in the ActiveConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
HeldConnectionDeviceType	The type device ID supplied in the HeldConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 133: RECONNECT_CALL_REQ Foating Part Field Name Message Body Format

Floating part Field name	Value	Data type	Maximum size
ActiveConnectionDeviceID[50] (required)	The device identifier of the currently active call connection.	STRING	64
HeldConnectionDeviceID[53] (required)	The device identifier of the held call connection.	STRING	64

Floating part Field name	Value	Data type	Maximum size
AgentInstrument[6] (optional)	The agent's IP phone number.	STRING	64

RECONNECT_CALL_CONF

The RECONNECT_CALL_CONF message, defined in the following table, confirms the processing completion of the Reconnect Call message request.

Table 134: RECONNECT_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

RETRIEVE_CALL_REQ

The RETRIEVE_CALL_REQ message, defined in the following tables, allows a client to retrieve an existing held connection.

Table 135: RETRIEVE_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
HeldCallID	The Call ID value assigned to the held call by Unified CCX.	UINT	4
HeldConnectionDeviceType	The connection type of the device supplied in the HeldConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 136: RETRIEVE_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
HeldConnectionDeviceID[53] (required)	The device ID of the held call connection.	STRING	64
AgentInstrument[6] (optional)	The agent's IP phone number.	STRING	64

RETRIEVE_CALL_CONF

The RETRIEVE_CALL_CONF message, defined in the following table, confirms the processing completion of the Retrieve Call request.

Table 137: RETRIEVE_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

TRANSFER_CALL_REQ

The TRANSFER_CALL_REQ message, defined in the following tables, allows a client to transfer a held call with an active call at the same device.

This request merges the two calls with connections to a single common device. Both of the connections with the common device become NULL and their device IDs are released.

Table 138: TRANSFER_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
ActiveCallID	The Call ID value assigned to the currently active call by Unified CCX.	UINT	4
HeldCallID	The Call ID value assigned to the held call by Unified CCX.	UINT	4

Fixed part Field name	Value	Data type	Byte size
ActiveConnectionDeviceType	The connection type of the device supplied in the ActiveConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
HeldConnectionDeviceType	The connection type of the device supplied in the HeldConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	USHORT	2
reserved	Set this value to 0.	BOOL	2
reserved	Set this value to 0.	BOOL	2
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2

Table 139: TRANSFER_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ActiveConnectionDeviceID$_{[50]}$ (required)	The device ID of the currently active call connection.	STRING	64
HeldConnectionDeviceID$_{[53]}$ (optional)	The device ID of the held call connection. Either a HeldConnectionDeviceID or DialedNumber is required.	STRING	64
AgentInstrument$_{[6]}$ (required)	The agent's IP phone number.	STRING	64

Floating part Field name	Value	Data type	Maximum size
DialedNumber[11] (optional)	The telephone number dialed.	STRING	40
CallVariable1[13] (optional)	Call-related variable data.	STRING	40
CallVariable2[14]. (optional) through CallVariable9[21] (optional)	…	…	…
CallVariable10[22] (optional)	Call-related variable data.	STRING	40
CallWrapupData[46] (optional)	Call-related wrapup data.	STRING	40
NamedVariable[82] (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray[83] (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252
AccountCode[78] (optional)	Account code used by CTI applications.	STRING	40

TRANSFER_CALL_CONF

The TRANSFER_CALL_CONF message, defined in the following tables, confirms the processing completion of the Transfer Call request.

Table 140: TRANSFER_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
NewCallID	The Call ID value assigned to the resulting transferred call by Unified CCX.	UINT	4
NewConnectionDeviceType	The type of device ID supplied in the NewConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
NumParties	The number of active connections associated with this conference call, up to a maximum of 16.	USHORT	2

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0xffff.	USHORT	2
reserved	This value is set to 3.	USHORT	2

Table 141: TRANSFER_CALL_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
NewConnectionDeviceID[47] (required)	The device ID of the connection between the call and the device.	STRING	64

SEND_DTMF_SIGNAL_REQ

The SEND_DTMF_SIGNAL_REQ message, defined in the following tables, allows a client to have the ACD transmit a sequence of DTMF tones on behalf of a call party.

Table 142: SEND_DTMF_SIGNAL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4
ConnectionDeviceType	The connection type of the device supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
ToneDuration	The duration in milliseconds of the DTMF digit tones. A value of 0 can be used to select a default value. Can be ignored if Unified CCX is unable to alter the DTMF tone timing.	USHORT	2
PauseDuration	Specifies the duration in milliseconds of the DTMF inter-digit spacing. A value of 0 can be used to select a default value. Can be ignored if Unified CCX is unable to alter the DTMF tone timing.	UINT	4

Table 143: SEND_DTMF_SIGNAL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The ID of the connection between the call and the device.	STRING	64
DTMFString [67] (required)	The sequence of tones to be generated.	STRING	32
AgentInstrument[6] (optional)	The agent's IP phone number.	STRING	64

SEND_DTMF_SIGNAL_CONF

The SEND_DTMF_SIGNAL_CONF message, defined in the following table, confirms the processing completion of the send DTMF signal request.

Table 144: SEND_DTMF_SIGNAL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

SET_CALL_DATA_REQ

This message is sent by a client or Unified CCX to set one or more call variables and/or call wrap-up data. The combination of CallID, ConnectionDeviceType, and ConnectionDeviceID uniquely identify the call to be operated on. Variables not provided in the message are not affected. See also Call Context Data, on page 38.

The SET_CALL_DATA_REQ and SET_CALL_DATA_CONF messages are defined in the following table and in SET_CALL_DATA_CONF, on page 162.

Table 145: SET_CALL_DATA_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4

Fixed part Field name	Value	Data type	Byte size
reserved	Set this value to 1.	UINT	4
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2
CallType	The general classification of the call type. See CallType Values, page 9-13.	USHORT	2
CalledPartyDisposition	Indicates the disposition of the called party. See Disposition Values, page 9-19.	USHORT	2
reserved	Set this value to 0.	UINT	4
reserved	Set this value to 1.	UINT	4

Table 146: SET_CALL_DATA_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The ID of the connection between the call and the device.	STRING	64
ANI$_{[8]}$ (optional)	The calling line ID of the caller.	STRING	40
CallerEnteredDigits$_{[12]}$ (optional)	The digits entered by the caller in response to IVR prompting.	STRING	40
CallVariable1$_{[13]}$ (optional)	Call-related variable data.	STRING	40
CallVariable2$_{[14]}$. (optional) through CallVariable9$_{[21]}$ (optional)	…	…	…

Floating part Field name	Value	Data type	Maximum size
CallVariable10[22] (optional)	Call-related variable data.	STRING	40
CallWrapupData[46] (optional)	Call-related wrapup data.	STRING	40
NamedVariable[82] (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray[83] (optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252
CustomerAccountNumber[96] (optional)	Customer Account Number.	STRING	32

SET_CALL_DATA_CONF

The SET_CALL_DATA_CONF message confirms the processing completion of the SET_CALL_DATA_REQ message.

When the requested call variables have been updated, and the new values are guaranteed to remain set in the event that the CTI session is abnormally terminated, Unified CCX responds to the client that requested the update with the SET_CALL_DATA_CONF message.

Table 147: SET_CALL_DATA_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

SUPERVISE_CALL_REQ

At any time, for monitoring quality of service, training, and so on, a supervisor client can send a SUPERVISE_CALL_REQ message to Unified CCX to request barge-in or interception of a call. At the end of such call supervision, a supervisor client should send a SUPERVISE_CALL_REQ message with SUPERVISOR_CLEAR as the SupervisorAction value to disconnect the supervisor's device from the call. The SUPERVISE_CALL_REQ message, defined in the following tables, allows a supervisor client to supervise an agent's call, either through barge-in or interception.

Table 148: SUPERVISE_CALL_REQ Fixed Part Field Name Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
AgentCallID	The Call ID value assigned to the call by Unified CCX. This Call ID value is normally the Integer CallID on the agent's device.	UINT	4
SupervisorCallID	The Call ID value of the supervisor. If there is no supervisor call, this field must be set to 0xffffffff.	UINT	4
AgentConnectionDeviceType	The type of device ID supplied in the AgentConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
SupervisorConnectionDeviceType	The type of device ID supplied in the SupervisorConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
SupervisoryAction	One of the values from Table 3 specifying the desired call supervision operation.	USHORT	2

Table 149: SUPERVISE_CALL_REQ Floating Part Field Name Message Body Format

Floating part Field name	Value	Data type	Maximum size
AgentConnectionDeviceID[90]	The identifier of the connection of the agent call and the agent's device. Either the CallID and the ConnectionDeviceID, or one of AgentExtension, AgentID, or AgentInstrument must be provided.	STRING	64
SupervisorConnectionDeviceID[91]	The identifier of the connection of the supervisor call and the supervisor's device. Either the CallID and the ConnectionDeviceID, or one of AgentExtension, AgentID, or AgentInstrument must be provided.	STRING	64

Floating part Field name	Value	Data type	Maximum size
AgentExtension$_{[4]}$	The agent's IP phone extension. Either CallID and ConnectionDeviceID, or one of AgentExtension, AgentID, or AgentInstrument must be provided.	STRING	16
AgentInstrument$_{[6]}$ (optional)	The agent's IP phone number.	STRING	64
SupervisorInstrument$_{[85]}$	The supervisor's IP phone number.	STRING	64
AgentID$_{[194]}$ (optional)	The agent's login ID	STRING	32

Supervisory Action Values

Table 150: Supervisory Action Values

Supervisory Action	Description	Value
SUPERVISOR_MONITOR	The supervisor device is to be connected to the call for silent monitoring. This allows the supervisor to hear all parties participating in the ICD call.	1
SUPERVISOR_BARGE_IN	The supervisor device is to be connected to the call as an active participant. This allows the supervisor to speak to all parties participating in the call, as in a conference.	3
SUPERVISOR_INTERCEPT	The supervisor device is to be connected to the call as an active participant and the agent connection will be dropped.	4

SUPERVISE_CALL_CONF

The SUPERVISE_CALL_CONF message, defined in the following tables, confirms the processing completion of the SUPERVISE_CALL_REQ message.

Table 151: SUPERVISE_CALL_CONF Fixed Part Field Name Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4

Fixed part Field name	Value	Data type	Byte size
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 152: SUPERVISE_CALL_CONF Floating Part Field Name Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The identifier of the connection between the call and the agent device that is being supervised.	STRING	64

SUPERVISOR_ASSIST_REQ

When an agent needs supervisor assistance, an agent can send a SUPERVISOR_ASSIST_REQ message to Unified CCX asking for assistance from a team supervisor.

Once an available supervisor is found, a call with calltype SUPERVISOR_ASSIST is initiated and a SUPERVISOR_ASSIST_CONF is sent to the requesting client. If no supervisor can be found a FAILURE_CONF response is returned to the requesting client.

The SUPERVISOR_ASSIST_REQ message, defined in the following tables, allows a client to notify the client agent's supervisor that assistance with the indicated call is required.

Table 153: SUPERVISOR_ASSIST_REQ Fixed Part Field Name Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
CallID	The Call ID value of the call with which the agent needs assistance. Can contain the special value 0xffffffff when there is no related call.	UINT	4
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 154: SUPERVISOR_ASSIST_REQ Floating Part Field Name Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The identifier of the connection between the call and the agent's device.	STRING	64
AgentExtension$_{[4]}$	The agent's IP phone extension.	STRING	16
AgentInstrument$_{[6]}$ (optional)	The agent's IP phone number	STRING	64
AgentID$_{[194]}$ (optional)	The agent's Unified CCX login.	STRING	32

SUPERVISOR_ASSIST_CONF

The SUPERVISOR_ASSIST_CONF message, defined in the following tables, confirms the processing completion of the SUPERVISOR_ASSIST_REQ message.

Table 155: SUPERVISOR_ASSIST_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
CallID	The Call ID value assigned to the resulting SupervisorAssist call by Unified CCX.	UINT	4
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	Set this value to 0xffff	USHORT	2
reserved	Set this value to 0.	USHORT	2

Table 156: SUPERVISOR_ASSIST_CONF Floating Part Field Name Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The identifier of the device connection associated with the new call.	STRING	64

SUPERVISOR_ASSIST_EVENT

The SUPERVISOR_ASSIST_EVENT message, defined in the following tables, indicates that a client requested supervisor assistance.

Table 157: SUPERVISOR_ASSIST_EVENT Fixed Part Field Name Message Body Format

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 1.	UINT	4
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4
ConnectionDeviceType	The type of device ID supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
reserved	This value is set to 0.	UINT	4

Table 158: SUPERVISOR_ASSIST_EVENT Floating Part Field Name Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The identifier of the connection between the call and the agent's device. In CTI Protocol Version 14, the general rules for the Primary.Actual Field Format will apply to this field.	STRING	64
ClientID[1] (required)	The ID of the client making the notification.	STRING	64
Text$_{[7]}$	Text describing the assistance requested.	STRING	255

BAD_CALL_REQ

The BAD_CALL_REQ message, defined in the following tables, allows a CTI client to notify Unified CCX of the bad quality of a call.

Given this information, Unified CCX can log information on the call that can help diagnose the issue.

Table 159: BAD_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
ConnectionDeviceType	The connection type of the device supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallID	The Call ID value of the call that the agent needs assistance with. Can contain the special value 0xffffffff when there is no related call.	UINT	4

Table 160: BAD_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID$_{[25]}$ (required)	The ID of the connection between the call and the agent's device.	STRING	64
AgentID$_{[194]}$ (optional)	The agent's Unified CCX login.	STRING	32

BAD_CALL_CONF

The BAD_CALL_CONF message, defined in the following table, confirms the processing completion of the BAD_CALL_REQ message.

Table 161: BAD_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

Miscellaneous Messages

This section contains the following message definitions:

- QUERY_QUEUE_STATISTICS_REQ, on page 169
- QUERY_QUEUE_STATISTICS_CONF, on page 169
- QUERY_AGENT_QUEUE_STATISTICS_REQ, on page 172
- QUERY_AGENT_QUEUE_STATISTICS_CONF, on page 172
- QUERY_DEVICE_INFO_REQ, on page 173
- QUERY_DEVICE_INFO_CONF, on page 174
- QUERY_SUMMARY_STATISTICS_REQ, on page 175
- QUERY_SUMMARY_STATISTICS_CONF, on page 175
- SNAPSHOT_CALL_REQ, on page 177
- SNAPSHOT_CALL_CONF, on page 178
- SNAPSHOT_DEVICE_REQ, on page 181
- SNAPSHOT_DEVICE_CONF, on page 181

QUERY_QUEUE_STATISTICS_REQ

The QUERY_AGENT_QUEUE_STATISTICS_REQ message, defined in the following tables, allows a CTI client to obtain the current call handling statistics for one of the client agent's CSQs. To avoid impacting system performance, clients should not request queue statistics too frequently. Depending on the needs of the client application, updating queue statistics after each call is handled can be appropriate.

Table 162: QUERY_QUEUE_STATISTICS_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4

Table 163: QUERY_QUEUE_STATISTICS_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$CSQID_{[62]}$	The ID of the Contact Service Desk.	INT	4

QUERY_QUEUE_STATISTICS_CONF

The QUERY_QUEUE_STATISTICS_CONF message, defined in the following tables, confirms the processing completion of the QUERY_AGENT_QUEUE_STATISTICS_REQ message.

