# Aspect® Unified IP® CTI Portal EDK Guide

7.2

The content of this publication is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Aspect Software, Inc. Aspect Software, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this publication. Aspect Software, Inc. reserves the right to change information in this publication without notice, as a result of product enhancements or other reasons.

Aspect, Aspect Software, Advanced List Management, Advanced Voice Portal, Aspect Application Foundation, Blended Interaction, CallCenter, Aspect Contact, Contact Server, Aspect Customer Self Service, DataMart, Enterprise Contact Center, Aspect On Demand, Optimized Collections, Performance Management, Productive Workforce, Quality Management, Real-Time Reporting, Seamless Customer Service, Aspect Speech Analytics, Spectrum, Streamlined Collections, Unified IP, Unified Command and Control, Unified Outreach, Uniphi Connect, Workforce Management, and Workforce Optimization are either trademarks or registered trademarks of Aspect Software, Inc., in the United States and/or other countries. Use of any Aspect Software, Inc. trademark is prohibited unless expressly approved in writing in advance by an authorized representative of Aspect Software, Inc. Microsoft Windows®, Microsoft SQL Server®, Microsoft Lync™ are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Any other brands, product names, company names, logos, trademarks, and/or service marks used in this publication are the property of their respective owners.

The works of authorship contained in this publication, including but not limited to all design, text and images and the software described herein, are owned, except as otherwise expressly stated, by Aspect Software, Inc., or its affiliates or licensors. The entire contents of this publication are protected by United States' and worldwide copyright laws and treaty provisions. In accordance with these terms, except as stated above, you may not copy, reproduce, modify, use, republish, upload, post, transmit or distribute in any way material from the publication. Further, you may not copy, modify or display any of Aspect Software, Inc.'s or its affiliates' trademarks, tradenames or logos appearing in this publication in any way without Aspect Software, Inc.'s express written consent. Aspect Software, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this publication. Except as permitted by such license, no part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, or otherwise, without the prior written permission of Aspect Software, Inc.

RESTRICTED RIGHTS LEGEND
This publication is provided with "Restricted Rights". No part of this publication may be photocopied, reproduced or transmitted, in any form or by any means, without the prior written consent of Aspect Software, Inc. Use, duplication, or disclosure by the United States Government is subject to the restrictions set forth in DFARS 252.227-7013 (c)(1)(ii) and FAR 52.227-19. Use of the materials by the Government constitutes acknowledgement of Aspect's proprietary rights in them. Aspect Software, Inc. is located at 300 Apollo Drive, Chelmsford, MA 01824, USA.

LIMITED RIGHTS NOTICE (DEC 2007)
(a) These data are submitted with limited rights under Aspect Software, Inc. ("Aspect") contracts with various government entities (the "Government"). These data may be reproduced and used by the Government with the express limitation that they will not, without written permission of the Aspect, be used for purposes of manufacture nor disclosed outside the Government; except that the Government may disclose these data outside the Government for the following purposes, if any, provided that the Government makes such disclosure subject to prohibition against further use and disclosure: None.

(b) This notice must be marked on any reproduction of these data, in whole or in part.

EXPORT REQUIREMENTS
This item is subject to U.S. export control laws and regulations. This item may not be exported, re-exported, re-transferred, disclosed or otherwise diverted contrary to U.S. export control laws and regulations.

NO WARRANTY
THE CONTENTS OF THIS PUBLICATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF QUALITY, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

ASPECT SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED AS A RESULT OF USING THE CONTENTS OF THIS PUBLICATION. IN NO EVENT SHALL ASPECT BE LIABLE FOR ANY INDIRECT, PUNITIVE, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGE (INCLUDING LOSS OF BUSINESS, REVENUE, PROFITS, USE, DATA OR OTHER ECONOMIC ADVANTAGE) HOWEVER IT ARISES, WHETHER FOR BREACH OR IN TORT, EVEN IF ASPECT HAS BEEN PREVIOUSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. BECAUSE SOME JURISDICTIONS PROHIBIT THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

PROGRAMMING AND USE OF PRODUCTS
THE PRODUCTS DESCRIBED IN THIS DOCUMENTATION CAN BE USED AND PROGRAMMED IN A WIDE VARIETY OF WAYS BASED UPON THE REQUIREMENTS OF YOUR PARTICULAR TECHNOLOGY ENVIRONMENT AND BUSINESS NEEDS. NOTWITHSTANDING THE USE OF EXAMPLES IN THE DOCUMENTATION OR THE PROVISION OF PROFESSIONAL SERVICES BY ASPECT, ASPECT RESELLERS OR ANY THIRD PARTY ENGAGED BY ASPECT, IT IS IN ALL CASES THE USER'S RESPONSIBILITY TO ENSURE THAT THE PRODUCTS ARE PROGRAMMED AND USED IN ACCORDANCE WITH ALL APPLICABLE LAWS AND REGULATIONS AND IN A MANNER THAT DOES NOT VIOLATE THE INTELLECTUAL PROPERTY AND OTHER RIGHTS OF ANY THIRD-PARTY.

Part Number: 3205-1787A
Date: June 28, 2013

# Contents


**About this Guide ..... vii**
Audience ..... vii
Organization of this Guide ..... vii

**Chapter 1 Introduction ..... 1-1**
Overview ..... 1-1
APIs in the External CTI ..... 1-2
Security ..... 1-2
High Availability (HA) ..... 1-2
Failover ..... 1-3
CTIPS Client Connections ..... 1-3
CTIPS Functionality ..... 1-3
Installation and Configuration ..... 1-4

**Chapter 2 Events, Requests, and Responses ..... 2-1**
Inbound Telephone Call Events ..... 2-1
Requests and Responses ..... 2-2
Request Messages ..... 2-2
Session and System ..... 2-2
Call Control Requests ..... 2-4
Response Messages ..... 2-4
Event Messages ..... 2-5

**Chapter 3 Requests ..... 3-1**
Session and System Requests ..... 3-1
Tenant Identifier and Session Identifier ..... 3-1
OpenConnection ..... 3-2
CloseConnection ..... 3-2
Online 3-3
Offline 3-3
AddMessageClass ..... 3-4
DeleteMessageClass ..... 3-4
AddFilter ..... 3-5
DeleteFilter ..... 3-5
Snapshot ..... 3-6
Agent Management Requests ..... 3-7
LoginAgent ..... 3-7
LogoutAgent ..... 3-8

PanicLogout ..... 3-9
SetAgentState ..... 3-10
GetAgentState ..... 3-11
SetDisposition ..... 3-11
Call Control Requests ..... 3-12
AcceptCall ..... 3-12
ClearCall ..... 3-13
ConferenceCall ..... 3-13
ConsultationCall ..... 3-14
GetCallData ..... 3-15
SetCallData ..... 3-15
HoldCall ..... 3-16
RetrieveCall ..... 3-17
MakeCall ..... 3-17
SendDTMFDigits ..... 3-18
TransferCall ..... 3-19
Call Routing Request ..... 3-20
RouteSelect ..... 3-20

**Chapter 4 Events ..... 4-1**
Event Description ..... 4-1
Void PublishCTIEvent (CTIEvent event) ..... 4-1
Int HeartbeatMessage() ..... 4-1
Int StartSession(UserCredentials userinfo, string sessionId) ..... 4-2
Event Messages ..... 4-2
Event Header ..... 4-2
Event Categories - CTIEventCategory ..... 4-3
EventType – Event Message Type ..... 4-3
RequestId ..... 4-3
Agent Events Descriptions ..... 4-3
AgentLoginStateChange Event ..... 4-3
AgentLoginRequestFailed Event ..... 4-5
AgentStateChange Event ..... 4-5
AudiopathStateChange Event ..... 4-6
EnterPasscode Event ..... 4-7
ServiceStateChange Event ..... 4-8
AgentRequestFailed Event ..... 4-9
Call Events Descriptions ..... 4-9
CallCleared Event ..... 4-9
CallCompleted Event ..... 4-10
CallConferenced Event ..... 4-11
CallConnected Event ..... 4-11
CallConsultationEvent ..... 4-13
CallDataUpdate Event ..... 4-13
CallDelivered Event ..... 4-14
CallEnd Event ..... 4-16

CallEstablished Event ..... 4-16
CallHeldEvent ..... 4-17
CallOffered Event ..... 4-18
CallOriginated Event ..... 4-19
CallRequestFailed Event ..... 4-20
CallRetrieved Event ..... 4-20
CallStarted Event ..... 4-21
CallTransferred Event ..... 4-22
DigitsDialed Event ..... 4-23
Call Route Event Descriptions ..... 4-23
RouteRequest Event ..... 4-23
RouteUsed Event ..... 4-24
RouteRejected Event ..... 4-26
RouteRequestFailed Event ..... 4-26
System Event Descriptions ..... 4-27
SnapShotUpdate Event ..... 4-27
LinkStatus Event ..... 4-28

**Chapter 5 Using the CTI Portal Server ..... 5-1**
Client Application Startup and Shutdown Sequence ..... 5-1
Opening a Session With CTIPS ..... 5-1
Registering for Events ..... 5-2
Adding Filters ..... 5-2
Placing a Session Online ..... 5-2
Requesting a Snapshot ..... 5-3
Placing a Session Offline ..... 5-3
Closing a Session With CTIPS ..... 5-4
How to Register for Events ..... 5-4
Examples of Registering an Event ..... 5-5
Registering for Agent Events ..... 5-5
Registering for Specific Agent Events ..... 5-5
Registering for Call Events ..... 5-5
Registering for Routing Events ..... 5-5
Registering for Specific Service Ids ..... 5-6

**Appendix A Data Types ..... A-1**
User Credentials ..... A-1
CTISnapShotAgentData ..... A-1
CTIAgentCapabilities ..... A-2
CTIAgentState ..... A-2
CTICallProgressSignalType ..... A-3
CTIServiceInfo ..... A-3
CTIServiceState ..... A-3
CTIMediaType ..... A-4
CTISnapShotCallData ..... A-4
CTIDestinationType ..... A-5
CTICallState ..... A-5

CTICallType ..... A-6
CTISnapShotServiceData ..... A-6
CTIMessageClass ..... A-7
CTIEventCategory ..... A-7
CTIEventType ..... A-7
CTICallInfo ..... A-8
CTICallDataItem ..... A-9
CTICallDataUpdateType ..... A-9
CTIEntityType ..... A-9
CTIFieldType ..... A-10
CTIAgentLoginState ..... A-10
CTIClearingPartyType ..... A-10
CTIRouteType ..... A-10
CTIRouteStatusType ..... A-11
CTIRejectReason ..... A-11
CTITransferType ..... A-12
CTIAudioPathState ..... A-12
CTIFailureType ..... A-12
CTIComponentType ..... A-13
CTIComponentState ..... A-14

**Appendix B Configuration ..... B-1**
Configuring CTIPS Using the Server Configurator ..... B-1
Adding a new CTI Portal Server ..... B-1
Modifying an Existing CTI Portal Server ..... B-2
Deleting an Existing CTI Portal Server ..... B-2

**Appendix C Summary of Services and Events ..... C-1**
Summary of Session, Registration, and System Services ..... C-1
Summary of Agent, Call Control, and Routing Services ..... C-2
Summary of Agent, Call Control, Routing, and System Events ..... C-3

**Appendix D Sample Code ..... D-1**
Sample client code ..... D-1

**Appendix E Event Flow Diagrams ..... E-1**
Call Arrives to Inbound Service ..... E-2
Route Request - Route Select to Invalid Destination ..... E-3
Route Request - Route Select to Valid Service ..... E-4
Blind Transfer to Agent ..... E-5
Blind Transfer to Valid Service ..... E-6
Consult Transfer to Service ..... E-7
Consult Transfer Agent ..... E-8
Blind Transfer to Unmanned Service ..... E-9
3-Way Conference Service ..... E-10
Agent to Agent ..... E-11

3-Way Conference Agent..... E-12
Conference Initiator Hang Up ..... E-13

**Index. ..... Index-1**

# About this Guide

This document provides detailed descriptions of events, requests, and responses that are generated in an Aspect Unified IP CTI Portal Server (CTIPS) environment. CTIPS allows application developers to control and manage the inbound, voice call handling, and agent outbound features of the Aspect Unified IP system.

For a list of published documents supporting other Aspect product applications, components, or utilities, refer to the *Documentation Set* topic in the appropriate *Product Release Note* document.

## Audience

This guide is for experienced application developers. Readers are required to have a basic understanding of and familiarity with the Microsoft .NET Framework and Windows Communication Foundation platform (WCF) as well as an understanding of sophisticated software applications such as Aspect Unified IP and Aspect Unified IP Unified Command and Control® - Administration.

For information about Training, Technical Support, commenting on the documentation, and a list of additional documentation see the appropriate product Release Notes document on the Aspect Software web site at http://www.aspect.com.

## Organization of this Guide

This guide consists of the following chapters:

- Chapter 1, Introduction, includes an introduction to the fundamentals of working with the Aspect Unified IP CTI Portal Server (CTIPS) API.
- Chapter 2, Events, Requests, and Responses, includes an introduction and description of the various events, requests, and responses that are generated in the CTIPS environment.
- Chapter 3, Requests, includes a brief description of CTI requests organized by request category. The parameters associated with each request are listed and described.
- Chapter 4, Events, includes event names, type, and descriptions.
- Chapter 5, Using the CTI Portal Server, describes various examples for implementing applications that use the CTIPS API.
- Appendix A, Data Types, includes detailed descriptions of each service and event statistic.
- Appendix B, Configuration, includes instructions on how to configure, modify, and delete the CTIPS using Server Configurator.

- Appendix C, Summary of Services and Events, includes of services and events, registration and monitoring, and session events.
- Appendix D, Sample Code, includes samples of client code.
- Appendix E, Event Flow Diagrams, contains event flow diagrams of various CTIPS scenarios.
- Index, provides a detailed index of the contents of this guide.