Today values represent statistics accumulated since StartTime as indicated in the message. Call counts and times are updated when any after-call work for the call is completed (calls currently in progress are not included in the statistics).

Table 164: QUERY_QUEUE_STATISTICS_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
Real-Time Statistics			
LoggedInAgents	The number of agents currently belonging to the CSQ who are currently logged in.	USHORT	2
InSessionAgents	The number of agents currently belonging to the CSQ who are currently In-Session (talking state).	USHORT	2
AvailableAgents	The number of agents currently available (ready state).	USHORT	2
UnAvailableAgents	The number of agents currently belonging to the CSQ who are currently Unavailable (not ready state).	USHORT	2
InWorkAgents	The number of agents currently belonging to the CSQ who are currently in Work state (work not ready state).	USHORT	2
SelectedAgents	The number of agents currently belonging to the CSQ who are currently in Selected state (reserved state).	USHORT	2
CallsInPriorityQueue1	The total number of calls that are in the priority queue 1 of CSQ.	UINT	4
CallsInPriorityQueue2	The total number of calls that are in the priority queue 2 of CSQ.	UINT	4
CallsInPriorityQueue3	The total number of calls that are in the priority queue 3 of CSQ.	UINT	4
CallsInPriorityQueue4	The total number of calls that are in the priority queue 4 of CSQ.	UINT	4
CallsInPriorityQueue5	The total number of calls that are in the priority queue 5 of CSQ.	UINT	4
CallsInPriorityQueue6	The total number of calls that are in the priority queue 6 of CSQ.	UINT	4

Fixed part Field name	Value	Data type	Byte size
CallsInPriorityQueue7	The total number of calls that are in the priority queue 7 of CSQ.	UINT	4
CallsInPriorityQueue8	The total number of calls that are in the priority queue 8 of CSQ.	UINT	4
CallsInPriorityQueue9	The total number of calls that are in the priority queue 9 of CSQ.	UINT	4
CallsInPriorityQueue10	The total number of calls that are in the priority queue 10 of CSQ.	UINT	4
Today (from StartTime until EndTime) statistics			
StartTime	The Date and Time that the following counters have started accumulating.	TIME	4
EndTime	The Date and Time that the following counters ended.	TIME	4
TotalCalls	The total number of calls that were associated to this CSQ in the current day.	UINT	4
OldestCallInQueue	The elapsed wait time for the oldest call currently in queue.	UINT	4
HandledCallsToday	The total number of calls that were handled by agents belonging to the CSQ in the current day.	UINT	4
CallsAbandoned	The total number of calls that were associated with the CSQ and were abandoned by the callers before connecting to an agent.	UINT	4
CallsDequeued	The total number of calls that were dequeued from the CSQ queue.	UINT	4
AverageTalkDuration	The average talk duration for the calls coming in to this CSQ.	UINT	4
AverageWaitDuration	The average wait time for a call before it gets connected to an agent.	UINT	4
LongestTalkDuration	The longest time that a caller was talking to an agent (for this CSQ).	UINT	4
LongestWaitDuration	The longest wait time before a caller got connected to an agent.	UINT	4

Table 165: QUERY_QUEUE_STATISTICS_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$CSQID_{[62]}$	The ID of the Contact Service Desk.	INT	4

QUERY_AGENT_QUEUE_STATISTICS_REQ

The QUERY_AGENT_QUEUE_STATISTICS_REQ message allows a CTI client to obtain the summary queue statistics for an agent for all the CSQs to which the agent belongs. To avoid impacting system performance, clients should not request agent statistics too frequently. Depending on the needs of the client application, updating agent statistics after each call is handled may be appropriate.

Table 166: QUERY_AGENT_QUEUE_STATISTICS_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4

Table 167: QUERY_AGENT_QUEUE_STATISTICS_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$AgentID_{[194]}$ (required)	The Agent ID.	STRING	32

QUERY_AGENT_QUEUE_STATISTICS_CONF

The QUERY_AGENT_QUEUE_STATISTICS_CONF message, defined in the following tables, confirms the completion of the QUERY_AGENT_QUEUE_STATISTICS_REQ message.

Table 168: QUERY_AGENT_QUEUE_STATISTICS_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
Real-Time Statistics			

Fixed part Field name	Value	Data type	Byte size
CallsInQueue	The sum of the total number of calls in queue for all the CSQs that this Agent belongs to.	UINT	4
OldestCallInQueue	The elapsed wait time of the oldest call currently in queue for any of the CSQs to which the Agent belongs.	UINT	4

Table 169: QUERY_AGENT_QUEUE_STATISTICS_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$AgentID_{[194]}$ (required)	The Agent ID.	STRING	

QUERY_DEVICE_INFO_REQ

The QUERY_DEVICE_INFO_REQ message, defined in the following tables, allows a client to retrieve general information about a specified device.

Table 170: QUERY_DEVICE_INFO_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
reserved	Set this value to 0.	USHORT	2

Table 171: QUERY_DEVICE_INFO_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$AgentInstrument_{[6]}$ (required)	The agent IP phone number.	STRING	64

QUERY_DEVICE_INFO_CONF

The QUERY_DEVICE_INFO_CONF message, defined in the following tables, confirms the processing completion of the Query Device Info request.

Table 172: QUERY_DEVICE_INFO_CONF Field Name Message Body Format

Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
reserved	This value is set to 21.	USHORT	2
reserved	This value is set to -1.	USHORT	2
reserved	This value is set to -1.	USHORT	2
reserved	This value is set to 0.	USHORT	2
reserved	This value is set to 0.	USHORT	2
MaxActiveCalls	The maximum number of concurrent calls that can be active at the device.	USHORT	2
MaxHeldCalls	The maximum number of concurrent calls that can be held at the device. Set to 0xFFFF if unknown or unavailable.	USHORT	2
MaxDevicesInConference	The maximum number of devices that can participate in conference calls at the device. Set to 0xFFFF if unknown or unavailable.	USHORT	2
MakeCallSetup	A bitwise combination of Agent State masks in which a MAKE_CALL_REQ message can be initiated.	UINT	4
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 0.	UINT	4
reserved	This value is set to 0.	UINT	4

Table 173: QUERY_DEVICE_INFO_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
reserved[70]	This value is set to 0xffff.	USHORT	2
reserved[71]	This value is set to 3.	USHORT	2

QUERY_SUMMARY_STATISTICS_REQ

The QUERY_SUMMARY_STATISTICS_REQ message, defined in the following table, allows a CTI client to obtain system summary statistics for Unified CCX. To avoid impacting system performance, clients should not request statistics too frequently.

Table 174: QUERY_SUMMARY_STATISTICS_REQ Field Name Message Body Format

Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4

QUERY_SUMMARY_STATISTICS_CONF

The QUERY_SUMMARY_STATISTICS_CONF message, defined in the following table, confirms the processing completion of the QUERY_SUMMARY_STATISTICS_REQ message.

Today values represent statistics accumulated since StartTime as indicated in the message. Call counts and times are updated when any after-call work for the call is completed (calls currently in progress are not included in the statistics).

Table 175: QUERY_SUMMARY_STATISTICS_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
Resource Statistics			
NumQueues	The number of CSQs configured in the Unified CCX system.	USHORT	2
LoggedInAgents	The number of agents who are currently logged in	USHORT	2

Fixed part Field name	Value	Data type	Byte size
InSessionAgents	The number of agents who are currently In-Session.	USHORT	2
AvailableAgents	The number of agents who are currently available.	USHORT	2
UnAvailableAgents	The number of agents who are currently Unavailable.	USHORT	2
InWorkAgents	The number of agents who are currently in Work state.	USHORT	2
SelectedAgents	The number of agents who are currently in Selected state.	USHORT	2
CallsInPriorityQueue1	The total number of calls that are in priority queue 1 for all CSQs.	UINT	4
CallsInPriorityQueue2	The total number of calls that are in priority queue 2 for all CSQs.	UINT	4
CallsInPriorityQueue3	The total number of calls that are in priority queue 3 for all CSQs.	UINT	4
CallsInPriorityQueue4	The total number of calls that are in priority queue 4 for all CSQs.	UINT	4
CallsInPriorityQueue5	The total number of calls that are in priority queue 5 for all CSQs.	UINT	4
CallsInPriorityQueue6	The total number of calls that are in priority queue 6 of all CSQs.	UINT	4
CallsInPriorityQueue7	The total number of calls that are in priority queue 7 for all CSQs.	UINT	4
CallsInPriorityQueue8	The total number of calls that are in priority queue 8 for all CSQs.	UINT	4
CallsInPriorityQueue9	The total number of calls that are in priority queue 9 for all CSQs.	UINT	4
CallsInPriorityQueue10	The total number of calls that are in priority queue 10 for all CSQs.	UINT	4
Today (from StartTime until EndTime) Statistics			
StartTime	The Date and Time that the following counters have started accumulating.	TIME	4

Fixed part Field name	Value	Data type	Byte size
EndTime	The Date and Time that the following counters ended.	TIME	4
TotalCalls	The total number of calls in the current day	UINT	4
OldestCallInQueue	The time that the oldest call currently in queue has been waiting.	UINT	4
HandledCallsToday	The total number of calls that were handled in the current day.	UINT	4
CallsAbandoned	The total number of calls that were abandoned by the callers before connecting to an agent.	UINT	4
AverageTalkDuration	The average talk duration.	UINT	4
AverageWaitDuration	The average wait time for a call before it gets connected to an agent	UINT	4
LongestTalkDuration	The longest time that a caller was talking to an agent	UINT	4
LongestWaitDuration	The longest wait time before a caller got connected to an agent.	UINT	4

SNAPSHOT_CALL_REQ

The SNAPSHOT_CALL_REQ message, defined in the following tables, allows a client to retrieve information on a specified call.

The call information provided consists of a list of the associated devices and the connection state for each device.

Table 176: SNAPSHOT_CALL_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
CallID	The Call ID value assigned to the call by Unified CCX.	UINT	4

Fixed part Field name	Value	Data type	Byte size
ConnectionDeviceType	The connection type of the device supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	USHORT	2

Table 177: SNAPSHOT_CALL_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ConnectionDeviceID[25] (required)	The identifier of the connection between the call and the device.	STRING	64

SNAPSHOT_CALL_CONF

The SNAPSHOT_CALL_CONF message, defined in the following tables, confirms the processing completion of the requested Snapshot Call data.

Table 178: SNAPSHOT_CALL_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4
CallType	This value is set to 0.	USHORT	2
reserved	This value is set to 0.	USHORT	2
NumCallDevices	The number of active devices associated with this call, up to a maximum of 16. This value also indicates the number of CallCallID, CallConnectionDeviceType, CallConnectionDeviceID, CallDeviceType, CallDeviceID, and CallDeviceConnectionState floating fields present in the floating part of the message.	USHORT	2
NumNamedVariables	The number of NamedVariable floating fields present in the floating part of the message.	USHORT	2
NumNamedArrays	The number of NamedArray floating fields present in the floating part of the message.	USHORT	2

Fixed part Field name	Value	Data type	Byte size
reserved	This value is set to 0.	USHORT	2
reserved (Only available in CTI Protocol version 11 and later.)	This value is set to 0.	INT	4

Table 179: SNAPSHOT_CALL_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
ANI$_{[8]}$ (optional)	The calling line ID of the caller.	STRING	40
DNIS$_{[10]}$ (optional)	The DNIS provided with the call.	STRING	32
DialedNumber$_{[11]}$(optional)	The number dialed.	STRING	40
CallerEnteredDigits$_{[12]}$ (optional)	The digits entered by the caller in response to IVR prompting.	STRING	40
reserved$_{[72]}$ (required)	This value is set to 0.	UINT	4
reserved $_{[73]}$ (required)	This value is set to 0.	UINT	4
CallVariable1$_{[13]}$ (optional)	Call-related variable data.	STRING	40
CallVariable2$_{[14]}$. (optional) through CallVariable9$_{[21]}$ (optional)	…	…	…
CallVariable10$_{[22]}$ (optional)	Call-related variable data.	STRING	40
CallWrapupData$_{[46]}$ (optional)	Call-related wrapup data.	STRING	40
NamedVariable$_{[82]}$ (optional)	Call-related variable data that has a variable name defined in the Unified CCX Editor. See NAMEDVARIABLE Data Format, on page 185, for the format of this field.	NAMEDVAR	251
NamedArray$_{[83]}$(optional)	Call-related variable data that has an array variable name defined in the Unified CCX Editor. See NAMEDARRAY Data Format, on page 186, for the format of this field.	NAMEDARRAY	252

Floating part Field name	Value	Data type	Maximum size
CallConnectionCallID[56] (optional)	The Call ID value assigned to one of the call device connections. There can be more than one CallCallID field in the message (see NumCallDevices, above).	UINT	4
CallConnectionDeviceType[57] (optional)	The connection type of the device supplied in the following CallConnectionDeviceID floating field. There can be more than one CallConnectionDeviceType fields in the message (see NumCallDevices, above). This field always immediately follows the corresponding CallConnectionCallID field. See ConnectionDeviceType Values, on page 196.	USHORT	2
CallConnectionDeviceID[58] (optional)	The device ID of one of the call connections. There can be more than one CallConnectionDeviceID field in the message (see NumCallDevices, above). This field always immediately follows the corresponding CallConnectionDeviceType field. See ConnectionDeviceType Values, on page 196.	STRING	64
CallDeviceType[59] (optional)	The device ID type of the device supplied in the following CallDeviceID floating field. See DeviceType Values, on page 192. There can be more than one CallDeviceType field in the message (see NumCallDevices, above). This field always immediately follows the corresponding CallConnectionDeviceID field.	USHORT	2
CallDeviceID[60] (optional)	The device ID of the subject device. There can be more than one CallDeviceID field in the message (see NumCallDevices, above). This field always immediately follows the corresponding CallDeviceType field.	STRING	64
CallDeviceConnectionState[61] (optional)	The local connection state of one of the call device connections. There can be more than one CallDeviceConnectionState field in the message (see NumCallDevices, above). This field always immediately follows the corresponding CallDeviceID field.	USHORT	2

SNAPSHOT_DEVICE_REQ

The SNAPSHOT_DEVICE_REQ message, defined in the following tables, allows a client to retrieve information on a specified device.

The device information provided consists of a list of the calls associated with the device and the current state of each call.

Table 180: SNAPSHOT_DEVICE_REQ Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	A unique ID generated by the CTI client for each request message. This ID is returned in the corresponding confirmation message.	UINT	4
reserved	Set this value to 1.	UINT	4
SnapshotDeviceType	For non-agent devices this indicates the type of the device specified in the following AgentInstrument floating field.	USHORT	2

Table 181: SNAPSHOT_DEVICE_REQ Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
$AgentInstrument_{[6]}$ (required)	The agent's IP phone number or any valid phone number.	STRING	64

SNAPSHOT_DEVICE_CONF

The SNAPSHOT_DEVICE_CONF message, defined in the following tables, confirms the requested completion of the Snapshot Device data request.

Table 182: SNAPSHOT_DEVICE_CONF Fixed Part Message Body Format

Fixed part Field name	Value	Data type	Byte size
InvokeID	Set to the same value as the InvokeID from the corresponding request message.	UINT	4

Fixed part Field name	Value	Data type	Byte size
NumCalls	The number of active calls associated with this device, up to a maximum of 16. This value also indicates the number of CallConnectionCallID, CallConnectionDeviceType, CallConnectionDeviceID, and CallState floating fields present in the floating part of the message.	USHORT	2

Table 183: SNAPSHOT_DEVICE_CONF Floating Part Message Body Format

Floating part Field name	Value	Data type	Maximum size
CallConnectionCallID[56] (optional)	The Call ID value assigned to one of the calls. There can be more than one CallConnectionCallID field in the message (see NumCalls, above).	UINT	4
CallConnectionDeviceType[57] (optional)	The connection type of the device supplied in the following CallConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196. There can be more than one CallConnectionDeviceType field in the message (see NumCalls, above). This field always immediately follows the corresponding CallConnectionCallID field.	USHORT	2
CallConnectionDeviceID[58] (optional)	Indicates the connection type supplied in the ConnectionDeviceID floating field. See ConnectionDeviceType Values, on page 196.	STRING	64
CallDeviceConnectionState[61] (optional)	The active state of the call (LocalConnectionState (LCS) Values, on page 193). There can be more than one CallState field in the message (see NumCalls, above). This field always immediately follows the corresponding CallConnectionDeviceID field.	USHORT	2

CHAPTER 10

Data Types and Message Constants

This reference chapter includes the following miscellaneous message constant values.

- Message Field Data Types, page 184
- Message Header Data Format, page 185
- NAMEDVARIABLE Data Format, page 185
- NAMEDARRAY Data Format, page 186
- Floating FieldDataID Values, page 187
- DeviceType Values, page 192
- LocalConnectionState (LCS) Values, page 193
- Call EventCause (CEC) Values, page 193
- SystemEventID Values, page 195
- ConnectionDeviceType Values, page 196
- Audio Codec Type Values, page 196
- CallType Values, page 197
- Control Failure (CF) Values, page 198
- Unified CCX Error Code Values, page 204
- Special Values, page 208
- Unified CCX Status Values, page 208
- Disposition Values, page 209
- Error (E) Status Codes, page 209
- Reason Codes for Agent State Change, page 211

Message Field Data Types

Table 184: Data Type Definitions , on page 184 lists the definitions of all the types of data that can appear in a Unified CCX CTI message.

Table 184: Data Type Definitions

Data type	Meaning	Byte size
CHAR	Signed integer, -128 to 127	1
UCHAR	Unsigned integer, 0 to 255	1
SHORT	Signed integer, -32,768 to 32,767	2
USHORT	Unsigned integer, 0 to 65,535	2
INT	Signed Integer, -2,147,483,648 to 2,147,483,647	4
UINT	Unsigned Integer, 0 to 4,294,967,295	4
BOOL	Boolean (False = 0, True = 1)	2
STRING[n]	Encoded using UTF-8 and can be a multi-byte or single byte (ASCII) string of length n. **Note** In a UTF string, each character may take up more than one byte. (used only in the floating part of a message)	n
UNSPEC[n]	Unspecified data occupying n consecutive bytes (used only in the floating part of a message).	n
TIME	A date/time, expressed as the number of seconds since midnight January 1, 1970 Coordinated Universal Time (UTC).	4
MHDR	Message header (see Table 185: Message Header (MHDR) Format , on page 185).	8
NAMEDVAR	A named call context variable (see NAMEDVARIABLE Data Format, on page 185).	3 … 251
NAMEDARRAY	A named call context array element (see NAMEDARRAY Data Format, on page 186).	4 … 252

Message Header Data Format

The Unified CCX CTI message header is in what is called the Message HeaDeR (MHDR) data format. This is a common format used for message headers that precede all messages exchanged between a client and a server. The following table defines the MHDR format.

Table 185: Message Header (MHDR) Format

Fixed part Field Name	Value	Data Type	Byte Size
BodyLength	The length of the message in bytes, excluding the size of the message header (the first 8 bytes).	UINT	4
MessageTypeID	This identifies the type of message and has a unique numeric value used to determine the format of the remainder of the message. Table 10: Message Types Listed by Message Class and by Message TypeName, on page 58 defines the all the messages in the message set with a unique MessageTypeID number that identifies each message. To enter a message type into a message header, enter the MessageTypeID number specified for that message in Table 10: Message Types Listed by Message Class and by Message TypeName, on page 58.	UINT	4

NAMEDVARIABLE Data Format

NAMEDVARIABLE data and NamedArray data are specially formatted floating data fields. There can be an arbitrary number of NamedVariable and NamedArray fields in a message, subject to a combined total limit of 2000 bytes.

The NAMEDVARIABLE data type is a call context variable that has been defined in the Unified CCX Script Editor. This variable length data type can appear in the floating part of a message and has the format defined in the following table.

Table 186: Named Call-Context Variable (NAMEDVAR) Format

Subfield	Value	Data Type	Maximum size
FieldDataID	NAMED_VARIABLE (= 82). This is the numeric floating field ID that indicates the following data is a named call-context variable.	UCHAR	1

Subfield	Value	Data Type	Maximum size
FieldLength	The total length of the VariableName and VariableValue fields, including the null-termination bytes. The value of this field may range from 3 to 251.	UCHAR	1
VariableName	The null-terminated defined name of the variable.	STRING	33
VariableValue	The null-terminated value of the variable.	STRING	211

For information on call context variables, see Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts and Volume 2, Editor Step Reference at http://www.cisco.com/en/US/products/sw/custcosw/ps1846/tsd_products_support_series_home.html

NAMEDARRAY Data Format

There may be an arbitrary number of NamedVariable and NamedArray fields in a message, subject to a combined total limit of 2000 bytes.

The NAMEDARRAY data type is a call context variable that has been defined in the Unified CCX Script Editor. This variable length data type may appear in the floating part of a message and has the format defined in the following table.

Table 187: Named Call Context Array Variable (NAMEDARRAY) Format

Subfield	Value	Data Type	Maximum size
FieldDataID	NAMED_ARRAY (= 83). The floating field tag that indicates that the following data is a named call context array variable.	UCHAR	1
FieldLength	The total length of the VariableIndex, VariableName and VariableValue fields, including the null-termination bytes. The value of this field may range from 4 to 252.	UCHAR	1
VariableIndex	The index of the array variable.	UCHAR	1
VariableName	The null-terminated defined name of the array variable.	STRING	33
VariableValue	The null-terminated value of the array variable.	STRING	211

For information on call context variables, see Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts and Volume 2, Editor Step Reference at http://www.cisco.com/en/US/products/sw/custcosw/ps1846/tsd_products_support_series_home.html

Floating FieldDataID Values

The following table lists all the FieldDataID numeric values that identify data fields in the floating part of a Unified CCX CTI message.

Table 188: Floating FieldDataID Values Organized by the FieldDataID Name

ID value	Floating FieldDataID name
1	ClientID
2	ClientPassword
3	ClientSignature
4	AgentExtension
5	reserved
6	AgentInstrument
7	Text
8	ANI
9	UserToUserInfo
10	DNIS
11	DialedNumber
12	CallerEnteredDigits
13-22	CallVar1 through CallVar10
23	CTIClientSignature
24	CTIClientTimeStamp
25	ConnectionDeviceID
26	AlertingDeviceID
27	CallingDeviceID
28	CalledDeviceID

ID value	Floating FieldDataID name
29	LastRedirectDeviceID
30	AnsweringDeviceID
31	HoldingDeviceID
32	RetrievingDeviceID
33	ReleasingDeviceID
34	FailingDeviceID
35	PrimaryDeviceID
36	SecondaryDeviceID
37	ControllerDeviceID
38	AddedPartyDeviceID
39	ConnectedPartyCallID
40	ConnectedPartyDeviceType
41	ConnectedPartyDeviceID
42	TransferringDeviceID
43	TransferredDeviceID
44	DivertingDeviceID
45	QueueDeviceID
46	CallWrapupData
47	NewConnectionDeviceID
48	reserved
49	AgentPassword
50	ActiveConnectionDeviceID
51	reserved
52	OtherConnectionDeviceID

ID value	Floating FieldDataID name
53	HeldConnectionDeviceID
54-55	reserved
56	CallConnectionCallID
57	CallConnectionDeviceType
58	CallConnectionDeviceID
59	CallDeviceType
60	CallDeviceID
61	CallDeviceConnectionState
62	CSQID
63-64	reserved
65	CSQState
66	reserved
67	DTMFString
68-77	reserved
78	AccountCode
79-80	reserved
81	ClientAddress
82	NamedVariable
83	NamedArray
84	reserved
85	SupervisorInstrument
86	reserved
87	AgentFlags
88-89	reserved

ID value	Floating FieldDataID name
90	AgentConnectionDeviceID
91	SupervisorConnectionDeviceID
92-94	reserved
95	CustomerPhoneNumber
96	CustomerAccountNumber
97-108	reserved
109-122	reserved
123	NextAgentState
124-125	reserved
126	SendingPort
127-128	reserved
129	MaxQueued
130-131	reserved
132	PreviousApplicationID
133	ApplicationName or CSQName
134	Description
135-136	reserved
137	FirstName
138	LastName
139-149	reserved
150	Duration
151-172	reserved
173	Extension
174-176	reserved

ID value	Floating FieldDataID name
177	ApplicationConfigKey
178	CSQConfigKey
179	AgentConfigKey
180	DeviceConfigKey
181-182	reserved
183	RecordType
184	CSQID, ApplicationID, or DeviceField0
185	AutoWork
186-188	reserved
189	AgentType
190	LoginID
191	NumCSQs
192	reserved
193	DeviceField1
194	AgentID
195	DeviceType
196-200	reserved
201	reserved
202	SecondaryConnectionCallID
203	reserved
204	TeamName
205	MemberType
206	EventDeviceID
207	reserved

ID value	Floating FieldDataID name
208	FltPeripheralID
209-216	reserved
224	MultilineAgentControl
228	NumPeripherals

DeviceType Values

Table 189: DeviceType Values , on page 192 lists the DeviceTypes that can be used in a Unified CCX CTI message with their descriptions and code numeric values. Message types that contain field names ending with ...ConnectionDeviceType or ...DeviceType use the DeviceType values listed in Table 189: DeviceType Values , on page 192.

Table 189: DeviceType Values

DeviceType	Description	Value
DEVID_NONE	No device ID is provided.	0xffff
DEVID_DEVICE_IDENTIFIER	The provided device ID identifies an IP phone (extension).	0
reserved		70
reserved		71
reserved		72
DEVID_CTI_PORT	The provided device ID identifies a CTI PORT	73
DEVID_ROUTE_POINT	The provided device ID identifies a ROUTE POINT	74
reserved		75
DEVID_AGENT_DEVICE	The provided device ID is the ID of an AGENT Device (phone)	76
DEVID_QUEUE	The provided device ID is the ID of a QUEUE	77
DEVID_NON_ACD_DEVICE_IDENTIFIER	The provided device ID is the ID belonging to Agent's non-ACD extension.	78

LocalConnectionState (LCS) Values

The LocalConnectionState is the local end state of a connection.

Table 190: LocalConnectionState Values , on page 193 lists the local connection states that can be included in a Unified CCX CTI message with their descriptions and numeric code values.

Table 190: LocalConnectionState Values

LocalConnectionState	Description	Value
LCS_NONE	Not applicable	0xffff
LCS_NULL	No relationship between call and device.	0
LCS_INITIATE	Device requesting service (“dialing”).	1
LCS_ALERTING	Device is alerting (“ringing”).	2
LCS_CONNECT	Device is actively participating in the call.	3
LCS_HOLD	Device is inactively participating in the call.	4
LCS_QUEUED	Device is stalled attempting to connect to a call, or a call is stalled attempting to connect to a device.	5
LCS_FAIL	A device-to-call or call-to-device connection attempt has been aborted.	6

Call EventCause (CEC) Values

Table 191: EventCause Values , on page 193 lists the causes of events that can be included in Unified CCX CTI messages with their numeric code values. The EventCause data field is described in CallType Fields, on page 43. An example message type containing this field is the CALL_CLEARED_EVENT message.

Table 191: EventCause Values

EventCause	Value
CEC_NONE	0xffff
reserved	1
CEC_ALTERNATE	2
CEC_BUSY	3
reserved	4

EventCause	Value
CEC_CALL_CANCELLED	5
reserved	6-8
CEC_CALL_FORWARD	9
CEC_CALL_NOT_ANSWERED	10
reserved	11-12
CEC_DEST_NOT_OBTAINABLE	13
reserved	14
CEC_INCOMPATIBLE_DESTINATION	15
reserved	16
CEC_KEY_CONFERENCE	17
reserved	18-21
CEC_NEW_CALL	22
CEC_NO_AVAILABLE_AGENTS	23
reserved	24-27
CEC_REDIRECTED	28
reserved	29-31
CEC_TRANSFER	32
reserved	33-34
CEC_TIME_OUT	35
reserved	36-42
CEC_SUPERVISOR_ASSIST	43
CEC_EMERGENCY_CALL	44
CEC_SUPERVISOR_CLEAR	45
CEC_SUPERVISOR_MONITOR	46

EventCause	Value
CEC_SUPERVISOR_WHISPER	47
CEC_SUPERVISOR_BARGE_IN	48
CEC_SUPERVISOR_INTERCEPT	49
CEC_CALL_PARTY_UPDATE_IND	50
Extended Call Cleared Event Causes	
reserved	1001-1013
CECX_DROP_HANDLED_OTHER	1014
reserved	1015-1056

SystemEventID Values

Table 192: System Event ID Values , on page 195 lists the system event IDs that can be included in the Unified CCX CTI SYSTEM_EVENT message, with their descriptions and numeric code values.

Table 192: System Event ID Values

SystemEventID	Description	Value
reserved		1
reserved		2
SYS_CCX_ONLINE	Unified CCX has gone online.	3
SYS_CCX_OFFLINE	Unified CCX has gone offline.	4
reserved		5
reserved		6
reserved		7
reserved		8
reserved		9
SYS_INSTRUMENT_OUT_OF_SERVICE	An Agent or Unified CCX device target has been removed from service.	10

SystemEventID	Description	Value
SYS_INSTRUMENT_BACK_IN_SERVICE	An EnterpriseAgent or IPCC device target has been returned to service.	11

ConnectionDeviceType Values

Table 193: ConnectionDevice Type Values, on page 196 lists the types of connecting devices that can be included in a Unified CCX CTI message with their descriptions and numeric code values.

Table 193: ConnectionDevice Type Values

ConnectionDeviceType	Description	Value
CONNECTION_ID_NONE	No ConnectionDevice type is provided.	0xffff
CONNECTION_ID_STATIC	The ConnectionDevice type value is stable over time between calls.	0
CONNECTION_ID_DYNAMIC	The ConnectionDevice type value is dynamic and may change between calls.	1

Audio Codec Type Values

Table 194: Code Values for Audio Codec Types

Audio Codec Type	Value
NONSTANDARD	0
G711ALAW64K	1
G711ALAW56K	2
G711ULAW64K	3
G711ULAW56K	4
G722_64K	5
G722_56K	6
G722_48K	7
G7231	8

Audio Codec Type	Value
G728	9
G729	10
G729ANNEXA	11
IS11172AUDIOCAP	12
IS13818AUDIOCAP	13
ACY_G729AASSN	14
DATA64	15
DATA56	16
GSM	17
ACTIVEVOICE	18

CallType Values

Table 195: CallType Values , on page 197 lists the call types that can be included within a message with their descriptions and values.