1

# Chapter 1
# Introduction

This chapter provides an overview of the Aspect Unified IP CTI Portal Server (CTIPS) API. The CTIPS supports inbound voice call handling and agent outbound calls only, and does not address outbound dial campaigns, chat, email, or workflow services. The CTIPS API is implemented using the Microsoft® .NET™ Framework and Windows® Communication Foundation platform (WCF), and exposes agent management, call control, and call routing functionality through a standard, service-oriented architecture (soa) based technology.

## Overview

The CTI Portal Server (External CTI) allows Unified IP the ability to publish real-time statistics of agent state, calls, and services to an external application. It also allows external operational control of calls and agent states. This gives an external application the ability to change specific agent states, query the state of an agent, and perform call control.

An external application can subscribe to receive real-time statistics that provide system resource status. The Unified IP system provides call progress statistics for all inbound and outbound calls. The external application receives information indicating that a call has been connected, held, retrieved, transferred, conferenced, and disconnected.

This external application builds a real-time call model. In addition, using the CTI Portal Server, an external application that has the ability to drive the routing rules, if necessary. The Unified IP system has the capability of proactively issuing route requests to an external application. This functionality provides the ability for an external application to route a call to a specific agent, service or external number. CTIPS exposes agent management, call control, call routing, and monitoring functionality to external client applications (client). The CTIPS API supports a request/synchronous response model. Once the client successfully issues a request and receives its response, the client receives one or more events from the Unified IP system through the CTIPS regarding the progress and completion of the client request. Certain events result in state changes within the client which cause the client to issue another request.

> Example: The client issues a **LoginAgent** request and receives a successful response from CTIPS indicating that agent login request is valid and has been initiated. Sometime later, the client receives an **AgentLoginStateChange** event indicating that the agent has been successfully logged in. If the agent's initial state was set to idle in the **LoginAgent** request, the client receives an **AgentStateChange** event indicating that the agent is idle.

The CTIPS provides the functionality to develop CTI-enabled applications. Some examples of applications that can be created:

- A CTI application can be developed to control the agent state. The application is set to the agent's current work state according to the type of work being performed.
- A CTI application can be developed to establish the data to display on a screen pop.
- A CTI application can control the media in a Unified IP system.
- A CTI application can perform third-party call control.

## APIs in the External CTI

The CTIPS presents two APIs. There is an Agent Management and Call Control interface and a notification interface (Publish Event).

- The Agent Management and Call Control interface enables the client to take control of a call for call processing and to control an agent's state.
- The Publish Event interface enables the client to receive notification of changes that occur within Aspect Unified IP.

# Security

You can configure the CTIPS component to communicate securely with its client.

You can configure it to use transport level security through certificates and by using user name and password authentication for authenticating the client for its interaction with Aspect Unified IP. When the CTIPS is configured for security, it is recommended that the CTIPS client application enable its interface to use transport security using certificates. If transport security is used by the client, ensure the CTIPS service machine is configured to accept the certificate presented by the CTIPS client application.

Refer to the *Aspect Unified IP Installation Guide t*o enable security on Aspect Unified IP system machines.

# High Availability (HA)

The CTIPS can operate in an Active/Active high availability (HA) scheme. In the event a component fails in the system, HA continues to provide the availability of resources in the system.

- There can be more than one instance of the CTI Portal Server on a single deployment of Unified IP.
- The CTI Portal Server is stateless. If the CTI Portal Server were to fail, after recovery, the client must attempt to reconnect and resubmit any subscription and filter information with the new server. After a successful re-connection, it is the client's responsibility to request a snapshot of the system to get a complete system-wide view and status of all agents, calls, and services.

- The CTI Portal Server is highly available for its clients. An instance of the CTI Portal Server is associated with a particular tenant. There may be multiple instances of the CTI Portal Server for each tenant.

# Failover

The client is configured with the URLs of the CTI Portal Server and it is the responsibility of the client to failover when a primary CTI Portal Server becomes unavailable.

# CTIPS Client Connections

- The maximum number of client connections to CTIPS is ten (10).
- If the allowable number of client connections is ten and an attempt is made to exceed that number, the CTI Portal server responds that the maximum number of connections has been reached.
- For first party call control, client applications must use the Agent SOAP SDK and not the API provided by the CTIPS. These two APIs can coexist and be used together on a Unified IP system.

# CTIPS Functionality

The CTI Portal Server allows Unified IP the ability to publish real-time statistics of agent state, calls, and services to an external application. It also allows external operational control of calls and agent state. This gives an external application the ability to change specific agent states, query the state of an agent, and perform call control.

- An external application can subscribe to receive real-time statistics that provide system resource status.
- The Unified IP system provides call progress statistics for all inbound and outbound calls.
- The external application will receive information indicating that a call has been connected, held, retrieved, transferred, conferenced, and disconnected. This external application will be able to build a real-time call model.
- In addition, using the CTI Portal Server, an external application has the ability to drive the routing rules. The Unified IP system has the capability of proactively issuing route requests to an external application. This functionality provides the ability for an external application to route a call to a specific agent, service or external number.

# Installation and Configuration

The CTI Portal Server is configured in the Aspect Server Configurator application.

- Logging levels, number of missed heartbeats, heartbeat interval, and web service port number are configured using this application.
- When exceptions occur, the CTI Portal Server returns concise error messages to the client and logs specific details on the server. The log information contains information according to the log levels that are set in the Server Configurator application (that is ERROR, INIT, TRACE, MIN, TRACE MAX).
- See the *Aspect Unified IP Server Configurator User Guide* for detailed information.
- The Aspect Unified IP Unified Command and Control - Administrator application is used to configure an Inbound Service for “external routing” enabling it for third party routing. See the *Aspect Unified IP Unified Command and Control - Administration System Administrator Guide* for detailed information.
- A system administrator will need to configure a user in Active Directory. Use the Aspect Unified IP Server Configurator application, to add this user to the CTIPS configuration.

# 2

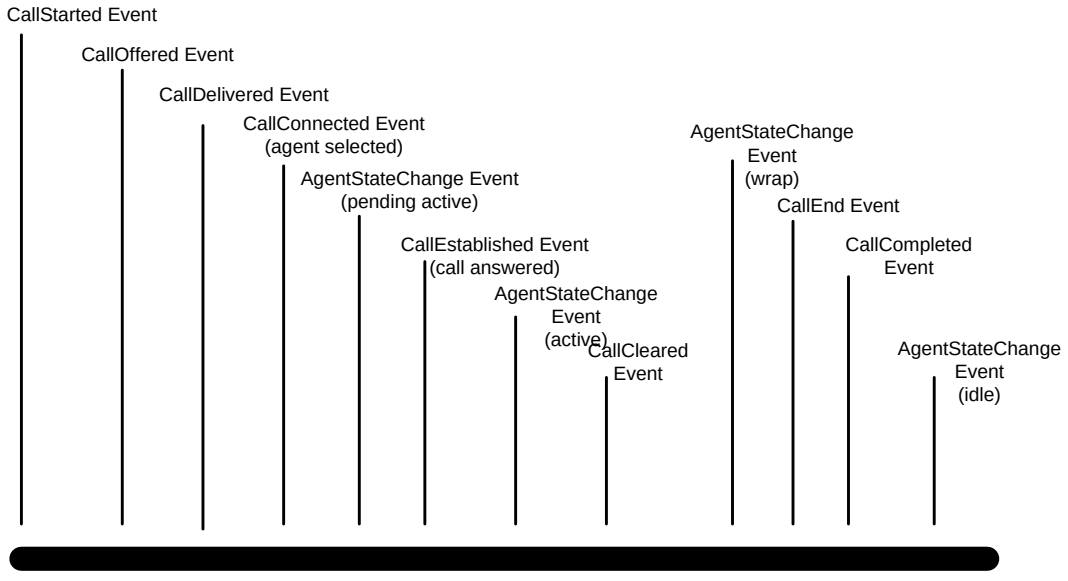
# Chapter 2

# Events, Requests, and Responses

When a call enters the Aspect Unified IP system, it undergoes various stages. **Events** called Asynchronous (or unsolicited) events get generated by the system. In a CTI environment, these events are sent to client applications that have opened a **session** with the CTI Portal Server and have **registered** for such events. Upon receiving these asynchronous events, client applications can take action by submitting a request. **Requests** are the mechanism that the application uses in response to an event or to request that a desired behavior happen (for example, TransferCall). An example of an asynchronous event is the CallDelivered Event.

## Inbound Telephone Call Events

The following figure represents events generated for a typical inbound telephone call. The figure depicts the events that are generated and the agent and call state transitions of a typical basic inbound call:




- **CallStarted Event** indicates that a new call has arrived or a new call has been initiated by the Unified IP system.
- **CallOffered Event** indicates that a call has arrived at the Unified IP system.

- **CallDelivered Event** is generated when the call is queued to an inbound service. The caller now hears ringing or attention retainers.
- **CallConnected Event (agent selected**) is generated when the call is presented to a device in either the ringing or entering distribution modes of the alerting state.
- **AgentStateChange Event (pending active)** is generated when the agent's state changes to "pending active."
- **CallEstablishedEvent (call answered)** is generated when the agent accepts the call.
- **AgentStateChange Event (active)** is generated after the agent accepts the call and the agent's state changes to "active."
- **CallCleared Event** is generated when a call has been disconnected by either the near end or the far end.
- **AgentStateChange Event (wrap)** is generated when the agent's state changes to "wrap".
- **CallEnd Event** is generated when all connections to a call have been cleared.
- **CallCompleted Event** is generated when all call segments are complete. This includes the agent wrap time if applicable for a call segment.
- **AgentStateChange Event (idle)** is generated when the agent‘s state changes to "idle" after the agent dispositions the call.

# Requests and Responses

A Request/Response mechanism is used in CTIPS. The CTI client application makes programmatic **requests** using the CTIPS API and receives a request specific **response** for each request. These requests can be used to perform **agent functionality, call control**, **call routing** and miscellaneous **session and system** level functionality.

## Request Messages

The client requests are categorized as follows:

## Session and System

- OpenConnection
- CloseConnection
- OnLine
- Offline
- AddMessageClass
- DeleteMessageClass
- AddFilter
- DeleteFilter
- Snapshot

### *Agent Requests*

- LoginAgent
- LogoutAgent
- PanicLogout
- SetAgentState
- GetAgentState
- SetDisposition

### *Route Requests*

- RouteSelect

## Call Control Requests

- AcceptCall
- ClearCall
- ConferenceCall
- ConsultationCall
- GetCallData
- SetCallData
- HoldCall
- MakeCall
- RetrieveCall
- TransferCall
- SendDTMFDigits

## Response Messages

After receiving a request message from the client, the CTIPS sends back a response message. Since the client and the CTIPS are separated by network boundaries, the response is structured as a synchronous response to the request.

Responses are characterized as follows:

- There is a one-to-one correlation between a request and a response.
- If the CTIPS can fully process the request, then CTIPS returns a success. If the CTIPS rejects the request outright (that is due to an invalid parameter), then the CTIPS returns a fault exception containing a descriptive error message. When the request is sent to the Unified IP system and finally completes at some later time, one or more notification **events** is sent to the client.

# Event Messages

When the CTIPS receives a request and returns an acknowledgement response back to the client, it continues to process the request and sends the request to the Unified IP system. During the course of this processing, one or more events may be sent to the client indicating both progress of the request and eventual completion of the request. Following request completion, events may continue to be sent to the client. Events may also be sent from CTIPS to the client application in response to call or agent state change that wasn't in response to a request from a client, these events are known as unsolicited or asynchronous events.

Example:

1. The client receives an unsolicited RouteRequest event from CTIPS when an incoming call arrives at a service configured for external routing.
2. The client sends a Route Select Request to route the incoming call to an agent
3. The CTIPS accepts the Route Select Request and sends a Response/Acknowledgement message back to the client.
4. The CTIPS sends the Route Select Request on to the Unified IP system. When the call is routed to the agent, the Unified IP system sends a RouteUsed event to the CTIPS which then forwards the event to the client. This event logically completes the Route Select Request.
5. Once the incoming call has been successfully routed to the agent, a Call Connected event is sent to the client.

Events belong to one of the following categories

- EventCategoryAgent
- EventCategoryCall
- EventCategoryRoute
- EventCategorySystem

# Chapter 3
# Requests

This chapter offers a brief description of CTI requests organized by request category. The parameters associated with each request are listed and described.

See Appendix A, Data Types for details on data types.

## Session and System Requests

This section provides a description of the Session and System type requests provided by CTIPS. Session requests are issued by the client application to establish, define and maintain a connection (session) with CTIPS.

### Tenant Identifier and Session Identifier

Aspect Unified IP has a multi-hosting concept where one instance of UIP provides functionality to multiple customers, each of these customers is referred to as a tenant.

For example, a single installation of Unified IP provides call center functionality for customers A, B and C. When the Unified IP system is configured, each customer (tenants) are configured with a unique tenant identifier. The tenant identifier (TenantId) is a unique, integer value, which must be provided by the client in all client requests to CTIPS.

**Note:** CTIPS only provides single tenant functionality. Some CTIPS requests (for example, **OpenConnection**) allow entry of a list of TenantIds. This provides future capabilities without changing the existing API. The client must provide a single TenantId in the OpenConnection request.

A session identifier (SessionId) is a unique string value provided by CTIPS to each CTIPS client to uniquely identify that client connection. CTIPS supports up to 10 simultaneous client connections. When a client initially opens a connection with CTIPS, CTIPS generates a SessionId and returns it to the client in the OpenConnection response message. The client must include this SessionId in all subsequent requests to CTIPS.

# OpenConnection

The OpenConnection request opens a client connection (session) to the CTI Portal Server. This is the first request that a client issues to the CTI Portal Server. This request returns a unique session identifier (**SessionId**) to the client. This SessionId is used in all other client requests.