For further information on CallType fields, see CallType Fields, on page 43.

Table 195: CallType Values

CallType	Description	Value
CALLTYPE_INVALID	The call type is not valid.	0
CALLTYPE_CCX_IN	Inbound Unified CCX call.	1
reserved		2-3
CALLTYPE _TRANSFER_IN	Transferred inbound call.	4
reserved		5
CALLTYPE _OTHER_IN	Inbound call.	6
reserved		7-8
CALLTYPE _OUT	Outbound call.	9

CallType	Description	Value
CALLTYPE _AGENT_INSIDE	Agent inside call.	10
reserved		11
CALLTYPE _CONSULT	Consult call.	12
reserved		13-14
CALLTYPE _CONFERENCE	Conference call.	15
reserved		16
CALLTYPE_PREVIEW	Call is an outbound preview call.	17
CALLTYPE_RESERVATION	Call is an outbound reservation call.	18
CALLTYPE_ASSIST	Call to supervisor for assistance.	19
CALLTYPE_EMERGENCY	Emergency call.	20
CALLTYPE_SUPERVISOR_ MONITOR	Supervisor silently monitoring call.	21
reserved		22
CALLTYPE_ SUPERVISOR_BARGE_IN	Supervisor conferenced into call.	23
CALLTYPE_ SUPERVISOR_INTERCEPT	Supervisor replaces agent on call.	24
reserved		25-26

Control Failure (CF) Values

Table 196: Control Failure Code Values , on page 198 lists the Control Failure code values that can appear in the CONTROL_FAILURE_CONF message.

Table 196: Control Failure Code Values

FailureCode	Description	Value
CF_GENERIC_UNSPECIFIED	An error has occurred that is not one of the following error types.	0
CF_GENERIC_OPERATION	An operation error occurred (no specific details available).	1

FailureCode	Description	Value
CF_REQUEST_INCOMPATIBLE_WITH_OBJECT	The request is not compatible with the object.	2
CF_VALUE_OUT_OF_RANGE	The parameter has a value that is not in the range defined for the server.	3
CF_OBJECT_NOT_KNOWN	The parameter has a value that is not known to the server.	4
CF_INVALID_CALLING_DEVICE	The calling device is not valid.	5
CF_INVALID_CALLED_DEVICE	The called device is not valid.	6
CF_INVALID_FORWARDING_DESTINATION	The forwarding destination device is not valid.	7
CF_PRIVILEGE_VIOLATION_ON_SPECIFIED_DEVICE	The specified device is not authorized for the service.	8
CF_PRIVILEGE_VIOLATION_ON_CALLED_DEVICE	The called device is not authorized for the service.	9
CF_PRIVILEGE_VIOLATION_ON_CALLING_DEVICE	The calling device is not authorized for the service.	10
CF_INVALID_CSTA_CALL_ID	The call ID is not valid.	11
CF_INVALID_CSTA_DEVICE_ID	The device ID is not valid.	12
CF_INVALID_CSTA_CONNECTION_ID	The connection ID is not valid.	13
CF_INVALID_DESTINATION	The request specified a destination that is not valid.	14
CF_INVALID_FEATURE	The request specified a feature that is not valid.	15
CF_INVALID_ALLOCATION_STATE	The request specified an allocation state that is not valid.	16
CF_INVALID_CROSS_REF_ID	The request specified a cross-reference ID that is not in use at this time.	17
CF_INVALID_OBJECT_TYPE	The request specified an invalid object type.	18
CF_SECURITY_VIOLATION	Security error (no specific details available).	19

FailureCode	Description	Value
CF_GENERIC_STATE_INCOMPATIBILITY	The request is not compatible with the condition of a related device.	21
CF_INVALID_OBJECT_STATE	The object is in the incorrect state for the request.	22
CF_INVALID_CONNECTION_ID_FOR_ACTIVE_CALL	The active connection ID in the request is invalid.	23
CF_NO_ACTIVE_CALL	There is no active call for the request.	24
CF_NO_HELD_CALL	There is no held call for the request.	25
CF_NO_CALL_TO_CLEAR	There is no call associated with the given connection ID.	26
CF_NO_CONNECTION_TO_CLEAR	There is no call connection for the given connection ID.	27
CF_NO_CALL_TO_ANSWER	There is no alerting call to be answered.	28
CF_NO_CALL_TO_COMPLETE	There is no active call to be completed.	29
CF_GENERIC_SYSTEM_RESOURCE_AVAILABILITY	The request failed due to lack of system resources (no specific details available).	31
CF_SERVICE_BUSY	The service is temporarily unavailable.	32
CF_RESOURCE_BUSY	An internal resource is busy.	33
CF_RESOURCE_OUT_OF_SERVICE	The service requires a resource that is out of service.	34
CF_NETWORK_BUSY	The server sub-domain is busy.	35
CF_NETWORK_OUT_OF_SERVICE	The server sub-domain is out of service.	36
CF_ OVERALL_MONITOR_LIMIT_EXCEEDED	The request would exceed the server's overall resource limits.	37
CF_ CONFERENCE_MEMBER_LIMIT_EXCEEDED	The request would exceed the server's limit on the number of conference members.	38
CF_ GENERIC_SUBSCRIBED_RESOURCE_AVAILABILITY	The request failed due to lack of purchased or contracted resources (no specific details available).	41
CF_ OBJECT_MONITOR_LIMIT_EXCEEDED	The request would exceed the server's specific resource limits.	42

FailureCode	Description	Value
CF_ EXTERNAL_TRUNK_LIMIT_EXCEEDED	The request would exceed the limit of external trunks.	43
CF_ OUTSTANDING_REQUEST_LIMIT_EXCEEDED	The request would exceed the limit of outstanding requests.	44
CF_ GENERIC_PERFORMANCE_MANAGEMENT	The request failed as a performance management mechanism (no specific details available).	51
CF_PERFORMANCE_LIMIT_EXCEEDED	The request failed because a performance management limit was exceeded.	52
CF_ SEQUENCE_NUMBER_VIOLATED	The server has detected an error in the sequence number of the operation.	61
CF_ TIME_STAMP_VIOLATED	The server has detected an error in the time stamp of the operation.	62
CF_ PAC_VIOLATED	The server has detected an error in the PAC of the operation.	63
CF_ SEAL_VIOLATED	The server has detected an error in the Seal of the operation.	64
CF_ GENERIC_UNSPECIFIED_REJECTION	The request has been rejected (no specific details available).	70
CF_ GENERIC_OPERATION_REJECTION	The requested operation has been rejected (no specific details available).	71
CF_ DUPLICATE_INVOCATION_REJECTION	The request duplicated another request for the same service.	72
CF_ UNRECOGNIZED_OPERATION_REJECTION	The request specified an unrecognized operation.	73
CF_ MISTYPED_ARGUMENT_REJECTION	The request contained a parameter of the wrong type for the requested operation.	74
CF_ RESOURCE_LIMITATION_REJECTION	The request would have exceeded a resource limitation.	75
CF_ ACS_HANDLE_TERMINATION_REJECTION	The request specified an ACS handle that is no longer in use.	76
CF_ SERVICE_TERMINATION_REJECTION	The request failed because the required service has been terminated.	77

FailureCode	Description	Value
CF_ REQUEST_TIMEOUT_REJECTION	The request failed because a timeout limit was exceeded.	78
CF_ REQUESTS_ON_DEVICE_EXCEEDED_REJECTION	The request would have exceeded the limits of the device.	79
Extended Control failure codes		
CF_INVALID_AGENT_ID_SPECIFIED	The request specified an invalid AgentID.	256
CF_INVALID_PASSWORD_SPECIFIED	The request specified an invalid agent password.	257
CF_INVALID_AGENT_ID_OR_PASSWORD_SPECIFIED	The request specified an invalid AgentID and/or invalid agent password.	258
CF_SPECIFIED_AGENT_ALREADY_SIGNED_ON	The request failed because the specified agent is already logged in.	259
CF_INVALID_LOGON_DEVICE_SPECIFIED	The request specified an invalid logon device.	260
CF_INVALID_ANSWERING_DEVICE_SPECIFIED	The request specified an invalid answering device.	261
CF_INVALID_SKILL_GROUP_SPECIFIED	The request specified an invalid agent skill group.	262
CF_INVALID_CLASS_OF_SERVICE_SPECIFIED	The request specified an invalid class of service.	263
CF_INVALID_TEAM_SPECIFIED	The request specified an invalid team	264
CF_INVALID_AGENT_WORKMODE	The request specified an invalid agent work mode.	265
CF_INVALID_AGENT_REASON_CODE	The request specified an invalid agent reason code.	266
CF_ADJUNCT_SWITCH_COMM_ERROR	A communication error occurred on the datalink between the Unified ICME system and the Unified CCX system.	267
CF_AGENT_NOT_PARTY_ON_CALL	The specified agent is not a party on the indicated call.	268
CF_INTERNAL_PROCESSING_ERROR	An internal error occurred in the Unified CCX system while processing the request.	269

FailureCode	Description	Value
CF_TAKE_CALL_CONTROL_REJECTION	The Unified CCX system refused a Unified ICME request to take control of a call.	270
CF_TAKE_DOMAIN_CONTROL_REJECTION	The Unified CCX system refused an Unified ICME request to take control of a domain.	271
CF_REQUESTED_SERVICE_NOT_REGISTERED	The Unified ICME system is not registered on the Unified CCX system for the requested service.	272
reserved		273-282
CF_SPECIFIED_EXTENSION_ALREADY_IN_USE	Specified extension is already in use. Use a different extension.	283
CF_ARBITRARY_CONF_OR_XFER_NOT_SUPPORTED	Arbitrary conference or transfer is currently not supported.	284
CF_NETWORK_TRANSFER_OR_CONSULT		285
CF_NETWORK_TRANSFER_OR_CONSULT_FAILED		286
CF_DEVICE_RESTRICTED	Device is restricted, not under CTI control.	287
CF_LINE_RESTRICTED	Line is restricted, not under CTI control.	288
CF_MAXIMUM_LINE_EXCEEDED	The number of configured extensions for the agent device exceeds the maximum allowed.	291
CF_SHARED_LINES_NOT_SUPPORTED	An extension on the agent device is shared with one or more other devices.	292
CF_EXTENSION_NOT_UNIQUE	An extension on the agent device is not unique and is configured on multiple partitions.	293
CF_UNKNOWN_INTERFACE_CTRLR_ID	The Interface Controller ID is unknown.	1001
CF_INVALID_INTERFACE_CTRLR_TYPE	The Interface Controller type is not valid.	1002
CF_SOFTWARE_REV_NO_SUPPORTED	The current software revision is not supported.	1003
CF_UNKNOWN_PID	One of the reserved fields in the fixed part of the message is incorrect.	1004

FailureCode	Description	Value
CF_INVALID_TABLE_SPECIFIED	An invalid table was specified.	1005
reserved		1006
CF_UNKNOWN_ROUTING_CLIENT_ID	The RoutingClientID is unknown.	1007
CF_RC_SERVICE_INACTIVATE	The routing client service is not active.	1008
CF_INVALID_DIALED_NUMBER	The dialed number is invalid.	1009
CF_INVALID_PARAMETER	A parameter supplied in the request is invalid.	1010
CF_UNKNOWN_ROUTING_PROBLEM	An unspecified error occurred during routing.	1011
reserved		1012
CF_UNSUPPORTED_RC_MESSAGE_REVISION	The requested routing client service protocol version is not supported.	1013
CF_UNSUPPORTED_IC_MESSAGE_REVISION	The requested interface controller service protocol version is not supported.	1014
reserved		1015

Unified CCX Error Code Values

Table 197: Unified CCX Error Code Values

Unified CCX Error Code	Description	Value
CCX_DEFAULT_ERROR_CODE	Unknown problem.	0
CCX_JTAPI_CCM_PROBLEM	JTAPI Cisco Unified Communications Manager problem.	88001
CCX_LOOKUP_FAILURE	Unified CCX lookup failure.	88002
CCX_AGENT_DN_NOT_ASSIGNED	The Unified CCX agent dialed number has not been assigned.	88003
CCX_AGENT_DEVICE_OFF	The Unified CCX agent device is off.	88004
CCX_AGENT_DEVICE_BUSY	The Unified CCX agent device is busy.	88005

Unified CCX Error Code	Description	Value
CCX_RMCM_CONTACT_NULL	There is no Unified CCX RmCm contact.	88006
CCX_RMCM_CONTACT_INVALID	The Unified CCX RmCm contact is invalid.	88007
CCX_AGENT_NULL	There is no Unified CCX agent.	88008
CCX_AGENT_INVALID	The Unified CCX agent is invalid.	88009
CCX_INVALID_ARGUMENT	There is an invalid Unified CCX command argument.	88010
CCX_NO_CALLBACK_FROM_AGENT	There is no callback from a Unified CCX agent	88011
CCX_CALL_ID_NULL	There is no Unified CCX call ID.	88012
CCX_CALL_OBJECT_NULL	There is no Unified CCX call object.	88013
CCX_JTAPI_PROVIDER_NULL	There is no Unified CCX JTAPI provider.	88014
CCX_DIALED_NUMBER_NULL	There is no Unified CCX dialed number.	88015
CCX_DEVICE_ID_NULL	There is no Unified CCX device ID.	88016
CCX_TERMINAL_CONNECTION_NULL	There is no Unified CCX terminal connection.	88017
CCX_CALL_CONTROL_MASK_NOT_ON	The Unified CCX call control mask is not on.	88018
CCX_AGENT_QUEUE_STATUS	The status of the Unified CCX agent queue.	88019
CCX_ICD_RTDM_NULL	The Unified CCX Real Time Data Manager does not exist.	88020
CCX_CSQ_ICD_STATUS	The status of the Unified CCX CSQ.	88021
CCX_OVERALL_ICD_STATUS	The overall Unified CCX system status.	88022
CCX_INVALID_SUPERVISORY_ACTION	The Unified CCX supervisory action is invalid.	88023
CCX_CALL_DATA_UPDATE_MASK_NOT_ON	The Unified CCX call data update mask is not on.	88024
CCX_OVERALL_LIMIT_EXCEEDED	The Unified CCX overall limit is exceeded.	88025
CCX_CALL_ID_INVALID	The Unified CCX call ID is invalid.	88026

Unified CCX Error Code	Description	Value
CCX_SUPERVISOR_UNAVAILABLE	The Unified CCX supervisor is unavailable.	88027
CCX_UNSUPPORTED_OPERATION	The Unified CCX operation is unsupported.	88028
CCX_CALL_ID_NOT_IN_PROVIDER	The specified Unified CCX call ID is currently not in the RmCm JTAPI provider.	88029
CCX_DEVICE_RESTRICTED	The Unified CCX device is restricted on the Cisco Unified Communications Manager.	88030
CCX_LINE_RESTRICTED	The Unified CCX line is restricted on the Cisco Unified Communications Manager.	88031
CCX_AGENT_LINE_ON_MULTIPLE_DEVICES	The Unified CCX agent line is on multiple devices.	88032
CCX_USER_OPERATION_FAILED	The Unified CCX user operation failed.	88033
CCX_PROVIDER_OPERATION_FAILED	The Unified CCX provider operation failed.	88034
CCX_PASSWORD_LOCKED	The Unified CCX password is locked.	88035
CCX_PASSWORD_EXPIRED	The Unified CCX password has expired.	88036
CCX_PASSWORD_EXPIRED_SUCCESS	The Unified CCX password has successfully expired.	88037
CCX_PASSWORD_HACKED_LOCKED	The Unified CCX password has been hacked and is locked.	88038
CCX_PASSWORD_INACTIVE_LOCKED	The Unified CCX password is inactive and is locked.	88039
CCX_PASSWORD_MUST_CHANGE	The Unified CCX password must be changed.	88040
CCX_AGENT_DEVICE_IPv6	Agent login is not supported for IPv6-enabled devices.	88041
CCX_TRANSFER_FAILED_DUE_TO_CCM_PROBLEM	External transfer is restricted due to logical partitioning policy.	88042
CCX_CONFERENCE_FAILED_DUE_TO_CCM_PROBLEM	Conference is unavailable due to logical partitioning policy.	88043

Unified CCX Error Code	Description	Value
CCX_AGENT_LOGIN_NOTSUPPORTED_JALENABLEDPHONE	Agent login is not supported for JAL-enabled phones. This field is not applicable from Unified CCX 8.0(1) release onward.	88044
CCX_AGENT_LOGIN_NOTSUPPORTED_DTALENABLEDPHONE	Agent login is not supported for DTAL-enabled phones. This field is not applicable from Unified CCX 8.0(1) release onward.	88045
CCX_BIB_NOT_CONFIGURED	Built-In Bridge is not turned on in the agent phone.	88046
CCX_AGENT_DEVICE_NOT_SUPPORTED	Agent phone is earlier than Cisco IP Phone 7960 series, hence it does not support Silent Monitoring call.	88047
CCX_SILENT_MONITOR_OR_RECORDING_SETTINGS_NOT_CONFIGURED	Silent Monitoring or Recording capability of the Unified CCX RmCm Application User is removed in Cisco Unified Communications Manager.	88048
CCX_SILENT_MONITOR_OR_RECORDING_AGENT_CALL_STATE_NOT_TALKING	Agent phone ICD line to be monitored or recorded is not in the Talking state (on hold, ringing, and other states).	88049
CCX_AGENT_NOT_ASSOCIATED	Agent is not associated with the supervisor.	88050
CCX_RECORDING_ALREADY_INPROGRESS	Agent phone ICD line to be recorded is already participating in another recording session.	88051
CCX_RECORDING_CONFIG_NOT_MATCHING	Agent phone ICD line to be recorded does not have correct recording configuration in Cisco Unified Communications Manager.	88052
CCX_BIB_RESOURCE_NOT_AVAILABLE	Agent phone Built-In Bridge resource is not available.	88053
CCX_ILLEGAL_CALL_STATE	Call at the agent phone ICD line cannot perform the operation in its current state.	88054
CCX_CALLMANAGER_OPERATION_TIMEOUT	Cisco Unified Communications Manager operation timed out (failed).	88055
CCX_MONITORING_ALREADY_INPROGRESS	Agent phone line to be monitored is already participating in another monitoring session.	88056

Unified CCX Error Code	Description	Value
CCX_RECORDING_LICENSE_LIMIT_EXCEEDED	Number of concurrent recordings exceeds the value of recording count available in the Recording License.	88057

Special Values

Table 198: Special Values , on page 208 shows the numeric code values used to define sizes and limits, indicate special IDs, and unspecified data elements in Unified CCX CTI messages.

Table 198: Special Values

Constant	Description	Value
MAX_NUM_DEVICES	The maximum number of call devices that can be in a message list.	16
MAX_CSQ_COUNT	In AGENT_STATE_EVENT message	99
MAX_AGENTS	In AGENT_TEAM_CONFIG message	64
NULL_CALL_ID	Indicates that no call ID is supplied.	0xFFFFFFFF
NULL_APPLICATION	Indicates that no Service is supplied.	0xFFFFFFFF
NULL_CSQ	Indicates that no CSQ is supplied.	0xFFFFFFFF
NULL_LINEHANDLE	In message constants	0xFFFF

Unified CCX Status Values

Table 199: Unified CCX Status Values , on page 208 lists the numeric codes with their descriptions for the status of the Unified CCX. The SYSTEM_EVENT message contains the Unified CCX Status field.

Table 199: Unified CCX Status Values

Unified CCX status	Description	Mask value
CCX_INITIALIZING	The server is initializing	0x00000100
CCX_SHUTTING_DOWN	The server is shutting down	0x00000200
CCX_SHUTDOWN	The server is shut down	0x00000300
CCX_IN_SERVICE	The server is in service	0x00000400

Unified CCX status	Description	Mask value
CCX_PARTIAL_SERVICE	The server is in Partial Service	0x00000500
CCX_OUT_OF_SERVICE	The server failed to initialize	0x00000600

Disposition Values

Table 200: Call-Party Disposition Values , on page 209 lists all the numeric call-party disposition numeric code values with their meanings that can occur in Unified CCX CTI messages.

Table 200: Call-Party Disposition Values

Disposition Code	Meaning
1-14	reserved
15	Redirected
16-27	reserved
28	reserved
29	Announced Transfer
30	Conferenced
31-52	reserved

Error (E) Status Codes

Table 201: Error Status Codes , on page 209 lists the status codes with their numeric code values that may be included in the FAILURE_CONF and FAILURE_EVENT Unified CCX CTI messages.

Table 201: Error Status Codes

Status code	Description	Value
E_CTI_NO_ERROR	No error occurred.	0
E_CTI_INVALID_VERSION	The server does not support the protocol version number requested by the client.	1
E_CTI_INVALID_MESSAGE_TYPE	A message with an invalid message type field was received.	2

Status code	Description	Value
E_CTI_INVALID_FIELD	A message with an invalid floating field tag was received.	3
E_CTI_SESSION_NOT_OPEN	No session is currently open on the connection.	4
E_CTI_SESSION_ALREADY_OPEN	A session is already open on the connection.	5
E_CTI_REQUIRED_DATA_MISSING	The request did not include one or more floating items that are required.	6
E_CTI_INVALID_PERIPHERAL_ID	One of the reserved fields in the fixed part of the message is incorrect.	7
E_CTI_INVALID_AGENT_DATA	The provided agent data item(s) are not valid.	8
E_CTI_AGENT_NOT_LOGGED_ON	The indicated agent is not currently logged on.	9
E_CTI_DEVICE_IN_USE	The indicated agent IP phone is already associated with a different client.	10
E_CTI_NEW_SESSION_OPENED	This session is being terminated due to a new session open request from the client.	11
reserved		12
E_CTI_INVALID_CALLID	A request message was received with an invalid CallID value.	13
reserved		14-16
E_CTI_UNSPECIFIED_FAILURE	An unspecified error occurred.	17
E_CTI_INVALID_TIMEOUT	The IdleTimeout field contains a value that less than 20 seconds (4 times the minimum heartbeat interval of 5 seconds).	18
reserved		19-22
E_CTI_INVALID_FIELD_LENGTH	A floating field exceeds the allowable length for that field type.	23
reserved		24-91
E_CTI_SERVER_NOT_MASTER	The server is a standby server.	92
E_CTI_INVALID_CSQ	The specified CSQ is invalid.	93
E_CTI_JTAPI_CCM_PROBLEM	Indicates a JTAPI or Cisco Unified Communications Manager problem.	94

Status code	Description	Value
reserved		95
E_CTI_AUTO_CONFIG_RESET	The client needs to resend the CONFIG_REQUEST_KEY_EVENT and the CONFIG_REQUEST_EVENT message.	96

Reason Codes for Agent State Change

Table 1 lists the reason codes for agent state change, description, and their numeric code values for change in agent's state that may be included in the AGENT_STATE_EVENT Unified CCX CTI messages.

Table 202: Reason Codes for Agent State Change

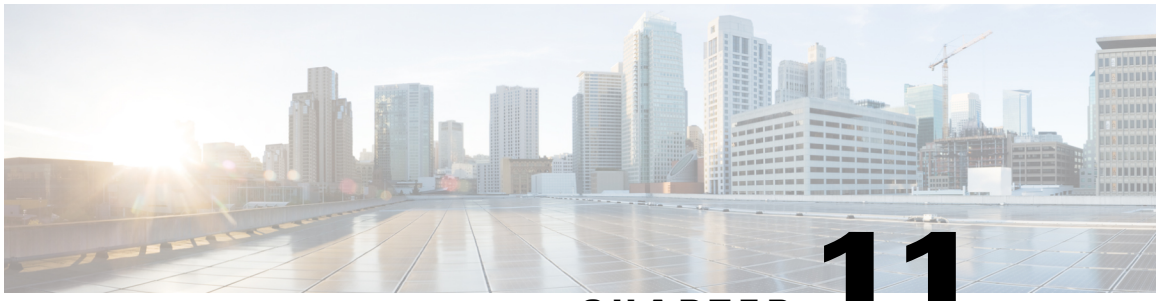
Reason codes for agent state change	Description	Value
AGT_RELOGON	The Agent logs in again.	32767
AGT_CLOSE_CAD	The Agent closes the CAD	32766
AGT_CONN_DOWN	Connection between the Unified CCX and CAD is down.	32765
CCX_FAILURE	Unified CCX failure.	32764
AGT_RNA	The Agent did not answer the call.	32763
AGT_OFFHOOK	The Agent goes off-hook.	32762
AGT_NON_ICD	The Agent receives a direct non-business call.	32761
AGT_LOGON	The Agent logs in.	32760
AGT_PHONE_DOWN	The Agent's phone is down.	32759
AGT_WORK_TIMER_EXP	The Agent's work timer expired.	32758
CM_FAILOVER	Call Manager failover.	32757
AGT_PHONE_UP	The Agent's phone is up.	32756
AGT_CALL_ENDED	The Agent's Call ended.	32755
AGT_DEVICE_RESTRICTED	The Agent's device is restricted.	32754
AGT_LINE_RESTRICTED	The Agent's line is restricted.	32753

Reason codes for agent state change	Description	Value
AGT_CANCEL_RESERVATION	The Agent cancels reservation for preview outbound call.	32752
AGT_SKIPS	The Agent skips the preview outbound call.	32751
AGT_ICD_EXTENSION_CHANGED	The Agent's ICD extension has changed.	32750
AGT_RESERVED_OUTBOUND_DIRECT PREVIEW	The Agent is reserved for a preview outbound call.	32746
AGT_RESERVED_OUTBOUND	The Agent is reserved for a progressive or predictive outbound call.	32747
EMAIL_CONTACT_NOT_ACCEPTED	Auto accept of email failed.	32743
ACCX_MASTERSHIP_CHANGE		32744

PART III

System Level Information

- Client Application Development Guidelines, page 215
- Document History, page 245



CHAPTER 11

Client Application Development Guidelines

This chapter describes the following topics.

- Configuring a Client Program on Unified CCX, page 215
- Debugging a Unified CCX Client Program, page 215
- Call-Event Message Flows as Seen in the Log Files, page 217

Configuring a Client Program on Unified CCX

The Unified CCX CTI server is part of the RmCm subsystem of the Unified CCX system. Therefore, before clients can connect to a Unified CCX CTI server, the RMCM subsystem must be in service.

Unified CCX listens on port 42027 for incoming client connections. You can set this port number in the System Parameters window of the Unified CCX Administration web page. The parameter name for the port number is RmCm TCP Port.

Debugging a Unified CCX Client Program

Unified CCX provides detailed Unified CCX CTI message-level traces in the MIVR log files. You can use these trace messages to help find the cause of a problem. For example, if a message from a client is ill-formed, you will most likely find an exception associated with processing the message in the trace log file.

You should be aware of the following concerning message-level traces:

- To enable a message-level trace, you must turn on the trace for the ICD_CTI subfacility from the Unified CCX Administration Trace Configuration web page. It is also recommended to turn on trace for the RmCm, RM, and CM subsystems.
- To enable a detailed trace (low-level), in the Trace Configuration web page, for the Miscellaneous ICD_CTI subfacility, you must also set all trace debugging levels.
- To access a MIVR log file, select **System** > **Engine** > **Trace Files** from the Unified CCX Administration web page.

See the Unified CCX Administration online help or the Cisco Unified Contact Centre Express Administration Guide for complete details on configuring, enabling, and accessing message-level trace log files.

You can use trace messages in Unified CCX for the following purposes:

- To examine message field values:

  At a high-level trace, the message content is displayed in text format that includes field names and their values.

  Note Trace messages are primarily intended to be used by Cisco engineers. Contact the Cisco support center for assistance if you need help.

  Some field names used in a Unified CCX high-level trace log file are different from the ones used in this document. In addition, some fields listed with names in a high-level trace are listed as reserved in this document. Finally, some fields listed in a trace are used internally only and are therefore not documented in this guide. Because of these discrepancies, you should not use a high-level trace unless you are comfortable with the terminology used in the trace messages.

- To examine call events in detail:

  Unified CCX can dump the incoming and outgoing messages in binary using hexadecimal format. That is the most detailed level of message content that you can examine.

- To trace a message as it appears multiple times at its different processing stages in the trace log.

Example Trace Log-File Excerpts

The following are two example trace log-file excerpts for the same BEGIN_CALL event: one at the high-level text trace and one at the low-level binary trace.

A high-level trace lists each message that is sent with its type, length, field names, and field values. Even though some field names used in the high-level trace log file are different from those documented in this guide (as explained in the previous section), you can still can still get a lot of information from a high-level trace.

A low-level trace lists the bytes used by each message and what is stored in each byte. To understand this information, you need to count the bytes and their order reserved for each field in a particular message definition.

Example 1: High-Level (Text) Trace

```
7091923: Nov 20 10:43:59.435 EDT %MIVR-ICD_CTI-7-UNK:
CSOutboundMsgProcessor: got an event message:
{ length=-1 type=BEGIN_CALL_EVENT, monitorID: 0, peripheralID: 1,
peripheralType: 21, numCTIClients: 0, numNamedVars: 0,
numNamedArrays: 0, callType: 1, ConnectionDeviceType: 0,
connectionCallID: 16777488, calledPartyDisposition: 0,
connectionDeviceID: 9026, ani[2]: 2006, dnis: 7000, dialedNumber: null,
callerEnteredDigits: null, callVar1: null, callVar2: null,
callVar3: null, callVar4: null, callVar5: null, callVar6: null,
callVar7: null, callVar8: null, callVar9: null, callVar10: null,
wrapupData: null }
```

[2] If the ANI is empty use the CallingDeviceID instead of ANI.