**Syntax**

**ResponseCode** OpenConnection(**OpenConnectionArgs** args**)**;

**Parameters**

The fields below are properties of the OpenConnectionArgs type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | List of tenant Identifiers. | Lists<int> |
| UserInfo | Custom type which contains *Username (string)* and *Password (string)* properties. | **UserCredentials** |
| ClientCallbackURL | The PublishCTIEvents URL provided by the client application so that CTIPS can publish events to the client. | String |

**Return Value**

The **ResponseCode** object contains a SessionId field of type string as well as a status field of type int. If the OpenConnection request is successful then the session Id contains a unique identifier and the status field is set to 0 (success). If the OpenConnection request fails then the status field contains a **status code** that depicts the type of error.

# CloseConnection

The client issues this request to end the session between client and server (CTIPS).

**Syntax**

**int** CloseConnection(**CloseConnectionArgs** args)

**Parameters**

The field below is a property of the CloseConnectionArgs type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |

# Online

The OnLine request sets the session to the ONLINE state, making the client eligible to receive messages from the CTI Portal Server. Prior to sending the OnLine request the client sets up the appropriate filtering using the **AddMessageClass** and **AddFilter** requests to avoid receiving any unnecessary messages.

**Syntax**

**int** OnLine(**OnLineArgs args**)

**Parameters**

The fields below are properties of the OnLineArgs type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier. | Int |
| SessionId | Set this to the return value from the OpenConnection request | String |

# Offline

The OffLine request sets the session to the OFFLINE state. The client is not able to receive messages from the CTI Portal Server until placed back ONLINE using the OnLine request.

**Syntax**

**int** OffLine(**OffLineArgs args**)

**Parameters**

The fields below are properties of the OffLineArgs type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| SessionId | Set this to the return value from the OpenConnection request | String |

# AddMessageClass

The AddMessageClass request allows the client to set a filter based on the class of message.

**Syntax**

**int** AddMessageClass(**AddMessageClassArgs** args)

**Parameters**

The fields below are properties of the **AddMessageClassArgs** type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| SessionId | Set this to the return value from the OpenConnection request | String |
| MessageClass | Set to one of the following<br>• EventCategoryAgent<br>• EventCategoryCall<br>• EventCategoryRoute<br>• EventCategorySystem | **CTIMessageClass** |

# DeleteMessageClass

The DeleteMessageClass allows the client to delete a filter based on the class of message.

**Syntax**

**int** DeleteMessageClass(**DeleteMessageClassArgs** args)

**Parameters**

The fields below are properties of the DeleteMessageClassArgs type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| SessionId | Set this to the return value from the OpenConnection request | String |
| MessageClass | Set to one of the following<br>• EventCategoryAgent<br>• EventCategoryCall<br>• EventCategoryRoute<br>• EventCategorySystem | **CTIMessageClass** |

# AddFilter

The AddFilter request allows the client to add a filter based on a specific agent, service, or switch id. Adding a filter allows the client to minimize the amount of unnecessary events that the client receives

**Syntax**

**int** AddFilter(**AddFilterArgs** args)

**Parameters**

The fields below are properties of the **AddFilterArgs** type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| SessionId | Set this to the return value from the OpenConnection request | String |
| FieldType | The field to filter on.<br>Set to one of the following<br>• FT_AGENT_ID<br>• FT_SERVICE_ID<br>• FT_SWITCH_ID | **CTIFieldType** |
| FieldValue | The value of the field to filter on. | String |

# DeleteFilter

The DeleteFilter request allows the client to delete a filter based on a specific agent, service, or switch id

**Syntax**

**int** DeleteFilter(DeleteFilterArgs args)

**Parameters**

The fields below are properties of the **DeleteFilterArgs** type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| SessionId | Set this to the return value from the | String |

| Field Names | Description | Type |
|---|---|---|
| | OpenConnection request | |
| FieldType | The field to filter on.<br>Set to one of the following<br>• FT_AGENT_ID<br>• FT_SERVICE_ID<br>• FT_SWITCH_ID | **CTIFieldType** |
| FieldValue | The value of the field to filter on. | String |

# Snapshot

The Snapshot request allows the client to retrieve a current snapshot of the system (that is, agent, service and call information). The client must use this request after creating a session with the CTIPS and is OnLine to understand the present state of any calls, services, or agents it wants to monitor or control. The Snapshot request returns a Snapshot response and later generates one or more **SnapshotUpdate** events to deliver the requested information.

## *Syntax*

**int** Snapshot(**SnapshotArgs args**)

## *Parameters*

The fields below are properties of the SnapshotArgs type.

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| SessionId | Set this to the return value from the OpenConnection request | String |
| EntityType | An array that contains the entities of interest;<br>• AgentInfo<br>• CallInfo<br>• ServiceInfo | **CTIEntityType[]** |

# Agent Management Requests

This section provides a description of the Agent Management type requests provided by CTIPS.

### *Request Identifier, Agent Identifier, and Call Identifiers*

- The request identified (RequestId) is a unique integer value, provided by the client in all agent management, call control, and routing requests. It is the client's responsibility to generate a unique RequestId. The RequestId is present in all agent management requests and all call control requests. All events generated by the Unified IP system contains this request identifier and is sent to the client by the CTIPS. The client uses this request identifier to correlate any requests made with any events received. Unsolicited events that are not generated due to a client request returns an empty (null) RequestId.
- The Agent Identifier (AgentId) is provided to the client in events following a login request. The client must include the agent Identifier in all agent requests following a successful agent login. This Id is used by the Unified IP system to identify the agent.
- Call Identifiers (CallId and ConsultationCallId) are internal identifiers used in the Unified IP system to identify calls. The CallId identifies the agent/customer call, and the optional ConsultationCallId identifies a consultation call to a third party.
  - CallId is present in all EventCategoryCall events sent to the client and must be sent in call management requests sent from the client to control agent/customer calls.
  - ConsultationCallId is present in events relating to placing an outbound call, consultation call, blind transfer, consultation transfer, and creating a conference. It must be present in requests which manage agent/third party consultation calls.

## LoginAgent

The LoginAgent request must be used by the client application to Login a specific Unified IP agent.

### *Syntax*

**int** LoginAgent(**LoginAgentArgs args)**

### *Parameters*

The fields below are properties of the **LoginAgentArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier; provided by the client. | Int |

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| AgentLoginName | Agent's login name | String |
| StationId | Agents device, phone number or address | String |
| Initial State | The agent's desired initial state after the agent is successfully logged in.<br>Set to one of the following:<br>• Invalid<br>• NotReady<br>• NotReadyPark<br>• Idle<br>• PendingActive<br>• Active<br>• Wrap<br>• ActiveWrap<br>• WrapWarning<br>• Reserved<br>• MultiLine | **CTIAgentState** |
| MultiLine | Set to one of the following:<br>0 – client application does not support multiline<br>1 – client application supports multline; agent can be on two voice calls concurrently. | **Int** |

# LogoutAgent

A request to the Unified IP system to logout the specified agent.

## *Syntax*

**int** LogoutAgent(**LogoutAgentArgs args**)

## *Parameters*

The fields below are properties of the **LogoutAgentArgs** type.

| Field Names | Description | Type |
|---|---|---|

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier. | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| ReasonCode | Logout reason code | Int |

# PanicLogout

A request for Unified IP to perform a Panic Logout of the specified agent or agents. This request immediately logs out agents who were in idle state. Agents that were on a call are placed in “pending logout state, no dispositioning”. The client must supply an array of one or more agent login names (AgentLoginNameList) of the agents to be logged out.

## *Syntax*

**int** PanicLogout(**PanicLogoutArgs args**)

## *Parameters*

The fields below are properties of the **PanicLogoutArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier. | Int |
| TenantId | Tenant Identifier | Int |
| AgentLoginNameList | Array of agents’ login name to be logged out | String[] |
| LogoutAllAgents | Set to "true" to logout all agents.<br>• If LogoutAllAgents is false, the AgentLoginNameList must be set.<br>• If LogoutAllAgents is true, it is not necessary to provide the AgentLoginNameList. | Bool |

# SetAgentState

The SetAgentState request to the Unified IP system can be used to change the current state of an agent.

## *Syntax*

**int** SetAgentState(**SetAgentStateArgs args**)

## *Parameters*

The fields below are properties of the **SetAgentStateArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier. | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| NewState | Agent's new state. Set to one of the following<br>• NotReady<br>• NotReadyPark<br>• Idle<br>• PendingActive<br>• Active<br>• Wrap<br>• WrapWarning<br>• Reserved | **CTIAgentState** |

# GetAgentState

A request to the Unified IP system to get the state of the agent.

### *Syntax*

**int** GetAgentState(**GetAgentStateArgs** args)

### *Parameters*

The fields below are properties of the **GetAgentStateArgs** type:

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |

# SetDisposition

A request to the Unified IP system to set the disposition of an agent

### *Syntax*

**int** SetDisposition(**SetDispositionArgs args**)

### *Parameters*

The fields below are properties of the **SetDispositionArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request. | String |
| RequestId | Request Identifier. | Int |
| TenantId | Tenant Identifier. | Int |
| AgentId | Agent Identifier. | Int |
| CallId | Call Identifier. | Int |

| Field Names | Description | Type |
|---|---|---|
| DispositionId | The disposition identifier as configured in Unified Command and Control - Administrator application. | Int |
| SummaryDispositionSuccessful | If “true”, the call is considered positively dispositioned. | Bool |
| WrapNotRequired | Set to “true” to bypass wrap state. | Bool |

# Call Control Requests

The Call Control interface allows the client to take control of a call for call processing.

## AcceptCall

Allows an agent to either accept or reject a call that has been queued to a service or an agent.

### *Syntax*

**int** AcceptCall(**AcceptCallArgs** args)

### *Parameters*

The fields below are properties of the **AcceptCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier provided in the preceding Call Connected Event. | Int |
| AcceptCall | Accept call if true, If not true, reject the call. | Bool |
| RejectReason | If the call is rejected, the reject reason identifier configured in Unified Command and Control - Administrator application. | Int |

# ClearCall

Allows the client the ability to clear all devices in a 2-way or 3-way call. The client can clear either leg of a 3-way call by supplying the CallId or the ConsultationCallId.

### *Syntax*

**int** ClearCall(**ClearCallArgs** args)

### *Parameters*

The fields below are properties of the **ClearCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier for agent/customer or the Call identifier for a consultation call. | Int |

# ConferenceCall

Provides a conference of an existing held call and another active call (consultation call).

### *Syntax*

int ConferenceCall(**ConferenceCallArgs** args)

### *Parameters*

The fields below are properties of the **ConferenceCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier for agent, customer, or non consultation outbound call. | Int |
| ConsultationCallID | Call identifier of the consultation call between the agent and a third party. The customer is associated with the call specified in CallId, and the agent and consultation party are in the call specified in ConsultationCallId. All three parties will be joined into a conference. | Int |

# ConsultationCall

Places an existing active call at a device on hold and initiates a new call. A consultative call is placed to another agent or an external device (using a number that can be dialed) as defined by the DestinationType and DestinationAddress fields.

## *Syntax*

**int** ConsultationCall(**ConsultationCallArgs** args)

## *Parameters*

The fields below are properties of the **ConsultationCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier for agent, customer or non consultation outbound call. | Int |
| DestinationAddress | Agent Login Name, Service Id, or external number. | String |
| ServiceId | Service Identifier | Int |

| Field Names | Description | Type |
|---|---|---|
| DestinationType | Can only be set to one of the following values:<br>• Agent<br>• External<br>• Service | **CTIDestinationTyp**e |
| UserData | 20 user defined fields | String |

# GetCallData

Allows the client to retrieve user defined call associated data.

### *Syntax*

int GetCallData(**GetCallDataArgs** args)

### *Parameters*

The fields below are properties of the **GetCallDataArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |

# SetCallData

Allows the client to set user defined call associated data.

### *Syntax*

**int** SetCallData(**SetCallDataArgs** args)

### *Parameters*

The fields below are properties of the **SetCallDataArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier. | Int |
| UserData | 20 user defined variable fields | String[] |

## HoldCall

Allows the client the ability to place a specific call on hold.

### *Syntax*

**int** HoldCall(**HoldCallArgs** args)

### *Parameters*

The fields below are properties of the **HoldCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call Identifier for agent or customer call | Int |

# RetrieveCall

Allows the client the ability to connect to a call that had previously been placed on hold.

### *Syntax*

**int** RetrieveCall(**RetrieveCallArgs** args)

### *Parameters*

The fields below are properties of the **RetrieveCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call Identifier for agent or customer call | Int |

# MakeCall

Allows the client to establish a call between two devices.

### *Syntax*

**int** MakeCall(**MakeCallArgs** args)

### *Parameters*

The fields below are properties of the **MakeCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier for agent, customer, or non consultation outbound call. | Int |
| DestinationType | • Agent<br>• Service<br>• External<br>• NoDestination | **CTIDestinationType** |
| ServiceId | Optional service identifier associated with the outbound call if there is no existing agent/customer call (CallId = -1) | Int |
| WrapState | Normally false, set to true if the agent wants to initiate an outbound call while still in the wrap state of a previous call (wrap dial) | Bool |
| DestinationAddress | The AgentLoginName, ServiceId, external number, or address of the third party to dial. | String |

# SendDTMFDigits

Dials a digit for a call that has already been initiated. To send more digits, call SendDTMFDigits multiple times.

## *Syntax*

int SendDTMFDigits(**SendDTMFDigitsArgs** args)

## *Parameters*

The fields below are properties of the **SendDTMFDigitsArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |

| Field Names | Description | Type |
|---|---|---|
| AgentId | Agent Identifier | Int |
| CallId | Call identifier for agent/customer or non consultation outbound call. | Int |
| DTMFDigits | Digit to dial | String |

# TransferCall

Allows the client to transfer a two-party call that is in a connected state.

### *Syntax*

**int** TransferCall(**TransferCallArgs** args)

### *Parameters*

The fields below are properties of the **TransferCallArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier for agent, customer, or non consultation outbound call. | Int |
| DestinationType | • Agent<br>• Service<br>• External | **CTIDestinationType** |
| DestinationAddress | The AgentLoginName, ServiceId, external number or address of the third party to dial. Must be provided when CTITransferType is set to Blind. | String |
| ConsultationCallId | The call identifier of the consultation call between the agent and consultation party. Must be provided when CTITransferType is set to Consultation. | Int |

| Field Names | Description | Type |
|---|---|---|
| TransferType | • Blind<br>• Consultation | **CTITransferType** |
| UserData | 20 user defined fields | String[] |

# Call Routing Request

## RouteSelect

The RouteSelect request is used by the client to provide Unified IP with a route destination in response to a RouteRequest Event. The client application issues this request to route a call to a specific agent, service, external entity or to allow Unified IP to perform default routing. The request must provide a RouteType and a RouteDestination.

- To route the call to a specific agent, the RouteType must be set to RouteToAgent and the RouteDestination must be set to the index of the agent.
- To route the call to a specific service, the RouteType must be set to RouteToService and the RouteDestination must be set to the ServiceId of the service.
- To route the call to an external entity, the RouteType must be set to RouteToExternal and the RouteDestination must be set to a number that can be dialed or the address of the external party.
- To allow Unified IP to perform it's configured default routing, the RouteType must be set to RouteToDefault.

The client provides up to 20 user defined call associated data variables through the UserData[].

### *Syntax*

**int** RouteSelect(**RouteSelectArgs** args)

### *Parameters*

The fields below are properties of the **RouteSelectArgs** type.

| Field Names | Description | Type |
|---|---|---|
| SessionId | Set this to the return value from the OpenConnection request | String |
| RequestId | Request Identifier | Int |

| Field Names | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| CallId | Call identifier for agent, customer, or non consultation outbound call. | Int |
| RouteActionType | • RouteToAgent<br>• RouteToService<br>• RouteToExternal<br>• RouteToDefault | **CTIRouteType** |
| RouteDestination | The AgentLoginName, ServiceId, external number or address of the third party to dial. | String |
| UserData | 20 user defined fields | string[] |

4

# Chapter 4
# Events

The client application receives events through invocation of webservices hosted on the client side. The events generated by CTIPS are encapsulated under one generic type (CTIEvent) and needs to be cast into an appropriate event type (specified in the following sections) based on EventCategory and EventType specified in the event. Clients are supplied with expected Web Services Description Language (WSDL) that they need to implement to support receiving of events.

See Appendix A, Data Types for details on data types.

## Event Description

The clients to CTIPS implement a webservice that supports HTTP sessions and has three methods as described in the following sections:

### Void PublishCTIEvent (CTIEvent event)

This method is used by CTIPS to send an event (an event to which the client has subscribed or a system event) generated by the Unified IP platform to the client. The CTIEvent is cast into an appropriate type based on the EventCategory and EventType specified in the CTIEvent.

**Note:** This method does not block and returns within milliseconds.

### Int HeartbeatMessage()

This method is used by CTIPS to determine if the CTIPS client is still operational. The return from HeartbeatMessage is not checked for any specific return value.

**Note:** This method does not block and returns immediately.

If this method fails, CTIPS determines that the network connectivity to the client is lost or the client is no longer operational and disposes its connection to and from the CTIPS client. It is the client's responsibility to reconnect. At the time of reconnect, the heartbeat messaging is restarted.

Events from the CTIPS to the client are sent over HTTP, allowing the client and the CTIPS to be separated by process and machine boundaries.

### Int StartSession(UserCredentials userinfo, string sessionId)

This method is called by CTIPS when a client calls the OpenConnection() method. CTIPS sends the client user information, (that is, username and password) as well as the client's session Id. The client validates this information and can start the heartbeat timer.

## Event Messages

Significant events can occur at any time after a client has established a session with CTIPS, set up its event filtering, and is online. The client uses the event filtering mechanism described in ***Registration and Filtering Requests*** to ensure that unwanted and unnecessary messages are not sent to the client.

An event message is structured to contain an event header followed by event specific information. Events are grouped into the following categories: agent, call, route and system

## Event Header

The fields depicted in the table below are common to every event sent to the client. These fields are set if the data is present. If the data is not present, the fields are set to 0. The RequestId is used to correlate any event with a request made by a client.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to:<br>• EventCategoryAgent<br>• EventCategoryCall<br>• EventCategoryRoute<br>• EventCategorySystem | **CTIEventCategory** |
| EventType | Event Type (that is, CallCleared) | **CTIEventType** |
| RequestId | Request Identifier. | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |

### Event Categories - CTIEventCategory

The EventCategory field of the event header contains an enumeration CTIEventCategory that depicts the category to which the event belongs. Events are grouped in the following categories:

- Agent - agent login, logout, readiness state, and audiopath state events
- Call - call events related to the active call between the agent, customer, and third party.
- Route - inbound call routing related events
- System - other system events, that is Snapshot Update, Component Status, Service, LinkStatus Event

### EventType – Event Message Type

The EventType field of the event header contains an enumeration of Event Types that depict the type of event that is received. Event types are named to reflect the actual Unified IP call, agent, routing or system state change that generated the event (that is, CallEstablished, AgentStateChange, RouteRequest, and others.)

### RequestId

The RequestId field of the event header contains an integer that matches the RequestId of the original request issued by the client. See Chapter 3, Requests for more information on RequestId. If the event is not a product of a client request (an unsolicited event) this field is set to 0.

## Agent Events Descriptions

The agent events interface defines the contract of specific events that the client expects to receive. The definition of all agent events is described below:

### AgentLoginStateChange Event

This event is generated after issuing a **LoginAgent** request or **LogoutAgent** request. This event describes the login state of the agent.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryAgent | **CTIEventCategory** |
| EventType | Set to AgentLoginStateChange | **CTIEventType** |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| SwitchId | Switch Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| AgentLoginName | Agent's login name | String |
| ResourceGroupId | Resource group identifier | Int |
| CircuitId | Circuit identifier | Int |
| ChannelId | Channel identifier | Int |
| LoginState | AgentLoggedIn, AgentLoggedOut | **CTIAgentLoginState** |
| AgentWorkGroupId | Set if LoginState is AgentLoggedIn otherwise -1 | Int |
| AgentCapabilities | Set to<br>• Invalid<br>• Standard Agent<br>• Director<br>• On Demand | **CTIAgentCapabilities[]** |
| AgentStationAddress | Agent's station | String |
| AgentMailbox | Agent's email address | String |
| ServiceState | An array of service/service state associated with the agent. | **ServiceState[]** |
| SipIpAddress | For future use | string |
| SipPortNumber | For future use | Int |
| ServiceState | Services and their states that are associated with this agent. | **CTIServiceInfo[]** |
| AgentCTIId | Static agent identifier that is always associated with this agent when the agent logs in. | **Int** |

## AgentLoginRequestFailed Event

This event is sent on agent requests that could not be successfully carried out. This event is generated when there is a failure after issuing one of the following requests: **LoginAgent** and **LogoutAgent**.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryAgent | **CTIEventCategory** |
| EventType | Set to AgentLoginRequestFailed | **CTIEventType** |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| AgentLoginName | Agent's login name | String |
| FailureType | Set to one of the Failure Type. | **CTIFailureType** |
| FailureReason | Text description of the failure. | String |

## AgentStateChange Event

This event describes the agent's readiness state with respect to handling calls after an agent is logged on. This event is generated after issuing a LoginAgent, LogoutAgent, Accept Call, ClearCall, SetAgentState, GetAgentState, or SetDisposition request.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryAgent | **CTIEventCategory** |
| EventType | Set to AgentStateChange | **CTIEventType** |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SwitchId | Switch Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| AgentState | Set to one of the following:<br>• Invalid<br>• NotReady<br>• NotReadyPark<br>• Idle<br>• PendingActive<br>• Active<br>• Wrap<br>• ActiveWrap<br>• WrapWarning<br>• Reserved<br>• MultiLine | **CTIAgentState** |
| ServiceIds | A list of all service identifiers associated with this agent. | int[] |
| SipIpAddress | For future use. | string |
| SipIpPort | For future use. | Int |

## AudiopathStateChange Event

When an agent is logged on, this event describes the agent's media state (currently audio only.)

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryAgent | **CTIEventCategory** |
| EventType | Set to AudioPathStateChange | **CTIEventType** |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| ResourceGroupId | Resource Group identifier | Int |
| CircuitId | Circuit Identifier | Int |
| ChannelId | Channel Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| AudioPathState | Set to one of the following<br>• None<br>• Connected<br>• Disconnected | **CTIAudioPathState** |
| ServiceIds | A list of all service identifiers associated with this agent. | Int[] |
| SipIpAddress | For future use. | string |
| SipIpPort | For future use. | Int |

## EnterPasscode Event

This event is sent when “Bypass 3-digit passcode” option on the Gateway Configuration Window is not checked and all agents associated with the gateway are required to enter a 3-digit passcode when logging on.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryAgent | **CTIEventCategory** |
| EventType | Set to EnterPasscodeEvent | **CTIEventType** |
| RequestId | Request identifier | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| ResourceGroupId | Resource group identifier | Int |
| PasscodeDigits | A 3 digit string generated by CenterCord. The agent must enter this passcode in order to proceed with the login | String |
| ReEnterFlag | This flag is false when the event is sent for the first time during the agent login process. If the agent does not enter the correct passcode, this even is sent again with the ReEnterFlag set to true. | Bool |

| Field Name | Description | Type |
|---|---|---|
| ServiceIds | A list of service identifiers associated with the agent. | Int[] |

## ServiceStateChange Event

This event is sent to indicate state changes associated with existing services. The event also contains an array of AgentId's associated the service.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryAgent | **CTIEventCategory** |
| EventType | Set to ServiceStateChange | **CTIEventType** |
| RequestId | Request identifier | Int |
| Tenant Id | Tenant Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | Service identifier | Int |
| AgentId | Agent Identifier | Int |
| ServiceState | Set to the following<br>• NoState<br>• Active<br>• InActive<br>• AddService<br>• RemoveService<br>• UnManned<br>• Holiday<br>• Pause<br>• Failed<br>• Wait<br>• Activating | **CTIServiceState** |
| RequestIdList | An array of AgentIds associated with the service that changed state. | Int[] |

### AgentRequestFailed Event

This event is sent on agent requests that could not be successfully carried out. This event is generated when there is a failure after issuing one of the following requests: **SetAgentState**, **GetAgentState**, or **SetDisposition**.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryAgent | **CTIEventCategory** |
| EventType | Set to AgentRequestFailed | **CTIEventType** |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| FailureType | Set to one of the Failure Type | **CTIFailureType** |
| FailureReason | Textural description of the failure. | String |

## Call Events Descriptions

The call events interface defines the contract of specific call events that the client can expect to receive.

### CallCleared Event

The CallCleared event indicates that a call has been disconnected by either the near end or the far end. The call data for this call remains valid until the CallEnd event is generated.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallCleared | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| ConsultationCallId | Consultation call identifier. | Int |
| ClearingPartyType | Set to NearEnd or FarEnd depending on whether the agent or customer hangs up the call first. | **CTIClearingPartyType** |

## CallCompleted Event

CallCompleted event is generated when all call segments are complete. This includes the agent wrap time if applicable for a call segment. The following fields listed below in the Call Completed event will have the same values as in the Call Started event.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallCompleted | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| ResourceGroupId | Resource group Id | Int |
| CircuitId | Circuit identifier | Int |
| ChannelId | Channel identifier | Int |

## CallConferenced Event

CallConferenced event indicates that a call conference has been established.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallConferenced | **CTIEventType** |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| ConsultingCallId | Call Identifier of the consultating agent involved in the conference. | Int |
| ConsultedCallId | Call Identifier of the consultated agent involved in the conference. | Int |
| ConsultationAgentId | Consultating agent identifier. | Int |
| ConferenceCallId | CallIdentifier of the conference. | Int |

## CallConnected Event

The CallConnected event indicates that a call is presented to a device in either the Ringing or Entering Distribution modes of the alerting state. This event contains data required for the call screen pop. All available relevant call data is delivered with the CallConnected event in the CallInfo array. If the agent is allowed to manually accept or reject the call, the ResponseRequired is set to true.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallConnected | **CTIEventType** |

| Field Name | Description | Type |
|---|---|---|
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP Site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | SiteId where the call originated. | Int |
| CallInfo | Call-related information and user defined call data. | **CTICallInfo[]** |
| ResponseRequired | True if the agent can accept or reject the call | Bool |
| RejectReasonRequired | True if the agent must supply a reason for rejecting a call. | Bool |
| PlayAudioAlert | True if the agent is configured to hear an audio alert when a call is connected to his device. | Bool |
| CallType | The type of call that is queued (for ex: DID, ACD, and others.)<br>Set to the one of the following:<br>• CallTypeUnavailable<br>• InboundAcd<br>• InboundDid<br>• InboundConsultation<br>• InboundInternal<br>• InboundIvr<br>• OutboundAod<br>• OutboundConsultation<br>• OutboundConsultationTransfer<br>• OutboundMakecall<br>• Conference<br>• InboundIpniq | CTICallType |
| OriginatingAgentId | Set to the originating agent if available. | Int |
| ResourceGroupId | Resource group Id | Int |

| Field Name | Description | Type |
|---|---|---|
| CircuitId | Circuit Identifier | Int |
| ChannelId | Channel Identifier | Int |