Example 2: Low-Level (Binary) Trace

```
7091948: Nov 20 10:43:59.435 EDT %MIVR-ICD_CTI-7-UNK:writing...
MHDR:    0000: 00 00 00 36 00 00 00 17 ...6....
MDATA:   0000: 00 00 00 00 00 00 00 01   00 15 00 00 00 00 00 00 ....
0010: 00 01 00 00 01 00 01 10   00 00 08 05 32 30 30 36  ............2006
0020: 00 0B 05 37 30 30 30 00   19 05 39 30 32 36 00 0A ..7000...9026..
0030: 05 37 30 30 30 00                                  .7000.
```

It is not the intention of this document to give a full description of traces in the Unified CCX logs. For more detailed information on debugging and using trace files in Unified CCX, see the *Cisco Unified Contact Center Express Servicing and Troubleshooting Guide* and the *Cisco Unified Contact Center Express Administration Guide*.

The System Impact of Client Programs

Any client program connected to Unified CCX generates additional demand of resources on the Unified CCX system. Client developers must inform their customers about this impact and a potential degradation of system performance when they enable a trace.

The system resource requirement of the client program should be measured and the corresponding Unified CCX supported agent count, CSQ count, call rate, and database size should be reduced.

Handling Reserved Messages and Reserved Fields in Client Programs

A client program may receive reserved messages and messages with reserved fields. If a message is a reserved message, the client program should ignore the message by skipping body-length bytes after the message header:

- If a message contains reserved fixed fields, the client program should skip that number of bytes based on the field size.
- If a message contains a reserved floating field, the client program should skip the number of bytes identified in the FieldDataLength field plus two bytes.

Call-Event Message Flows as Seen in the Log Files

One of the main functions of Unified CCX is to process and queue calls. Although there is no guarantee of call events, understanding typical call event message flows helps a developer to write client programs.

The following call-event message flows list in a call event the messages that are sent, to whom they are sent, and what they indicate.

Note The message order can vary, depending on the situation. There is no guarantee of a particular message order, though, the message order in these examples are typical.

Since message order is not guaranteed, Unified CCX might send an agent selection message before a call queued message even though Unified CCX first queues the call and then selects the agent for the call.

This section describes the following common call-event message flows:

- Example Message Flow for a Queued and Answered Call, on page 218

- Example Message Flow for a Transferred Call, on page 225
- Example Message Flow for a Conferenced Call, on page 232
- Example Message Flows for an Outbound Option Call, on page 236

Example Message Flow for a Queued and Answered Call

In this example message flow, Unified CCX is connected to the Cisco Unified Communications Manager. Unified CCX is configured with two agents: one agent is associated with extension 2005 and the other agent is associated with extension 2006.

The agent on extension 2005 is acting as a caller. One client program is connected for the agent on extension 2005. Another client program is connected for the agent 2006.

One client program is connected as a bridge mode client. The agent with extension 2006 is associated with a CSQ. The CSQ is associated with the trigger 7000.

The call flow starts when the agent on extension 2005 makes a call to the 7000 trigger. Calls are processed using the system icd.aef script.

Note For each of the tables in this section, the value in the **Caller 2005** column is Yes if extension 2005 is a logged in agent, No if extension 2005 is not a logged in agent.

1 The caller on extension 2005 makes the call to trigger 7000.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	

`MsgType: BEGIN_CALL_EVENT`
`  NumCTIClients: 0`
`  NumNamedVars: 0`
`  NumNamedArrays: 0`
`  CallType: 9`
`  ConnectionDeviceType: 0`
`  CallID: 16777507`
`  CalledPartyDisposition: 0`
`  ConnectionDeviceID: 2005`
`  ANI`[3]`: 2005`
`...`
	Yes	No	Yes	This message indicates the beginning of the call. Notice that the CallID is 16777507.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: CALL_SERVICE_INITIATED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ApplicationID: -1 CSQID: -1 CallingDeviceType: 76 LocalConnectionState: 1 EventCause: 65535 ConnectionDeviceID: 2005 CallingDeviceID: 2005	Yes	No	Yes	This message indicates the device 2005 is off-hook. This message is generated only if device 2005 is monitored.
MsgType: CALL_ORIGINATED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ApplicationID: -1 CSQID: -1 CallingDeviceType: 76 CalledDeviceType: 74 LocalConnectionState: 1 EventCause: 65535 ConnectionDeviceID: 2005 CallingDeviceID: 2005 CalledDeviceID: 7000	Yes	No	Yes	This message indicates the call has dialed a number 7000. Notice that the CalledDeviceID is 7000. This message is only generated if device 2005 is monitored.
MsgType: CALL_DELIVERED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ApplicationID: 2 CSQID: -1 AlertingDeviceType: 73 CallingDeviceType: 76 CalledDeviceType: 73 LastRedirectedDeviceType: 65535 LocalConnectionState: 2 EventCause: 22 NumNamedVars: 0 NumNamedArrays: 0 ConnectionDeviceID: 9019 AlertingDeviceID: 9019 CallingDeviceID: 2005 CalledDeviceID: 9019 LastRedirectedDeviceID: SecondaryCallID: 0 ANI: 2005 DNIS: 7000 DialedNumber: 7000 ...	Yes	No	Yes	The Unified CCX trigger is alerted. The call from device 2005 is delivered to CTI port 9019 in a Call Control Group. If the ANI is empty use the CallingDeviceID instead of ANI.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: CALL_ESTABLISHED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ApplicationID: 2 CSQID: -1 AnsweringDeviceType: 73 CallingDeviceType: 76 CalledDeviceType: 73 LastRedirectedDeviceType: 65535 LocalConnectionState: 3 EventCause: 22 ConnectionDeviceID: 9019 AnsweringDeviceID: 9019 CallingDeviceID: 2005 CalledDeviceID: 9019 LastRedirectedDeviceID:	Yes	No	Yes	The Unified CCX trigger accepts the call. CTI port 9019 accepts the call. This is the result of the Accept step in the icd.aef script.

3

2 The call is queued.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: CALL_QUEUED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ApplicationID: 2 QueueDeviceType: 77 CallingDeviceType: 76 CalledDeviceType: 74 LastRedirectedDeviceType: 65535 NumCSQs: 1 LocalConnectionState: 5 EventCause: 65535 ConnectionDeviceID: 9019 QueueDeviceID: 1 CallingDeviceID: 2005 CalledDeviceID: 7000 LastRedirectedDeviceID: CsqID[0]: 1	Yes	No	Yes	This message indicates that the call is queued in a CSQ with the ID 1.

3 The agent becomes Ready.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: AGENT_STATE_EVENT CSQState: 3 StateDuration: 0 CSQID: -1 AgentState: 3 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2006 AgentID: agt2006 NextAgentState: 65535	Yes	No	Yes	The agent on extension 2006 changes state to Ready. Notice that the AgentState is 3.

4 Agent is reserved.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: AGENT_STATE_EVENT CSQState: 3 StateDuration: 0 CSQID: -1 AgentState: 3 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2006 AgentID: agt2006 NextAgentState: 65535	Yes	Yes	Yes	The CSQ identifies the agent on extension 2006 and reserves that agent.

5 The agent phone on extension 2006 rings.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: CALL_DELIVERED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ApplicationID: 2 CSQID: 1 AlertingDeviceType: 76 CallingDeviceType: 76 CalledDeviceType: 76 LastRedirectedDeviceType: 65535 LocalConnectionState: 2 EventCause: 22 NumNamedVars: 0 NumNamedArrays: 0 ConnectionDeviceID: 2006 AlertingDeviceID: 2006 CallingDeviceID: 2005 CalledDeviceID: 2006 LastRedirectedDeviceID: SecondaryCallID: 16777508 ...	Yes	Yes	Yes	The call is delivered to the agent device (Extension) 2006.

6 The agent at extension 2006 answers the call.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: AGENT_STATE_EVENT CSQState: 4 StateDuration: 0 CSQID: -1 AgentState: 4 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2006 AgentID: agt2006 NextAgentState: 3	Yes	Yes	Yes	When the agent on extension 2006 answers the call, the agent state is changed to Talking.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	

`MsgType: CALL_DATA_UPDATE_EVENT`
`  NumCTIClients: 0`
`  NumNamedVars: 0`
`  NumNamedArrays: 0`
`  CallType: 10`
`  ConnectionDeviceType: 0`
`  CallID: 16777507`
`  CalledPartyDisposition: 0`
`  ConnectionDeviceID: 2005`
`...`
	Yes	Yes	Yes	Call data is sent to client programs.

`MsgType: CALL_ESTABLISHED_EVENT`
`  ConnectionDeviceType: 0`
`  CallID: 16777507`
`  ApplicationID: -1`
`  CSQID: 1`
`  AnsweringDeviceType: 76`
`  CallingDeviceType: 76`
`  CalledDeviceType: 76`
`  LastRedirectedDeviceType: 73`
`  LocalConnectionState: 3`
`  EventCause: 22`
`  ConnectionDeviceID: 2006`
`  AnsweringDeviceID: 2006`
`  CallingDeviceID: 2005`
`  CalledDeviceID: 2006`
`  LastRedirectedDeviceID: 9019`
	Yes	Yes	Yes	When the agent on extension 2006 answers the call, the call is connected.

`MsgType: CallDequeuedEvent`
`  ConnectionDeviceType: 65535`
`  CallID: 16777507`
`  ApplicationID: -1`
`  QueueDeviceType: 0`
`  NumQueued: 0`
`  NumCSQs: 0`
`  LocalConnectionState: 3`
`  EventCause: 65535`
`  ConnectionDeviceID:`
`  QueueDeviceID:`
	Yes	Yes	Yes	Unified CCX removes the call from the CSQ.

7 The call is disconnected from the CTI port.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ReleasingingDeviceType: 73 LocalConnectionState: 65535 EventCause: 28 ConnectionDeviceID: 9019 ReleasingDeviceID: 9019	Yes	Yes	Yes	The call is disconnected from CTI port 9019.

8 The caller on extension 2005 hangs up.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777507 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2005 ReleasingDeviceID: 2005	Yes	Yes	Yes	The call is disconnected from the caller on extension 2005.
MsgType: AGENT_STATE_EVENT CSQState: 3 StateDuration: 0 CSQID: -1 AgentState: 3 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2006 AgentID: agt2006 NextAgentState: 65535	Yes	Yes	Yes	The state of Agent 2006 is changed back to Ready since the agent has the *Automatic Available* checkbox enabled.

Server sends the following message(s)	Message(s) received by			Notes
	Bridge mode clients	Agent mode 2006	Caller 2005	

`MsgType: CALL_CONNECTION_CLEARED_EVENT`
`ConnectionDeviceType: 0`
`CallID: 16777507`
`ReleasingDeviceType: 76`
`LocalConnectionState: 65535`
`EventCause: 65535`
`ConnectionDeviceID: 2006`
`ReleasingDeviceID: 2006`
	Yes	Yes	Yes	The call is disconnected from device 2006.

`MsgType: CALL_CLEARED_EVENT`
`ConnectionDeviceType: 0`
`CallID: 16777507`
`LocalConnectionState: 0`
`EventCause: 1014`
`ConnectionDeviceID: 2005`
	Yes	Yes	Yes	Since there are no more connections in the call, the call is deleted.

`MsgType: END_CALL_EVENT`
`ConnectionDeviceType: 0`
`ConnectionCallID: 16777507`
`ConnectionDeviceID:`
	Yes	Yes	Yes	This message indicates the call is terminated.

Example Message Flow for a Transferred Call

This example message flow starts at step 8 of the preceding message flow. Then, the agent on extension 2006 transfers the call to another agent instead of hanging up the call.

Note For each of the tables in this section, the value in the **Caller 2005** column is Yes if extension 2005 is a logged in agent, No if extension 2005 is not a logged in agent.

1 Agent 2006 consults Agent 2007.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_HELD_EVENT ConnectionDeviceType: 0 CallID: 16777507 HoldingingDeviceType: 76 LocalConnectionState: 4 EventCause: 65535 ConnectionDeviceID: 2006 HoldingDeviceID: 2006	Yes	Yes	No	No	The original call between 2005 and 2006 is on hold.
MsgType: BEGIN_CALL_EVENT NumCTIClients: 0 NumNamedVars: 0 NumNamedArrays: 0 CallType: 12 ConnectionDeviceType: 0 CallID: 16777535 CalledPartyDisposition: 0 ConnectionDeviceID: 2006 ANI: 2006...	Yes	Yes	No	No	The agent on extension 2006 begins a new consult call to the agent on extension 2007. If the ANI is empty use the CallingDeviceID instead of ANI.
MsgType: CALL_SERVICE_INITIATED_EVENT ConnectionDeviceType: 0 CallID: 16777535 ApplicationID: -1 CSQID: 1 CallingDeviceType: 76 LocalConnectionState: 1 EventCause: 65535 ConnectionDeviceID: 2006 CallingDeviceID: 2006	Yes	Yes	No	No	
MsgType: AGENT_STATE_EVENT CSQState: 8 StateDuration: 0 CSQID: -1 AgentState: 8 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2007 AgentID: agt2007 NextAgentState: 65535	Yes	No	Yes	No	

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_ORIGINATED_EVENT ConnectionDeviceType: 0 CallID: 16777535 ApplicationID: -1 CSQID: 1 CallingDeviceType: 76 CalledDeviceType: 76 LocalConnectionState: 1 EventCause: 65535 ConnectionDeviceID: 2006 CallingDeviceID: 2006 CalledDeviceID: 2007	Yes	Yes	No	No	
MsgType: CALL_DELIVERED_EVENT ConnectionDeviceType: 0 CallID: 16777535 ApplicationID: -1 CSQID: 1 AlertingDeviceType: 76 CallingDeviceType: 76 CalledDeviceType: 76 LastRedirectedDeviceType: 65535 LocalConnectionState: 2 EventCause: 22 NumNamedVars: 0 NumNamedArrays: 0 ConnectionDeviceID: 2007 AlertingDeviceID: 2007 CallingDeviceID: 2006 CalledDeviceID: 2007 LastRedirectedDeviceID: SecondaryCallID: 0 ...	Yes	Yes	Yes	No	

2 Agent 2007 answers the consult call.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: AGENT_STATE_EVENT CSQState: 4 StateDuration: 0 CSQID: -1 AgentState: 4 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2007 AgentID: agt2007 NextAgentState: 3	Yes	No	Yes	No	
MsgType: AGENT_STATE_EVENT CSQState: 4 StateDuration: 0 CSQID: -1 AgentState: 4 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2007 AgentID: agt2007 NextAgentState: 2	Yes	No	Yes	No	
MsgType: CALL_ESTABLISHED_EVENT ConnectionDeviceType: 0 CallID: 16777535 ApplicationID: -1 CSQID: 1 AnsweringDeviceType: 76 CallingDeviceType: 76 CalledDeviceType: 76 LastRedirectedDeviceType: 65535 LocalConnectionState: 3 EventCause: 22 ConnectionDeviceID: 2007 AnsweringDeviceID: 2007 CallingDeviceID: 2006 CalledDeviceID: 2007 LastRedirectedDeviceID:	Yes	Yes	No	No	

3 Agent 2006 completes the transfer of Caller 2005 to Agent 2007.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_DATA_UPDATE_EVENT NumCTIClients: 0 NumNamedVars: 0 NumNamedArrays: 0 CallType: 4 ConnectionDeviceType: 0 CallID: 16777533 CalledPartyDisposition: 0 ConnectionDeviceID: 2005 ...	Yes	Yes	No	Yes	
MsgType: CALL_TRANSFERRED_EVENT PrimaryDeviceType: 0 PrimaryCallID: 16777533 NumParties: 2 SecondaryDeviceType: 0 SecondaryCallID: 16777535 TransferringDeviceType: 76 TransferredDeviceType: 76 LocalConnectionState: 3 EventCause: 32 PrimaryDeviceID: 2006 SecondaryDeviceID: 2007 TransferringDeviceID: 2006 TransferredDeviceID: 2007 ConnectedPartyDCallID[0]: 16777533 ConnectedPartyDeviceType[0]: 0 ConnectedPartyDeviceID[0]: 2005 ConnectedPartyDCallID[1]: 16777533 ConnectedPartyDeviceType[1]: 0 ConnectedPartyDeviceID[1]: 2007	Yes	Yes	Yes	Yes	After the transfer, the consult call is merged with the original call.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_ESTABLISHED_EVENT ConnectionDeviceType: 0 CallID: 16777533 ApplicationID: -1 CSQID: 1 AnsweringDeviceType: 76 CallingDeviceType: 76 CalledDeviceType: 76 LastRedirectedDeviceType: 76 LocalConnectionState: 3 EventCause: 22 ConnectionDeviceID: 2005 AnsweringDeviceID: 2007 CallingDeviceID: 2005 CalledDeviceID: 2007 LastRedirectedDeviceID: 2006	Yes	No	Yes	Yes	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777535 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2006 ReleasingDeviceID: 2006	Yes	Yes	Yes	No	Device 2006 is dropped off the call after the transfer.
MsgType: AGENT_STATE_EVENT CSQState: 3 StateDuration: 0 CSQID: -1 AgentState: 3 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2006 AgentID: agt2006 NextAgentState: 65535	Yes	Yes	No	No	The state of the agent on extension 2006 is changed back to Ready.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777535 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2007 ReleasingDeviceID: 2007	Yes	No	No	No	The agent on device 2007 hangs up the call.
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777535 LocalConnectionState: 0 EventCause: 1014 ConnectionDeviceID: 2006	Yes	No	No	No	The consult call is deleted.
MsgType: END_CALL_EVENT ConnectionDeviceType: 0 ConnectionCallID: 16777535 ConnectionDeviceID:	Yes	No	No	No	The consult call is ended.