## CallConsultationEvent

CallConsultation event indicates that a consultation call has been established.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallConsultation | **CTIEventType** |
| RequestId | Request Identifier | Int |
| TenantId | Tenant Identifier | Int |
| SiteId | Unified IP Site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | SiteId where the call originated. | Int |
| ConsultationCallId | Call Identifier of the consultation call involved in the conference. | Int |
| ConsultationAgentId | Consultation Agent Identifier | Int |

## CallDataUpdate Event

This event that is generated after successfully processing a SetCallData or GetCallData request. If the request fails, the event contains a verbal description of the failure in the FailureReason field.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |

| Field Name | Description | Type |
|---|---|---|
| EventType | Set to CallDataUpdate | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | SiteId where the call originated. | Int |
| CallData (optional) | Call-related user defined variable data. See Appendix A, CTICallDataItem | **CTICallDataItem**[] |
| UpdateType | Status of the request. Set to one of the following:<br>• GetCallDataReturned<br>• GetCallDataNotReturned<br>• SetCallDataSucceeded | CTICallDataUpdateType |

## CallDelivered Event

The CallDelivered event indicates that a call is being queued to a call service. If mandatory attention retainers are enabled, they are played to the caller.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallDelivered | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is queued to. | Int |
| AgentId | Agent Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| CallType | The type of call that is queued (that is, DID, ACD and others.)<br>See CTICallType. | **CTICallType** |
| ANI | The number of the calling party. | string |
| DNIS | The dialed number. | string |

## CallEnd Event

The CallEnd event is generated when all connections to a call have been cleared. At this point the CallId and all associated Call Data are no longer valid.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallEnd | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The service id that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| AgentDisposition | Call disposition set by the agent. | Int |
| CallData (optional) | Call-related user defined variable data. See Appendix A, CTICallDataItem | **CTICallDataItem**[] |

## CallEstablished Event

The Established event indicates that a call was answered at a device. The CallEstablished event is generated when an agent accepts the call.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallEstablished | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| MediaType | Set to VoiceAnalog or VoiceSIP | **CTIMediaType** |
| SipIpAddress | Set if Media type = VoiceSIP | String |
| SipPortNumber | Set if Media type = VoiceSIP | Int |
| ResourceGroupId | Resource group Id | Int |
| CircuitId | Circuit Identifier | Int |
| ChannelId | Channel identifier | Int |
| ANI | The number of the calling party | String |
| DNIS | The dialed number | String |

## CallHeldEvent

CallHeld event indicates that a specific call is placed on hold.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallHeld | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |

| Field Name | Description | Type |
|---|---|---|
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |

## CallOffered Event

The CallOffered Event is generated when an inbound call arrives at the Unified IP system for processing. This indicates that a call is in a pre-delivery state. There is very little call data available at this point and the client application cannot act on this call until it has entered the Delivered state. This event is mainly used for monitoring and statistical reporting to determine when a call arrives in the system.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallOffered | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Set to 0 | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| ResourceGroupId | Resource group Id | Int |
| CircuitId | Circuit identifier | Int |
| ChannelId | Channel identifier | Int |
| ANI | The number of the calling party. | string |
| DNIS | The dialed number. | string |

## CallOriginated Event

The CallOriginated Event is generated whenever an outbound call is initiated by the Unified IP system. If the outbound call was initiated due to a client MakeCall or ConsultationCall request, the event contains the RequestId of the original request. If it is not initiated by a client MakeCall or ConsultationCall request, this field is set to 0.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallOriginated | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier or set to 0 | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| DestType | Agent, Service, or External | **CTIDestType** |
| Destination | The AgentId, ServiceId, or number to be dialed. | String |
| ResourceGroupId | Resource group Id | Int |
| CircuitId | Circuit identifier | Int |
| ChannelId | Channel identifier | Int |
| CallProgressSignal | Dial tone dispositions | **CTICallProgressSignalType** |

## CallRequestFailed Event

This event is returned to indicate that a request was not successfully carried out.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallRequestFailed | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| FailureType | Set to one of the CTIFailureType codes. (See Appendix A, CTIFailureType) | CTIFailureType |
| FailureReason | Textural description of the failure | StringFailureReason |

## CallRetrieved Event

A CallRetrieved event indicates that a call that had previously been placed on hold has been retrieved.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallRetrieved | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |

## CallStarted Event

CallStarted event indicates that a new call has arrived or a new call has been initiated by the Unified IP system.

For this event as well as the CallCompleted event, the definition of a call is as follows: It is defined as a set of one or more call segments that represent a “cradle to grave” interaction with the system. The Call Started event will get generated depending on the call type. This event occurs prior to the CallOffered event.

List of call types:

- Inbound – the Call Started event is generated when the call arrives at the system.
- Outbound campaign – the Call Started event is generated when the call commences dialing.
- Outbound Preview - the Call Started event is generated when the agent gets the screen pop (Note: Switch Id, Circuit Id, and Channel Id will be empty)
- Outbound Manual - the Call Started event is generated when the call is initiated
- Internal Manual - the Call Started event is generated when the call is initiated (Note: Switch Id, Circuit Id, and Channel Id will be empty)
- IPNIQ - the Call Started event is generated when the virtual request arrives at the system and the M3 script is started.
- Enterprise transfer – the Call Started event is generated when the call arrives at the remote system – there is not an M3 script involved

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallStarted | **CTIEventType** |
| TenantId | Tenant Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| ResourceGroupId | Resource group Id | Int |
| CircuitId | Circuit identifier | Int |
| ChannelId | Channel identifier | Int |

## CallTransferred Event

The CallTransferred event indicates that a call transfer was completed.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to CallTransferred | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| ConsultingCallId | Call Identifier of the consulting agent involved in the conference | Int |
| ConsultedCallId | Call Identifier of the consulted agent involved in the conference | Int |

| Field Name | Description | Type |
|---|---|---|
| ConsultationAgentId | Consulting agent identifier. | Int |
| TransferredCallId | Call Id associated with the transfer. | Int |
| TransferredCallSwitchId | The switch Id associated with the transfer. | Int |

### DigitsDialed Event

This event is generated after successfully processing the SendDTMFDigits request.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryCall | **CTIEventCategory** |
| EventType | Set to DigitsDialed | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | The Service Id of the service that the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |
| DigitString | The digits dialed. | String |

## Call Route Event Descriptions

The route events interface defines the contract of specific route events that the client may receive.

### RouteRequest Event

The RouteRequest event is an unsolicited event that is generated when a call arrives at an inbound service configured for external routing in the Unified IP system. The event indicates to

the client that a call with the given DNIS and ANI (optional) requires a route. The route must be specified within a given time period or the Unified IP system applies default routing action as configured in the inbound service.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryRoute | **CTIEventCategory** |
| EventType | Set to RouteRequest | **CTIEventType** |
| TenantId | TenantIdentifier | Int |
| RequestId | Set to 0 | Int |
| SwitchId | Switch Identifier | Int |
| SiteId | Unified IP site identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| RouteTimeout | The maximum number of seconds in which the external routing device must respond or default routing is applied. | Int |
| ResourceGroupId | Resource group id | Int |
| CircuitId | Channel Identifier | Int |
| ChannelId | Channel Identifier | Int |
| Media type | Depicts the type of media engaged in the call. | **CTIMediaType** |
| SipIpAddress | For future use. | String |
| SipPortNumber | For future use. | String |
| CallInfo | Call-related information and user defined call data. See Appendix A, CTICallInfo | **CTICallinfo[]** |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |

## RouteUsed Event

The RouteUsed event is generated when the Unified IP system routes a call that was initially delivered to an inbound call service configured for external routing. If the call was successfully routed due to a RouteSelect request from a CTI client, the RequestId field of the event contains the RequestId of the RouteSelect

request. If the call was routed using default routing rules because a valid RouteSelect was not received within the configured maximum queue time, the RequestId field of the event contains 0.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryRoute | **CTIEventCategory** |
| EventType | Set to RouteUsed | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | RequestId or 0 | Int |
| SwitchId | Switch Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| RouteStatus | Status associated with previous Route Select request.<br>• Route Success<br>• DefaultRouteUsed | **CTIRouteStatusType** |
| RouteType | • Invalid<br>• RouteToAgent<br>• RouteToService<br>• RouteToExternal<br>• RouteToDefault<br>• IVR<br>• Voicemail | **CTIRouteType** |
| Route Destination | The AgentId, ServiceId, or number to be dialed. | String |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |

## RouteRejected Event

This event is sent in response to a previous RouteSelect request to indicate that the request was not successful.

**Note:** If the RouteRequest times out due to the MaxQueue time being exceeded prior to receiving a RouteSelect, a RouteSelectFailed event is not sent. A RouteUsed event is sent with the RequestType and RequestId set to 0 and a default call routing is used. See RouteSelect and RouteUsed Event.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryRoute | **CTIEventCategory** |
| EventType | Set to RouteRejected | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SwitchId | Switch Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| ServiceId | The service Identifier | Int |
| RouteRejectReason | Set to one of the CTIRejectReason codes. See CTIRejectReason. | **CTIRejectReason** |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |

## RouteRequestFailed Event

This event is sent in response to a previous RouteSelect request to indicate that the Unified IP system encountered a failure when attempting to route the call to the selected location. The reasons for the failure might be that the call is no longer valid or the device in the selected route is already busy. A code indicating the type of failure is contained in the CTIFailureType field of the event and a textural description of the failure is contained in the CTIFailureReason field.

**Note:** If the RouteRequest times out due to the MaxQueue time being exceeded prior to receiving a RouteSelect, a RouteRequestFailed event is not sent. A RouteUsed event is sent with the RequestId set to "0" and a default call routing is used. See RouteSelect and RouteUsed Event.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategoryRoute | **CTIEventCategory** |

| Field Name | Description | Type |
|---|---|---|
| EventType | Set to RouteRequestFailed | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SwitchId | Switch Identifier | Int |
| ServiceId | Service Identifier | Int |
| AgentId | Agent Identifier | Int |
| CallId | Call identifier | Int |
| FailureType | Set to one of the CTIFailureReason codes. See CTIFailureType | **CTIFailure Type** |
| FailureReason | A textural description of the routing failure. | String |
| CallSequenceId | Call sequence identifier | long |
| CallOriginatingSiteId | Site where the call originated. | Int |

# System Event Descriptions

The system events interface defines the contract of specific system events that the client can receive.

## SnapShotUpdate Event

The Snapshot event is generated in response to a client issuing a Snapshot request. Multiple SnapShotUpdate events can be generated for a particular SnapShot request depending on the type of information being requested (that is, Agent, Call, Service or any combination) and the number of agents, calls, or services active in the Unified IP system at the time of the SnapShot request.

The SnapShotUpdate event contains the relevant data for all logged in agents, active calls, or configured services depending on what was requested in the SnapShot request. Each SnapShot Update event only contains information about one particular entity type (that is, Agents, Calls, or Services.)

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategorySystem | **CTIEventCategory** |

| Field Name | Description | Type |
|---|---|---|
| EventType | Set to SnapShotUpdate | **CTIEventType** |
| TenantId | Tenant Identifier | Int |
| RequestId | Request Identifier | Int |
| SiteId | Unified IP Site Identifier | Int |
| SnapShotType | CTIEntityType indicates the type entity information contained in the SnapShotUpdate event:<br>• AgentInfo<br>• CallInfo<br>• ServiceInfo | **CTIEntityType** |
| AgentData | CTISnapShotAgentData contains agent data pertaining to an agent logged into the Unified IP system at the time of the SnapShot request. | **CTISnapShotAgentData** |
| CallData | CTISnapShotCallData contains call data pertaining to an active call in the Unified IP system at the time of the SnapShot request. | **CTISnapShotCallData** |
| ServiceData | CTISnapShotServiceData contains service information for a service configured in Unified IP at the time of the SnapShot request. | **CTISnapShotServiceData** |
| SnapShotComplete | SnapShotComplete indicates that all SnapShotUpdate events for all SnapShotTypes requested were sent. | Bool |

## LinkStatus Event

The LinkStatus event is an unsolicited event that informs the CTIPS client that CTIPS has lost its connection to the Unified IP system or the Unified IP system has lost its connection to one of its switches.

| Field Name | Description | Type |
|---|---|---|
| EventCategory | Set to EventCategorySystem | **CTIEventCategory** |
| EventType | LinkStatus | **CTIEventType** |
| TenantId | Set to 0 | Int |
| RequestId | Set to 0 | Int |
| SwitchId | Switch Identifier | Int |

| Field Name | Description | Type |
|---|---|---|
| SiteId | Unified IP site identifier | Int |
| ServiceId | The service the call is attributed to. | Int |
| AgentId | Agent Identifier | Int |
| ComponentType | Set to one of the following:<br>• CenterCord<br>• Gateway | **CTIComponentType** |
| ComponentState | Set to one of the following:<br>• Up<br>• Down | **CTIComponentState** |

5

# Chapter 5
# Using the CTI Portal Server

This chapter describes various examples for implementing applications that use the CTIPS API.

## Client Application Startup and Shutdown Sequence

The following examples describe initial startup sequence of steps the client application implements:

- Opening a Session with CTIPS
- Registering for Events
- Adding Filters
- Placing a Session Online
- Requesting a Snapshot
- Placing a Session Offline
- Closing a Session with CTIPS

### Opening a Session With CTIPS

To allow access to the functionality provided by Aspect Unified IP, a client application must first establish a connection (session) with the Unified IP CTI Portal Server.

- A connection from the client application to the CTIPS (a session) is created by calling the OpenConnection() method with the appropriate information for the CTIPS instance to connect to.
- If the CTIPS receives a properly formatted OpenConnection() it acknowledges it by sending an OpenConnection response to the client application.
- If the OpenConnection is successful, then a unique session identifier is passed back to the client. This session identified must be used in all subsequent requests by the client.

  If the CTIPS receives bad or incomplete information from the OpenConnection request, it throws a fault exception containing an informative error message.