4 Caller 2005 ends the call.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777533 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2005 ReleasingDeviceID: 2005	Yes	No	Yes	Yes	The caller on extension 2005 hangs up.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: AGENT_STATE_EVENT CSQState: 2 StateDuration: 0 CSQID: -1 AgentState: 2 EventReasonCode: 32755 NumCSQs: 0 AgentExtension: 2007 AgentID: agt2007 NextAgentState: 65535	Yes	No	Yes	Yes	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777533 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2007 ReleasingDeviceID: 2007	Yes	No	Yes	Yes	
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777533 LocalConnectionState: 0 EventCause: 1014 ConnectionDeviceID: 2005	Yes	No	No	No	
MsgType: END_CALL_EVENT ConnectionDeviceType: 0 ConnectionCallID: 16777533 ConnectionDeviceID:	Yes	No	No	No	The call is terminated.

Example Message Flow for a Conferenced Call

In this example message flow, an Agent 2006 consults Agent 2007 to make a conferenced call. The call events in this message flow are similar to those in a consult-transfer up to the point before agent 2006 initiatates the conference.

Note For each of the tables in this section, the value in the **Caller 2005** column is Yes if extension 2005 is a logged in agent, No if extension 2005 is not a logged in agent.

1 Agent 2006 consults Agent 2007.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	

`MsgType: CALL_DATA_UPDATE_EVENT`
`  NumCTIClients: 0`
`  NumNamedVars: 0`
`  NumNamedArrays: 0`
`  CallType: 15`
`  ConnectionDeviceType: 0`
`  CallID: 16777539`
`  CalledPartyDisposition: 0`
`  ConnectionDeviceID: 2005`
`...`
	Yes	No	Yes	Yes	

`MsgType: CALL_CONFERENCED_EVENT`
`  PrimaryDeviceType: 0`
`  PrimaryCallID: 16777507`
`  NumParties: 3`
`  SecondaryDeviceType: 0`
`  SecondaryCallID: 16777541`
`  ControllerDeviceType: 76`
`  AddedPartyDeviceType: 76`
`  LocalConnectionState: 3`
`  EventCause: 17`
`  PrimaryDeviceID: 2006`
`  SecondaryDeviceID: 2007`
`  ControllerDeviceID: 2006`
`  AddedDeviceID: 2007`
`  ConnectedPartyDCallID[0]: 16777539`
`  ConnectedPartyDeviceType[0]: 0`
`  ConnectedPartyDeviceID[0]: 2005`
`  ConnectedPartyDCallID[1]: 16777539`
`  ConnectedPartyDeviceType[1]: 0`
`  ConnectedPartyDeviceID[1]: 2006`
`  ConnectedPartyDCallID[2]: 16777539`
`  ConnectedPartyDeviceType[2]: 0`
`  ConnectedPartyDeviceID[2]: 2007`
	Yes	Yes	Yes	Yes	The original call and the consult call are merged. The original call ID is used to identify the conferenced call.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777541 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2006 ReleasingDeviceID: 2006	Yes	Yes	Yes	No	The consult call is cleared at extension 2006.
MsgType: CALL_ESTABLISHED_EVENT ConnectionDeviceType: 0 CallID: 16777539 ApplicationID: -1 CSQID: 1 AnsweringDeviceType: 65535 CallingDeviceType: 76 CalledDeviceType: 65535 LastRedirectedDeviceType: 76 LocalConnectionState: 3 EventCause: 22 ConnectionDeviceID: 2005 AnsweringDeviceID: Unknown CallingDeviceID: 2005 CalledDeviceID: Unknown LastRedirectedDeviceID: 2006	Yes	Yes	Yes	Yes	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777541 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2007 ReleasingDeviceID: 2007	Yes	No	Yes	No	
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777541 LocalConnectionState: 0 EventCause: 1014 ConnectionDeviceID: 2006	YYeses	No	No	No	

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: END_CALL_EVENT ConnectionDeviceType: 0 ConnectionCallID: 16777541 ConnectionDeviceID:	Yes	No	No	No	Consult call is deleted.

2 Agent 2005 ends the call.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777539 ReleasingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2005 ReleasingDeviceID: 2005	Yes	Yes	Yes	Yes	

3 Agent 2006 ends the call.

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: AGENT_STATE_EVENT CSQState: 3 StateDuration: 0 CSQID: -1 AgentState: 3 EventReasonCode: 0 NumCSQs: 0 AgentExtension: 2006 AgentID: agt2006 NextAgentState: 65535	Yes	Yes	No	No	

Server sends the following message(s)	Message(s) received by				Notes
	Bridge mode clients	Agent mode 2006	Agent mode 2007	Caller 2005	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777539 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2006 ReleasingDeviceID: 2006	Yes	Yes	Yes	No	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777539 ReleasingingDeviceType: 76 LocalConnectionState: 65535 EventCause: 65535 ConnectionDeviceID: 2007 ReleasingDeviceID: 2007	Yes	No	Yes	No	
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: 0 CallID: 16777539 LocalConnectionState: 0 EventCause: 1014 ConnectionDeviceID: 2005	Yes	No	No	No	
MsgType: END_CALL_EVENT ConnectionDeviceType: 0 ConnectionCallID: 16777539 ConnectionDeviceID:	Yes	No	No	No	

Example Message Flows for an Outbound Option Call

This section has the following three message flows:

- The Outbound Subsystem Picks an Agent to Place a Call to a Customer Contact, on page 237
- The Agent Skips the Contact, on page 241
- The Agent Rejects the Reservation Call, on page 243

The Outbound Subsystem Picks an Agent to Place a Call to a Customer Contact

1 The agent is reserved.

Server sends the following message(s)	
MsgType: AGENT_STATE_EVENT CSQState: RESERVED StateDuration: 0 CSQID: N/A AgentState: RESERVED	EventReasonCode: 0 NumCSQs: 0 AgentID: chli4020 AgentExtension: 4020

2 Unified CCX creates a reservation call.

Server sends the following message(s)	
MsgType: BEGIN_CALL_EVENT NumCTIClients: 0 NumNamedVariables: 8 NumNamedArrays: 0 CallType: CALLTYPE_RESERVATION ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800097 CalledPartyDisposition: CD_INVALID ConnectionDeviceID: 420	ExpandedCallContxt: <TotalBytes: 161> BACampaign: LoadCamp BACampaignID: BAStatus: DO BAResponse: user.layout: OODefault BAAccountNumber: AC111 BADialedListID: 20779 BABuddyName: John,Doe
MsgType: CALL_DELIVERED_EVENT: CallID: 16800097 LocalConnectionState: LCS_ALERTIN AlertingDevice: 4020	ConnectionDeviceID: 4020 CalledDeviceID: 4020 EventCause: CEC_NEW_CALL

3 The agent accepts the customer contact through the SET_CALL_DATA_REQ mesage with BAResponse = Accept.

Note This message is not listed here since the outbound example call flows displayed here only show the server generated messages, and not the agent generated messages.

4 Unified CCX responds to the accept with a Call_DATA_UPDATE_EVENT message indicating the agent accepts the reservation call.

Server sends the following message(s)	
MsgType: CALL_DATA_UPDATE_EVENT NumCTIClients: 0 NumNamedVariables: 1 NumNamedArrays: 0 CallType: CALLTYPE_RESERVATION ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 NewConnectionDeviceType: CONNECTION_ID_STATIC NewConnectionCallID: 16800096	CalledPartyDisposition: CD_INVALID CampaignID: 0 QueryRuleID: 0 ConnectionDeviceID: 4020 NewConnectionDeviceID: 4020 ExpandedCallContxt: <TotalBytes: 20> BAResponse: Accept CustomerPhoneNumber: CustomerAccountNumber:

5 Unified CCX makes a call from the agent phone to the customer. Notice that this call ID is different from the reservation call ID.

Server sends the following message(s)	
MsgType:BEGIN_CALL_EVENT NumCTIClients: 0 NumNamedVariables: 10 NumNamedArrays: 0 CallType: CALLTYPE_PREVIEW ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 CalledPartyDisposition: CD_INVALID ConnectionDeviceID: 4020 DialedNumber: 4022 ExpandedCallContxt: <TotalBytes: 211>	BACampaign: LoadCamp BACampaignID: 2 BATimeZone: -00360 BAResponse: Accept BAStatus: CO BACustomerNumber: 4022 user.layout: OODefault BAAccountNumber: AC111 BADialedListID: 20779 BABuddyName: John,Doe

6 The reservation call is terminated.

Server sends the following message(s)	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800097 ReleasingDeviceType: DEVID_AGENT_DEVICE_IDENTIFIER	LocalConnectionState: LCS_NONE EventCause: CEC_NONE ConnectionDeviceID: 4020 ReleasingDeviceID: 4020
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 1680097	LocalConnectionState: LCS_NULL EventCause: CEC_NONE ConnectionDeviceID: 4020

Server sends the following message(s)	
MsgType: END_CALL_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC	ConnectionCallID: 16800097 ConnectionDeviceID:

7 This message indicates that the preview call has started. Noticed the BAStatus is changed to CO and the CallType is changed to Preview.

Server sends the following message(s)	

MsgType: CALL_DATA_UPDATE_EVENT
  NumCTIClients: 0
  NumNamedVariables: 10
  NumNamedArrays: 0
  CallType: CALLTYPE_PREVIEW
  ConnectionDeviceType:
CONNECTION_ID_STATIC
  ConnectionCallID: 16800096
  NewConnectionDeviceType:
CONNECTION_ID_STATIC
  NewConnectionCallID: 16800096
  CalledPartyDisposition: CD_INVALID
  CampaignID: 0
  QueryRuleID: 0
  ConnectionDeviceID: 4020
  NewConnectionDeviceID: 4020
	ExpandedCallContxt: <TotalBytes: 211>
	BACampaign: LoadCamp
	BACampaignID: 2
	BATimeZone: -00360
	BAResponse: Accept
	BAStatus: CO
	BACustomerNumber: 4022
	user.layout: OODefault
	BAAccountNumber: AC111
	BADialedListID: 20779
	BABuddyName: John,Doe
	CustomerPhoneNumber:
	CustomerAccountNumber:

8 Call events as a result of call control activities.

Server sends the following message(s)	

MsgType: CALL_DATA_UPDATE_EVENT
  NumCTIClients: 0
  NumNamedVariables: 1
  NumNamedArrays: 0
  CallType: CALLTYPE_PREVIEW
  ConnectionDeviceType:
CONNECTION_ID_STATIC
  ConnectionCallID: 16800096
  NewConnectionDeviceType:
CONNECTION_ID_STATIC
  NewConnectionCallID: 16800096
	CalledPartyDisposition: CD_INVALID
	CampaignID: 0
	QueryRuleID: 0
	ConnectionDeviceID: 4020
	NewConnectionDeviceID: 4020
	ExpandedCallContxt: <TotalBytes: 14>
	BAStatus: CO
	CustomerPhoneNumber:

Server sends the following message(s)	
MsgType: CALL_DATA_UPDATE_EVENT NumCTIClients: 0 NumNamedVariables: 1 NumNamedArrays: 0 CallType: CALLTYPE_PREVIEW ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 NewConnectionDeviceType: CONNECTION_ID_STATIC NewConnectionCallID: 16800096	CalledPartyDisposition: CD_INVALID CampaignID: 0 QueryRuleID: 0 ConnectionDeviceID: 4020 NewConnectionDeviceID: 4020 ExpandedCallContxt: <TotalBytes: 20> BATimeZone: -00360 CustomerPhoneNumber:
MsgType: CALL_DATA_UPDATE_EVENT NumCTIClients: 0 NumNamedVariables: 1 NumNamedArrays: 0 CallType: CALLTYPE_PREVIEW ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 NewConnectionDeviceType: CONNECTION_ID_STATIC NewConnectionCallID: 16800096	CalledPartyDisposition: CD_INVALID CampaignID: 0 QueryRuleID: 0 ConnectionDeviceID: 4020 NewConnectionDeviceID: 4020 ExpandedCallContxt: <TotalBytes: 24> BACustomerNumber: 4022 CustomerPhoneNumber:
MsgType: CALL_SERVICE_INITIATED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 ApplicationID: N/A CSQID: N/A	CallingDeviceType: DEVID_AGENT_DEVICE_IDENTIFIER LocalConnectionState: LCS_INITIATE EventCause: CEC_NONE ConnectionDeviceID: 4020 CallingDeviceID: 4020
MsgType: AGENT_STATE_EVENT CSQState: TALKING StateDuration: 0 CSQID: N/A AgentState: TALKING	EventReasonCode: 0 NumCSQs: 0 AgentID: chli4020 AgentExtension: 4020 NextAgentState: 5
MsgType: CALL_ORIGINATED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 ApplicationID: N/A CSQID: N/A CallingDeviceType: DEVID_AGENT_DEVICE_IDENTIFIER	CalledDeviceType: DEVID_DEVICE_IDENTIFIER LocalConnectionState: LCS_INITIATE EventCause: CEC_NONE ConnectionDeviceID: 4020 CallingDeviceID: 4020 CalledDeviceID: 4022