## Registering for Events

After opening a session with CTIPS, the client can register for events it is interested in receiving. The client application calls the AddMessageClass method and sets the MessageClass field to one of the following categories:

- EventCategoryAgent
- EventCategoryCall
- EventCategoryRoute,
- EventCategorySystem
- Placing a Session OffLine
- Closing a session with CTIPS

To register for all events, the client must call the AddMessageClass method for each category of events.

See How to Register for Events for more detailed information.

## Adding Filters

After opening a session with CTIPS and registering for events, the client can set additional filters when it wants to limit the amount of events it receives. For example, if the client is only interested in receiving events for a particular agent, switch, or service. The client application must call the AddFilter method and set the FieldType and FieldValue parameters accordingly. For example, to register for all events for service id 3, set the FieldType parameter to FT_SERVICE_ID and set the FieldValue parameter to "3".

## Placing a Session Online

After opening a session with CTIPS, registering for events, and adding additional filters if needed, the client application must go online by invoking the OnLine method making it eligible to receive messages from the Unified IP CTI Portal Server (CTIPS). After placing the client application online, the Unified IP CTI Portal Server generates a heartbeat event to the client indicating that the session is alive and online.

If the client has properly responded to the Heartbeat event, the Unified IP CTI Portal Server generates another Heartbeat event after a preconfigured time period has expired. The client application must then respond to this heartbeat event.

**Note:** This cycle of the Unified IP CTI Portal Server generating a heartbeat event and the client application sending back a heartbeat response message must continue for the duration of time that the client session is online. If the client application does not respond to the Heartbeat event before the timeout period expires, the Unified IP CTI Portal Server closes the connection to the client.

If the client is a monitoring client only and has connected to the CTI Portal Server, the client needs to closely monitor the heartbeat messages from the CTI Portal Server. The client must determine if it needs to reconnect to the CTI Portal Server. If the client does not receive any heartbeat messages from the CTI Portal Server, it must attempt to reconnect to the server.

**Note:** The Unified IP system allows configuring an allowable number of Missed Heartbeats through the Server Configurator to occur before considering the session closed and taking failover measures.

## Requesting a Snapshot

After a client creates a session, registers for events, and is placed online, it can send a Snapshot request to the CTIPS to obtain information on system-wide calls, agents, and services. A client can issue a Snapshot request for any combination of agent, call, and service state information.

1. The CTIPS must receive a properly formatted Snapshot() request. The CTIPS acknowledges receipt of the request by sending a Snapshot response to the client application.
2. The CTIPS forwards the request to the Unified IP system.
3. The Unified IP system gathers the required information and sends the information back to the CTI Portal Server.

The CTIPS sends a SnapShotUpdate event for every active call, agent logged in, or service depending on what event type was indicated in the Snapshot request. These SnapShotUpdate events provide the client application with a snapshot of the state of the system.

## Placing a Session Offline

A client can stop sending requests and receiving events without disabling a session. The session can be placed offline.

1. The session between the client application and the CTIPS can be placed offline by invoking the OffLine() method with the appropriate information for the CTIPS session.
2. The CTIPS must receive a properly formatted Offline() and acknowledge it by sending an OffLine response to the client application.
3. After the session is placed OffLine, the CTIPS no longer accepts requests or sends events to the client application with the exception of an OnLine request.

**Note:**

- If the CTIPS receives bad or incomplete information from the OffLine request, it returns an OffLine response with an informative error code status.
- If the Offline request fails after returning an OffLine response, the CTIPS generates an OffLine event with an informative error code status.

## Closing a Session With CTIPS

To close the connection from the client application to the CTIPS, the application must call the CloseConnection() method with the appropriate information for the CTIPS instance that is connected to the application.

1. The CTIPS must receive a formatted CloseConnection(). The CTIPS acknowledges it by sending a CloseConnectrion response to the client application.
2. After the client receives the CloseConnection response, the session is ended and no further requests, responses or events are generated for this session.

**Note:**

- If the Unified IP CTI Portal Server receives bad or incomplete information from the CloseConnection request, it returns a CloseConnection response with an informative error code status.
- If the CloseConnection fails after returning a CloseConnection response, the CTIPS generates a CloseConnection event with an informative error code status.

# How to Register for Events

After opening a session with CTIPS, the client can register for events. Asynchronous or unsolicited events are messages sent to the client that indicate an event to which the application can respond (for example: CallDeliveredEvent). The client can add more than one filter.

To register for events, the client application must call the AddMessageClass method and set the MessageClass field to one of the following categories:

- EventCategoryAgent
- EventCategoryCall
- EventCategoryRoute
- EventCategorySystem

To register for all events, the client must call the AddMessageClass method for each category.

To register to receive all agent, call, routing, and system events, the client application needs to call the AddMessageClass method for each category of events.There are four categories of event. The AddMessageClass method needs to be called four times. Each time the call is made, the MessageClass field must be set to the appropriate event category.

The CTIPS tracks all clients and their filters and publishes events to the clients according to their subscriptions and filters. See Appendix C, Summary of Services and Events.

# Examples of Registering an Event

The following examples describe how the client registers for specific events:


- Registering for Agent Events
- Registering for Specific Agent Events
- Registering for Call Events
- Registering for Routing Events
- Registering for Specific Service Ids


## Registering for Agent Events

Prior to invoking an OnLine method, applications can register for agent events only.

To register for agent events only, the client application invokes the AddMessageClass () method and sets the MessageClass field to CTIMessageClass.EventCategoryAgent.

## Registering for Specific Agent Events

Prior to invoking the OnLine method, client applications can register for specific agent events.

1. To register for specific agent events, the client application must first invoke the AddMessageClass () method and set the MessageClass field to CTIMessageClass.EventCategoryAgent.
2. The client application then invokes the AddFilter method. The client needs to set the FieldType to FT_AGENT_ID and the FieldValue to the agent's login name.

## Registering for Call Events

Before going online, applications may register for call events.

To register for call events, the client application invokes the AddMessageClass () method and sets the MessageClass to CTIMessageClass.EventCategoryCall.

## Registering for Routing Events

Before going online, client applications can register for routing events.

1. To register for route requests and other routing events, the application invokes the AddMessageClass () method and sets the MessageClass field to CTIMessageClass.EventCategoryRoute.

## Registering for Specific Service Ids

Before going online, client applications can register for specific Service Id events.

To register for specific service ids, the client application must first invokes the AddMessageClass () method and sets the MessageClass field to CTIMessageClass.EventCategorySystem. Then the client application must call AddFilter, setting the FieldType to FT_SERVICE_ID and the FieldValue parameter to the actual service id of "3".

A

# Appendix A
# Data Types

This appendix contains a listing of all CTIPS Data Types.

## User Credentials

```
public class UserCredentials
 {
    public string UserName;
    public string Password;
  }
```

## CTISnapShotAgentData

The CTISnapShotAgentData is returned in the SnapShotUpdate event and contains all available agent information for a particular agent that is logged into the Unified IP system.

```
public class CTISnapShotAgentData
{
        public int AgentId;
        public string AgentLoginName;
        public CTIAgentCapabilities[] AgentCapabilities;
        public int AgentWorkGroupId;
        public int CallId;
        public int ResourceGroupId;
        public int ChannelId;
        public int CircuitId;
        public int SwitchId;
        public CTIAgentState AgentState;
        public CTIServiceInfo[] Services;
        public CTIMediaType MediaType;
```

```
        public string SipIpAddress;
        public int SipPortNumber;
        public string AgentStationAddress
    }
```

# CTIAgentCapabilities

```
public enum CTIAgentCapabilities
  {
   INVALID
   StandardAgent,
   Director,
   OnDemand
   }
```

# CTIAgentState

```
public enum CTIAgentState
  {
   INVALID
   NotReady,
   NotReadyPark,
   Idle,
   PendingActive,
   Active,
   Wrap,
   WrapWarning,
   Reserved,
   MultiLine
   }
```

# CTICallProgressSignalType

```
public enum CTICallProgressSignalType
  {
   INVALID = 0
   None,
   Ringing,
   Busy,
   SitTone
   }
```

# CTIServiceInfo

```
public class CTIServiceInfo
  {
     public int ServiceId;
     public CTIServiceState ServiceState;
  }
```

# CTIServiceState

```
public enum CTIServiceState
  {
   NoState,
   Active,
   InActive,
   AddService,
   RemoveService,
   UnManned,
   Holiday,
   Pause,
   Failed,
   Wait,
   Activating
  }
```

# CTIMediaType

```
public enum CTIMediaType
 {
  INVALID
  VoiceAnalog,
  VoiceSip,
  Chat,
  Email,
  IM,
  Workflow
 }
```

# CTISnapShotCallData

The CTISnapShotCallData is returned in the SnapShotUpdate event. It contains all available call information for a particular call.

```
public class CTISnapShotCallData
{
    public int CallId;
    public long CallSequenceId;
    public int CallOriginatingSiteId;
    public int ResourceGroupId;
    public int ChannelId;
    public int CircuitId;
    public int SwitchId;
    public int ServiceId;
    public CTIDestinationType DestType;
    public string Destination;
    public CTICallState CallState;
    public string ANI;
    public string DNIS;
    public CTICallType CallType;
    public int AgentId;
```

```
    public CTIMediaType MediaType;
    public string SipIpAddress;
    public int SipPortNumber;
}
```

# CTIDestinationType

```
public enum CTIDestinationType
{
 INVALID
 Agent,
 Service,
 External,
 NoDestination
}
```

# CTICallState

```
public enum CTICallState
{
 INVALID
 CallCleared,
 CallConnected,
 CallConsulting,
 CallDelivered,
 CallEnd,
 CallEstablished,
 CallError,
 CallStateUnknown,
 CallOffered,
 CallOriginated,
 CallTransferred,
 CallOnHoldDialing,
 CallHeld,
 CallRetrieved,
 CallConferenced,
```

```
 DigitsDialed
}
```

# CTICallType

```
public enum CTICallType
{
 CallTypeUnavailable,
 InboundAcd,
 InboundDid,
 InboundConsultation,
 InboundInternal,
 InboundIvr,
 OutboundAod,
 OutboundConsultation,
 OutboundConsultationTransfer,
 OutboundMakecall,
 Conference,
 InboundIpniq
}
```

# CTISnapShotServiceData

The CTISnapShotServiceData is returned in the SnapShotUpdate event. It contains all available service information for a particular UIP service.

```
public class CTISnapShotServiceData
{
    public int ServiceId;
    public CTIServiceState ServiceState;
    public bool ExternalRoutingEnabled;
}
```

# CTIMessageClass

**public enum CTIMessageClass**

```
{
        INVALID,
        EventCategoryAgent,
        EventCategoryCall,
        EventCategoryRoute,
        EventCategorySystem
}
```

# CTIEventCategory

**public enum CTIEventCategory**

```
{
   INVALID,
   EventCategoryAgent,
   EventCategoryCall,
   EventCategoryRoute,
   EventCategorySystem
}
```

# CTIEventType

**public enum CTIEventType**

```
{
 INVALID,
 Filter,
 AgentLoginStateChange,
 AgentLoginRequestFailed,
 AgentStateChange,
 AgentRequestFailed,
 AudioPathStateChange,
 EnterPasscode,
```

```
    ServiceStateChange,
    CallCleared,
    CallCompleted,
    CallConferenced,
    CallConnected,
    CallConsultation,
    CallDataUpdate,
    CallDelivered,
    CallEnd,
    CallEstablished,
    CallHeld,
    CallOffered,
    CallOriginated,
    CallRequestFailed,
    CallRetrieved,
    CallStarted,
    CallTransferred,
    DigitsDialed,
    RouteRequest,
    RouteRequestFailed,
    RouteUsed,
    RouteReject,
    LinkStatus,
    SnapShot,
    SnapShotUpdate,
    SnapShotRequestFailed
}
```

# CTICallInfo

The CTICallInfo contains all available caller data usually associated with the agent screen pop. Most of the elements of the CTICallInfo are optional and will be set to NULL if the data is not available. The CTICallInfo contains the following elements:

```
public class CTICallInfo
{
    public string ANI;
```

```
    public string DNIS;
    public string FirstName;
    public string LastName;
    public string PhoneNumber;
    public string CallerId;
    public CTICallDataItem[] CallData;
}
```

# CTICallDataItem

The CTICallDataItem is a key value pair of user definable call associated data (that is, account number, street address, city, and others)

```
public class CTICallDataItem
{
    public string key;
    public string value;
}
```

# CTICallDataUpdateType

```
public enum CTICallDataUpdateType
{
 INVALID = 0,
 GetCallDataReturned = 1,
 GetCallDataNotReturned = 2,
 SetCallDataSucceeded = 3
}
```

# CTIEntityType

```
public enum CTIEntityType
{
 INVALID = 0,
 AgentInfo = 1,
 CallInfo,
```

ServiceInfo

}

# CTIFieldType

**public enum CTIFieldType**

{

INVALID = 0,

FT_AGENT_ID,

FT_SERVICE_ID,

FT_SWITCH_ID,

FT_TENANT_ID

}

# CTIAgentLoginState

**public enum CTIAgentLoginState**

{

INVALID = 0,

AgentLoggedIn,

AgentLoggedOut

}

# CTIClearingPartyType

**public enum CTIClearingPartyType**

{

INVALID,

NearEnd,

FarEnd,

Internal

}

# CTIRouteType

**public enum CTIRouteType**

```
{
 INVALID,
 RouteToAgent,
 RouteToService,
 RouteToExternal,
 RouteToDefault,
 Ivr,
 Voicemail
 }
```

# CTIRouteStatusType

**public enum CTIRouteStatusType**

```
{
 INVALID = 0,
 RouteSuccess = 1,
 RouteDefaultUsed
 }
```

# CTIRejectReason

**public enum CTIRejectReason**

```
{
 INVALID,
 NoRouteToDestination,
 RouteTimeout,
 CallCleared,
 CallAlreadyRouted,
 DestinationBusy,
 UnknownReason
 }
```

# CTITransferType

**public enum CTITransferType**

{

INVALID,

Blind = 1,

Consultation = 2

}

# CTIAudioPathState

**public enum CTIAudioPathState**

{

INVALID,

Connected,

Disconnected

}

# CTIFailureType

**public enum CTIFailureType**

{

INVALID,

InvalidPasscode,

InvalidAgent,

InvalidCallId,

DestinationBusy,

DestinationNoAnswer,

DestinationNoRoute,

NoRouteToDestination,

RouteTimeout,

GetCallDataNotReturned,

SetCallDataFailed,

InternalError,

UnKnownFailure,

InternalServerError,
GatewayLoginFailure,
AgentAlreadyLoggedIn,
AgentLicenseExceeded,
InvalidAgentName,
StationNotAvailable,
InvalidStateForRequest,
InvalidSwitchIdentifier,
NoActiveService,
DestinationAgentNotAvailable,
DestinationInvalidSwitchIdentifier,
DestinationInvalidNumber,
DestinationInvalidServiceIdentifier,
TrunkLicenseExceeded,
CustomerHungUpWaitingForTransfer,
ClearCallFailure,
MakeCallFailure,
ConsultationFailure,
CallTransferFailure,
ConferenceFailure,
PlayDigitsFailure,
RequestFailed
}

# CTIComponentType

**public enum CTIComponentType**

{
INVALID,
Gateway = 1,
CenterCord
}

# CTIComponentState

```
public enum CTIComponentState
  {
   INVALID,
   Up,
   Down
  }
```

B

# Appendix B
# Configuration

This appendix includes instructions on how to configure another instance of CTIPS using the Unified IP Server Configurator application. See the *Aspect Unified IP Server Configurator User Guide* for more detailed information on the application.