Server sends the following message(s)	
MsgType: CALL_ESTABLISHED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 ApplicationID: N/A CSQID: N/A AnsweringDeviceType: DEVID_NONE CallingDeviceType: DEVID_AGENT_DEVICE_IDENTIFIER CalledDeviceType: DEVID_NONE	LastRedirectDeviceType: DEVID_NONE LocalConnectionState: LCS_CONNECT EventCause: CEC_NEW_CALL ConnectionDeviceID: 4020 AnsweringDeviceID: Unknown CallingDeviceID: 4020 CalledDeviceID: Unknown

9 The outbound call is terminated.

Server sends the following message(s)	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800096 ReleasingDeviceType: DEVID_AGENT_DEVICE_IDENTIFIER	LocalConnectionState: LCS_NONE EventCause: CEC_NONE ConnectionDeviceID: 4020 ReleasingDeviceID: 4020
MsgType: AGENT_STATE_EVENT CSQState: WORK_NOT_READY StateDuration: 0 CSQID: N/A AgentState: WORK_NOT_READY	EventReasonCode: 0 NumCSQs: 0 AgentID: chli4020 AgentExtension: 4020
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIc ConnectionCallID: 16800096	LocalConnectionState: LCS_NULL EventCause: CECX_DROP_HANDLED_OTHER ConnectionDeviceID: 4020

The Agent Skips the Contact

1 The agent skips the contact. The agent sets BAResponse = Skip, and the server replies as follows.

Server sends the following message(s)	
MsgType: AGENT_STATE_EVENT CSQState: AVAILABLEState Duration: 0 CSQID: N/A AgentState: AVAILABLE	EventReasonCode: 0 NumCSQs: 0 AgentID: chli4020 AgentExtension: 4020
MsgType: END_CALL_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC	ConnectionCallID: 16800095 ConnectionDeviceID:

2 The agent becomes available.

Server sends the following message(s)	
MsgType: CALL_DATA_UPDATE_EVENT NumCTIClients: 0 NumNamedVariables: 1 NumNamedArrays: 0 CallType: CALLTYPE_RESERVATION ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800098 NewConnectionDeviceType: CONNECTION_ID_STATIC NewConnectionCallID: 16800098 CalledPartyDisposition: CD_INVALID	CampaignID: 0 QueryRuleID: 0 ConnectionDeviceID: 4020 NewConnectionDeviceID: 4020 ExpandedCallContxt: <TotalBytes: 18> BAResponse: Skip CustomerPhoneNumber: CustomerAccountNumber:
MsgType: AGENT_STATE_EVENT CSQState: AVAILABLE StateDuration: 0 CSQID: N/A AgentState: AVAILABLE	EventReasonCode: 0 NumCSQs: 0 AgentID: chli4020 AgentExtension: 4020

3 The reservation call is cleared.

Server sends the following message(s)	
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800099 ReleasingDeviceType: DEVID_AGENT_DEVICE_IDENTIFIER	LocalConnectionState: LCS_NONE EventCause: CEC_NONE ConnectionDeviceID: 4020 ReleasingDeviceID: 4020
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800099	LocalConnectionState: LCS_NULL EventCause: CEC_NONE ConnectionDeviceID: 4020

Server sends the following message(s)	
MsgType: END_CALL_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC	ConnectionCallID: 16800099 ConnectionDeviceID:

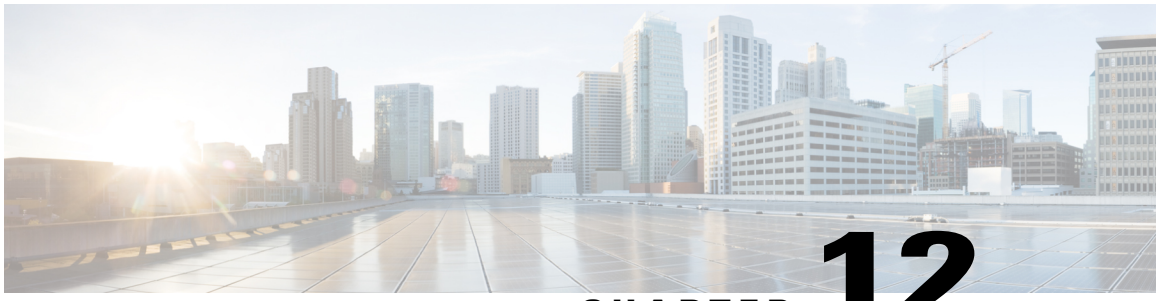
The Agent Rejects the Reservation Call

1 The agent sets BAResponse = *Reject* and the server responds as follows.

Server sends the following message(s)	
MsgType: CALL_DATA_UPDATE_EVENT NumCTIClients: 0 NumNamedVariables: 1 NumNamedArrays: 0 CallType: CALLTYPE_RESERVATION ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800100 NewConnectionDeviceType: CONNECTION_ID_STATIC NewConnectionCallID: 16800100	CalledPartyDisposition: CD_INVALID CampaignID: 0 QueryRuleID: 0 ConnectionDeviceID: 4020 NewConnectionDeviceID: 4020 ExpandedCallContxt: <TotalBytes: 20> BAResponse: Reject CustomerPhoneNumber: CustomerAccountNumber:
MsgType: CALL_CONNECTION_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800101 ReleasingDeviceType: DEVID_AGENT_DEVICE_IDENTIFIER	LocalConnectionState: LCS_NONE EventCause: CEC_NONE ConnectionDeviceID: 4020 ReleasingDeviceID: 4020

2 The reservation call is cleared.

Server sends the following message(s)	
MsgType: CALL_CLEARED_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC ConnectionCallID: 16800101	LocalConnectionState: LCS_NULL EventCause: CEC_NONE ConnectionDeviceID: 4020
MsgType: END_CALL_EVENT ConnectionDeviceType: CONNECTION_ID_STATIC	ConnectionCallID: 16800101 ConnectionDeviceID:



CHAPTER 12

Document History

This section includes the changes that have gone into each chapter/ topic of the document:

Version	Date	Affected chapter	History of changes
1.2	September 13, 2007	Chapter 9	Modified the information in Table 35: CONFIG_REQUEST_EVENT Fixed Part Message Body Format, on page 87 from: 1: Application Information 4: Agent Information**To**: 1: Agent Information 4: Application Information
1.2	September 13, 2007	Chapter 1	Table 1, included the CTI Version(s) supported for CRS 6.0
1.2	September 14, 2007	Chapter 9	Table 30: CONFIG_REQUEST_KEY_EVENT Fixed Part Message Body Format, on page 85 included a footnote for supported protocol version.
1.3	December 14, 2007	Chapter 5	Table 3: Agent States and Their Message Values, on page 32, removed AGENT_STATE_LOGIN Figure 7: Example Agent State Transitions, on page 34, edited the image to reflect the removal of AGENT_STATE_LOGIN.
1.4	January 28, 2008	Chapter 5	Table 3: Agent States and Their Message Values, on page 32, re-included AGENT_STATE_LOGIN with a modified description.
1.5	April 30, 2008	Chapter 9	In section CALL_CLEARED_EVENT, on page 118, 1) Marked **ConnectionDeviceType** as a **reserved field** with the following information in the **Value** column**:** "This value is set to 0". 2) Marked **ConnectionDeviceID** as a **reserved field**. with the following information in the **Value** column**:** "This value is left blank".

Version	Date	Affected chapter	History of changes
1.6	June 17, 2008	Chapter 9	Table 43: CONFIG_CSQ_EVENT Floating Part Message Body Format, on page 90 Marked the reserved field 216 (after Description) as "MRDomainID" field 216. Modified the values as follows: 0=Voice and 1=EMail
			Table 72: CALL_DELIVERED_EVENT Floating Part Message Body Format, on page 112 Modified the optional DNIS floating field ID from [25] to [10].
		Chapter 3	Included a new section on Connect CTI Sessions in a Clustered Unified CCX, on page 21.
1.7	September 9, 2008	Chapter 9	Table 164: QUERY_QUEUE_STATISTICS_CONF Fixed Part Message Body Format, on page 170 Moved “HandledCallsToday” field after the “OldestCallInQueue”.
			Table 117: CONFERENCE_CALL_REQ Fixed Part Message Body Format, on page 145 Modified the agentInstrument field for CONFERENCE_CALL_REQ to 'required' instead of 'optional'.
			Table 138: TRANSFER_CALL_REQ Fixed Part Message Body Format, on page 156 Modified the agentInstrument field for TRANSFER_CALL_REQ to 'required' instead of 'optional'.
1.8	March 6, 2009	Chapter 10	Added new Table 1 - "Reason Codes for Agent State Change"
		Chapter 9	Table 56: AGENT_STATE_EVENT Fixed Part Message Body Format, on page 98: Provided reference to Table 1 in the Description column for the EventReasonCode field.
1.9	June 24, 2009	Chapter 5	Table 3: Agent States and Their Message Values, on page 32: Added a new state called BUSY_OTHER.

Version	Date	Affected chapter	History of changes
1.10	June 30, 2009	Chapter 1	Table 1: Included the CTI Version(s) supported for Cisco Unified CCX 8.0(1).
		Chapter 9	Table 15: OPEN_CONF Floating Part Message Body Format, on page 76: Added the following new floating fields for CTI protocol versions 14 and later: • FltPeripheralID • MultilineAgentControl • NumPeripherals
			Added a new Primary.Actual field format section and included the general rules for this field after "Call-Event Messages" section for CTI protocol version 14. Updated the field description for the Call-Event messages that are affected by the general rules for the Primary.Actual field format.
			Added a new Reserved field in Table 128: MAKE_CALL_REQ Fixed Part Message Body Format, on page 152, which is present in CTI Protocol Version 14.
		Chapter 10	Table 189: DeviceType Values , on page 192: Added the following DeviceType value: DEVID_NON_ACD_DEVICE_IDENTIFIER Table 196: Control Failure Code Values , on page 198: Added the following new Control Failure code values: • CF_MAXIMUM_LINE_EXCEEDED • CF_SHARED_LINES_NOT_SUPPORTED • CF_EXTENSION_NOT_UNIQUE
1.11	July 1, 2009	Chapter 10	Table 1: Added the following new error code values: • CCX_AGENT_DEVICE_IPv6 • CCX_TRANSFER_FAILED_DUE_TO_CCM_PROBLEM • CCX_CONFERENCE_FAILED_DUE_TO_CCM_PROBLEM

INDEX

A

Agent State Masks 82
AGENT_STATE_EVENT 98
ALTERNATE_CALL_CONF 141
ALTERNATE_CALL_REQ 139
ANSWER_CALL_CONF 142
ANSWER_CALL_REQ 141
Audio Codec Type Values 196

B

BAD_CALL_CONF 168
BAD_CALL_REQ 167
BAResponse ECC variable 51
BEGIN_CALL_EVENT 105

C

CALL_CLEARED_EVENT 118
CALL_CONFERENCED_EVENT 123
CALL_CONNECTION_CLEARED_EVENT 119
CALL_DATA_UPDATE_EVENT 108
CALL_DELIVERED_EVENT 111
CALL_DEQUEUED_EVENT 133
CALL_DIVERTED_EVENT 128
CALL_ESTABLISHED_EVENT 114
CALL_FAILED_EVENT 121
CALL_HELD_EVENT 116
CALL_ORIGINATED_EVENT 120
CALL_QUEUED_EVENT 131
CALL_RETRIEVED_EVENT 117
CALL_SERVICE_INITIATED_EVENT 129
CALL_TRANSFERRED_EVENT 125
call-event message flows 217
CallType Values 197
CCX Status Values 208
CLEAR_CALL_CONF 143
CLEAR_CALL_REQ 142
CLEAR_CONNECTION_CONF 144
CLEAR_CONNECTION_REQ 143
CLOSE_CONF 79
CLOSE_REQ 79
CONFERENCE_CALL_CONF 146
CONFERENCE_CALL_REQ 144
CONFIG_AGENT_EVENT 92
CONFIG_APPLICATION_EVENT 90
CONFIG_BEGIN_EVENT 87
CONFIG_CSQ_EVENT 89
CONFIG_DEVICE_EVENT 93
CONFIG_END_EVENT 88
CONFIG_KEY_EVENT 85
CONFIG_REQUEST_EVENT 86
CONFIG_REQUEST_KEY_EVENT 85
ConnectionDeviceID Type Values 196
CONSULTATION_CALL_CONF 150
CONSULTATION_CALL_REQ 148
Control Failure Code Values 198
CONTROL_FAILURE_CONF 139
CRSErrorCode Values 204

D

debugging a client program 215
DeviceIDType Values 192
Disposition Values 209

E

END_CALL_EVENT 107
error handling 29
Error Status Codes 209
EventCause Values 193

F

FAILURE_EVENT 84
FAILUTE_CONF 83

FieldDataID Values 187
Floating FieldDataID Values 187

H

HEARTBEAT_CONF 77
HEARTBEAT_REQ 77
HOLD_CALL_CONF 151
HOLD_CALL_REQ 150

L

LocalConnection State Values 193

M

MAKE_CALL_CONF 153
MAKE_CALL_REQ 151
Message Field Data Types 184
message flow examples 217
Message Header Data Format 185

N

NAMEDARRAY Data Format 186
NAMEDVARIABLE Data Format 185

O

OPEN_CONF 75
OPEN_REQ 73
OPEN_REQUEST 79
Outbound Feature 49, 50, 51
about 49
and BAResponse ECC variable 51
and call classification 50
call event sequence 50
ECC variables used with 49

Q

QUERY_AGENT_QUEUE_STATISTICS_REQ 172
QUERY_AGENT_STATE _CONF 101
QUERY_AGENT_STATISTICS_CONF 172
QUERY_DEVICE_INFO_CONF 174
QUERY_DEVICE_INFO_REQ 173
QUERY_QUEUE_STATISTICS_CONF 169
QUERY_QUEUE_STATISTICS_REQ 169
QUERY_SUMMARY_STATISTICS_CONF 175
QUERY_SUMMARY_STATISTICS_REQ 175

R

RECONNECT_CALL_CONF 155
RECONNECT_CALL_REQ 154
RETRIEVE_CALL_CONF 156
RETRIEVE_CALL_REQ 155
RTP_STARTED_EVENT 134
RTP_STOPPED_EVENT 136

S

SEND_DTMF_SIGNAL_CONF 160
SEND_DTMF_SIGNAL_REQ 159
SET_AGENT_STATE_CONF 103
SET_AGENT_STATE_REQ 102
SET_CALL_DATA_CONF 162
SET_CALL_DATA_REQ 160
SNAPSHOT_CALL_CONF 178
SNAPSHOT_CALL_REQ 177
SNAPSHOT_DEVICE_CONF 181
SNAPSHOT_DEVICE_REQ 181
Special Values 208
SUPERVISE_CALL_CONF 164
SUPERVISE_CALL_REQ 162
SUPERVISOR_ASSIST_CONF 166
SUPERVISOR_ASSIST_EVENT 167
SUPERVISOR_ASSIST_REQ 165
SYSTEM_EVENT 77
SystemEventID Values 195

T

TEAM_CONFIG_CONF 97
TEAM_CONFIG_EVENT 95
TEAM_CONFIG_REQ 96
trace files 215
TRANSFER_CALL_CONF 158
TRANSFER_CALL_REQ 156

U

Unified CCX Status Values 208
Unsolicited Call-Event Message Masks 80