## Configuring CTIPS Using the Server Configurator

The Unified IP system provides a way to configure the reference to the CTI Portal Server. This reference identifies the server name and port number. This CTI Portal Server allows a place for configuring the Max Number of Allowed Missed Heartbeats and Heartbeat interval.

**Note:** It is the responsibility of the Aspect Unified IP Installer to install and configure the CTI Portal Server.

### Adding a new CTI Portal Server

This section explains how an Implementation Engineer creates the CTIPS using the Unified IP Server Configurator application.

1. In the main window of Server Configurator, click **File->New->Local Site Machines**.
2. Add a new **machine**. Specify the **name of the CTI Portal Server**.
3. Click the **Query Machine** button to determine the IP address of the CTI Portal Server.
4. Click **OK** to save the new machine.
5. Click **Servers** below the tenant that support the CTI Portal Server.
6. Click **New** to add a new Tenant Server.
7. Server Configurator displays a popup menu that displays the list of server types. Included in the list is a new server type called **CTI Portal Server**.
8. On the General tab, enter a **name** for the new server.
9. Select the **machine** that the component runs on and the port number for the CTI Portal Server. The default port number for the CTI Portal Server is nnnn. (The Port Number Text can be changed, so any port number can be entered.)

10. In the **Configuration** tab, specify the Heartbeat interval (default value 30) and Missed Heartbeats (default value 3). Enter a valid user. This user must exist in Active Directory. The Server Configurator enforces the minimum value (1) for the Missed Heartbeats and the maximum value (999).
11. In the Network Cards tab, select the **IP address** associated with the machine selected for the CTI Portal server.

## Modifying an Existing CTI Portal Server

1. On the main menu of Server Configurator, select **Servers.** The list of configured servers display in the right pane.
2. Right-click on the **CTI Portal Server** in the server list.
3. Select **Properties**. A Server Properties Dialog opens.
4. Modify the existing CTI Portal Server.
5. Click **OK** to save the changes.

## Deleting an Existing CTI Portal Server

1. On the main menu of Server Configurator, select **Servers.** The list of configured servers display in the right pane.
2. Right-click on the **CTI Portal Server** in the server list.
3. Click **Delete**.
4. A Confirmation Dialog displays. Confirm the deletion or Cancel to abort the deletion.

C

# Appendix C
# Summary of Services and Events

This appendix includes detailed descriptions of each service and events statistic.

## Summary of Session, Registration, and System Services

| Method Name | Description |
|---|---|
| OpenConnection | The OpenConnection request opens the client connection to the UIP CTI Portal Server. |
| CloseConnection | The CloseConnection request closes the client connection to the UIP CTI Portal Server. |
| OnLine | The OnLine request sets the connection to the ONLINE state, making the client eligible to receive messages from the UIP CTI Portal Server. |
| OffLine | The OffLine request sets the connection to the OFFLINE state, making the client ineligible to receive messages from the UIP CTI Portal Server. |
| AddMessageClass | The AddMessageClass request allows an application to receive all messages in a particular category (that is, agent events, call events, routing events, system events) |
| DeleteMessageClass | The DeleteMessageClass request allows the client to delete a filter based on the class of message. |
| AddFilter | The AddFilter request allows the client to add a filter based on a specific agent, service, or switch id. Adding a filter allows the client to minimize the amount of unnecessary events that the client receives. |
| DeleteFilter | The DeleteFilter request allows the client to delete a filter based on a specific agent, service, or switch id |
| Snapshot | The Snapshot method issues a server request to retrieve the current agent, call, and/or service information. |

# Summary of Agent, Call Control, and Routing Services

| Method Name | Description |
|---|---|
| AcceptCall | Answers a call that is ringing, queued, or being offered to a device. |
| ClearCall | Releases all of the devices associated with the specified call. |
| ConferenceCall | Provides a conference of an existing held call and another active call at a conferencing device. The two calls are merged into a single call at the conferencing device. |
| ConsultationCall | Places an existing active call at a device on hold and initiates a new call from the same device. |
| LoginAgent | Login the agent identified in the request. |
| LogoutAgent | Logout a currently logged in agent. |
| HoldCall | Places a specific connection on hold. |
| MakeCall | Establishes a call between two devices. |
| PanicLogoutAgent | Performs an immediate logout for agents. |
| RetrieveCall | Connects to a call that had previously been placed on hold. |
| TransferCall | Transfers a held call to the consulted party or allows the application to transfer a two-party call that is in a connected state (call must involve at least one trunk. The second party can be an agent.) |
| SetAgentState | Sets the agent state. |
| GetAgentState | Retrieves the agent state. |
| SetDisposition | Allows the application to set the disposition of the agent if required. |
| SendDTMFDigits | Dials a digit sequence for a call that has already been initiated. |
| RouteSelect | The RouteSelect request is used by the client to provide Unified IP with a route destination in response to a RouteRequest Event. |

# Summary of Agent, Call Control, Routing, and System Events

| Event Name | Description |
|---|---|
| AgentLoginStateChange | This event describes the login state of an agent. |
| AgentLoginRequestFailed | This event is sent if the login/logout agent request fails. |
| AgentStateChange | Once an agent is logged in, this event is sent when the agent's state changes or when the agent state is requested through a GetAgentState. |
| AgentRequestFailed | This event is sent if any agent request fails. |
| AudioPathStateChange | Once an agent is logged in, the event describes the agent's audio path state. |
| EnterPasscode | This event is sent when "Bypass 3-digit passcode" option is not selected on the Aspect Unified IP Unified Resource Manager. |
| CallCleared | This event indicates that a call leg has been disconnected. |
| CallCompleted | This event is sent when all call segments are complete. This includes the agent wrap time if applicable for a call segment. |
| CallConnected | This event indicates the presentation of a call to an agent. |
| CallConferenced | This event indicates that a 3-way conference exists between the agent, customer, and consultation party. |
| CallConsultation | This event indicates that a consultation call has been answered. |
| CallDataUpdate | This event is sent in response to SetCallData and GetCallData requests. When sent in response to a GetCallData request, the call data for the specified Call Identifier is contained in the event. |
| CallDelivered | Indicates that a call is being presented to a device or endpoint. |
| CallEnd | This event is generated when an agent leaves the "in wrapup" state. |
| CallEstablished | This event indicates that the incoming call voice path has been connected to the agent, and the incoming connected call is now active. |
| CallHeld | Occurs when a call is placed on hold. |
| CallOffered | Indicates that a call is in a pre-delivery state at a device (prior to ringing indication or delivering ringback). |

| Event Name | Description |
|---|---|
| CallOriginated | This event indicates that a make call or consultation call is in the dialing state. |
| CallRequestedFailed | Indicates that a call request cannot be completed. |
| CallRetrieved | Occurs when a call on hold is retrieved. |
| CallStarted | This event indicates that a new call has arrived or a new call has been initiated by the Unified IP system. |
| CallTransferred | This event indicates that a call has been successfully transferred. This is the result of either a blind or consultation transfer request. |
| DigitsDialed | This event indicates that DTMF digits have been dialed. |
| RouteRequest | This event indicates to the client that a call with the given DNIS and ANI requires a route. The client must send a subsequent Route Select request in order to provide routing instructions for the call within a configurable time period or default routing applies. |
| RouteRequestFailed | This event is sent to indicate that the RouteRequest was not processed to completion. |
| RouteUsed | This event is sent to the client after a Route Select Request is received and processed to indicate the final result of the Route Select. |
| RouteRejected | This event is sent in response to a previous Route Select request to indicate that the request was not successful. |
| ServiceStateChange | This event is sent in order to indicate a service state change. |
| LinkStatus | This event is sent to indicate the health of the system. |
| SnapShotUpdate | This event is generated after a Snapshot request is made and provides the application with a snapshot of the state of the system regarding agents, calls, and/or services. |
| SnapShotRequestFailed | Occurs when the SnapShot request fails. |

# Appendix D
# Sample Code

This appendix displays a sample of client code:

## Sample client code

```
public class ClientApplication
{
   public AgentCallServiceClient CTIPSClientProxy;

   public void Initialize()
   {
      CTIPSClientProxy  = new AgentCallServiceClient();

      OpenConnectionArgs ocArgs = new OpenConnectionArgs();

      ocArgs.ClientCallbackURL =
"http://10.128.98.146:8730/www.aspect.com/PublishEvents";
      ocArgs.TenantIds = new int[1];
      ocArgs.TenantIds[0] = 1;
      ocArgs.UserCredentials = new UserCredentials();
      ocArgs.UserCredentials.UserName = "Test01";
      ocArgs.UserCredentials.Password = "Test01";

      SessionObject so = new SessionObject();

      so = CTIPSClientProxy.OpenConnection(ocArgs);

      SessionId = so.SessionId;
      AddMessageClassArgs msgArgs = new AddMessageClassArgs();
      msgArgs.SessionId = SessionId;
      msgArgs.TenantId = 1;
      msgArgs.MessageClass = CTIMessageClass.EventCategoryAgent;
      // subscribe for all agent events
      result = CTIPSClientProxy.AddMessageClass(msgArgs);
```

```
            msgArgs.MessageClass = CTIMessageClass.EventCategoryCall;
            // subscribe for all call events
            result = CTIPSClientProxy.AddMessageClass(msgArgs);

            msgArgs.MessageClass = CTIMessageClass.EventCategorySystem;
           // subscribe for all system events
           result = CTIPSClientProxy.AddMessageClass(msgArgs);

           msgArgs.MessageClass = CTIMessageClass.EventCategoryRoute;
           // subscribe for all route events
           result = CTIPSClientProxy.AddMessageClass(msgArgs);

           OnLineArgs online = new OnLineArgs();
           online.SessionId = SessionId;
           online.TenantId = 1;
           // go online to start receiving events
           result = CTIPSClientProxy.OnLine(online);
        }

    }
```

E

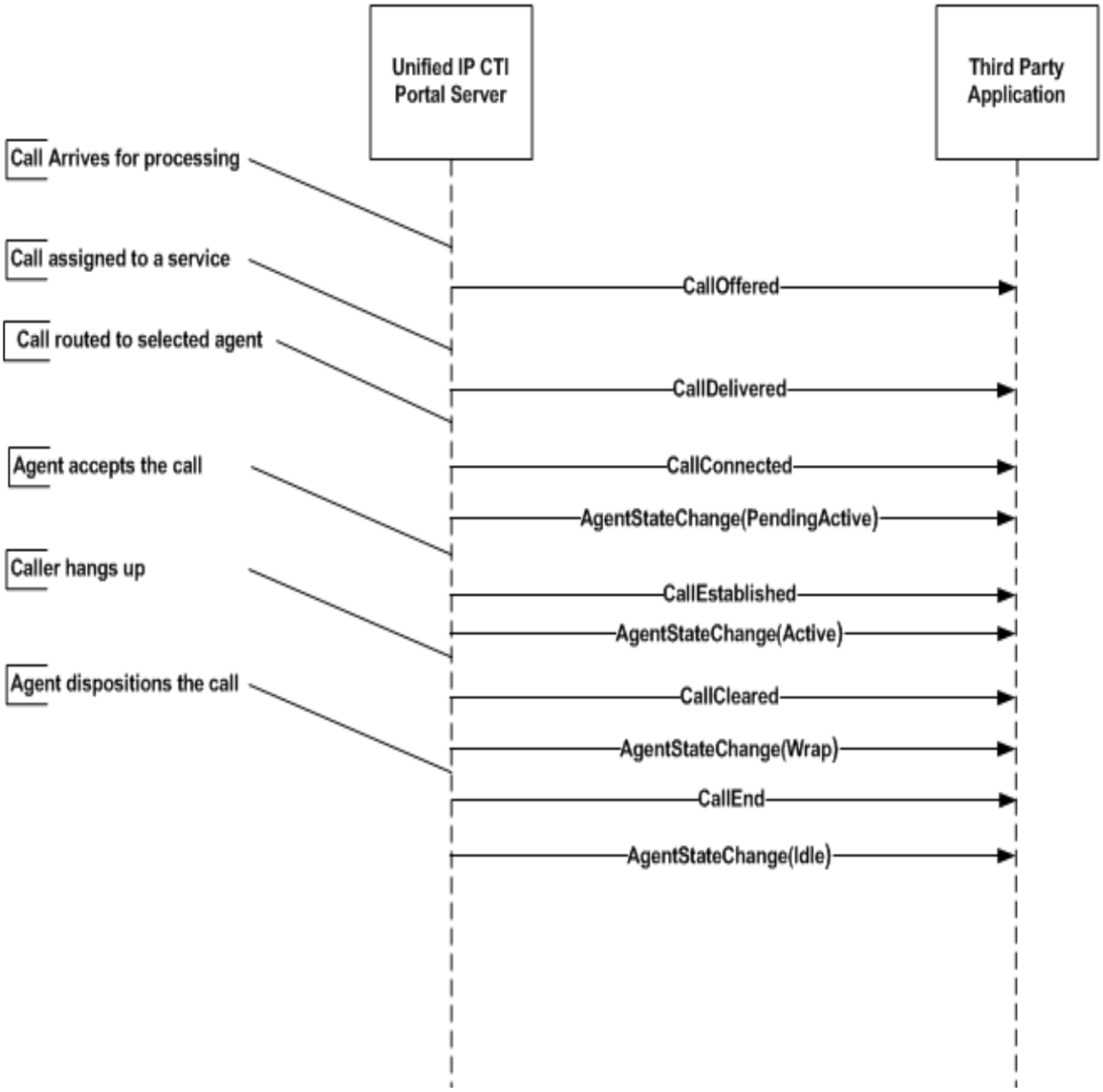
# Appendix E
# Event Flow Diagrams

This appendix contains event flow diagrams of the following:


- Call Arrives to Inbound Service
- Route Request - Route Select to Invalid Destination
- Route Request - Route Select to Valid Service
- Blind Transfer to Agent
- Blind Transfer to Valid Service
- Consult Transfer to Service
- Consult Transfer Agent
- Blind Transfer to Unmanned Service
- 3-Way Conference Service
- Agent to Agent
- 3-Way Conference Agent
- Conference Initiator Hang Up

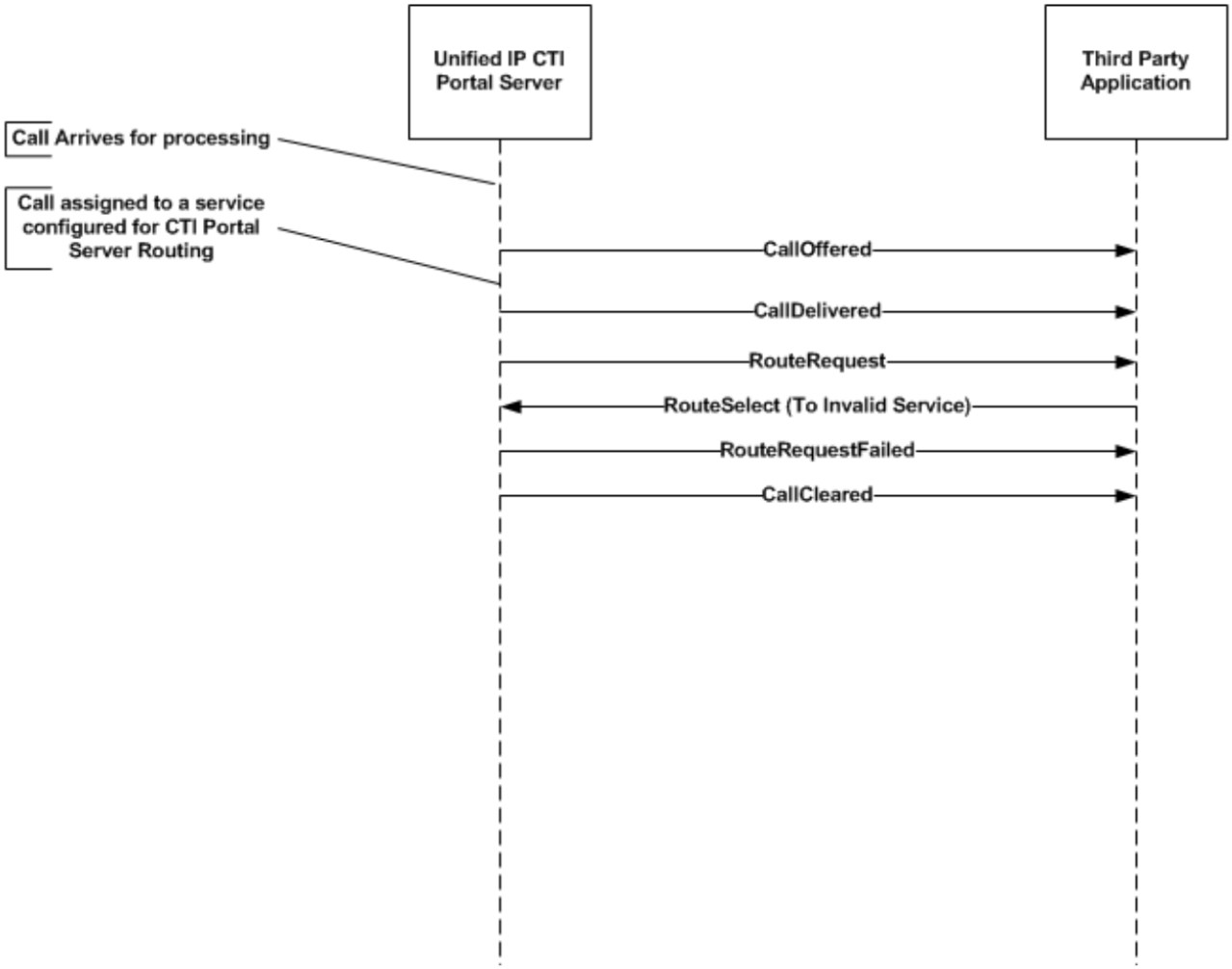
# Call Arrives to Inbound Service

Event flow for a simple Inbound call.



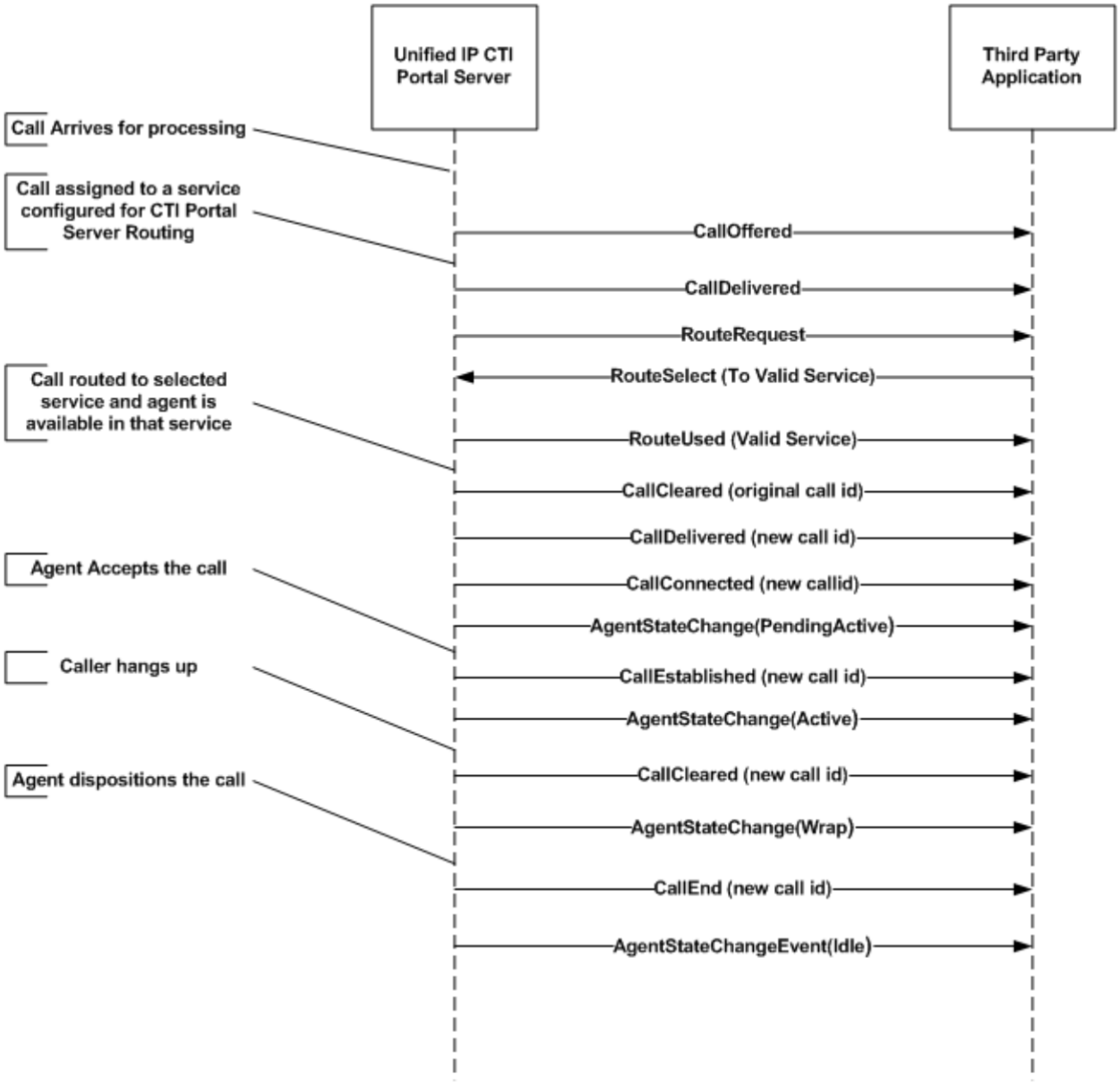
# Route Request - Route Select to Invalid Destination

Inbound call assigned to service configured for External Routing. Call routed to an invalid service.



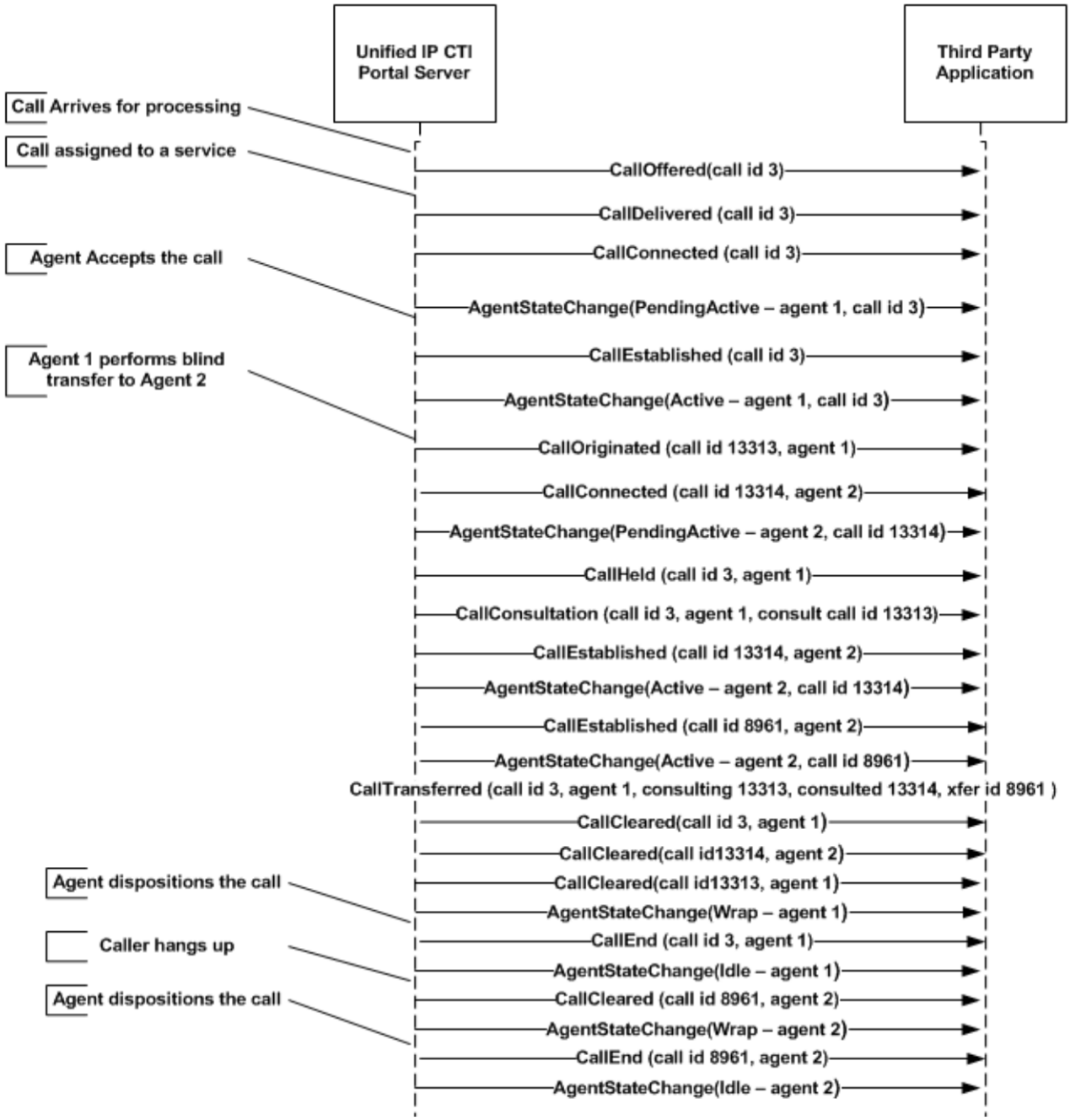
# Route Request - Route Select to Valid Service

Inbound call assigned to service configured for External CTI Routing. Call routed to a valid service.



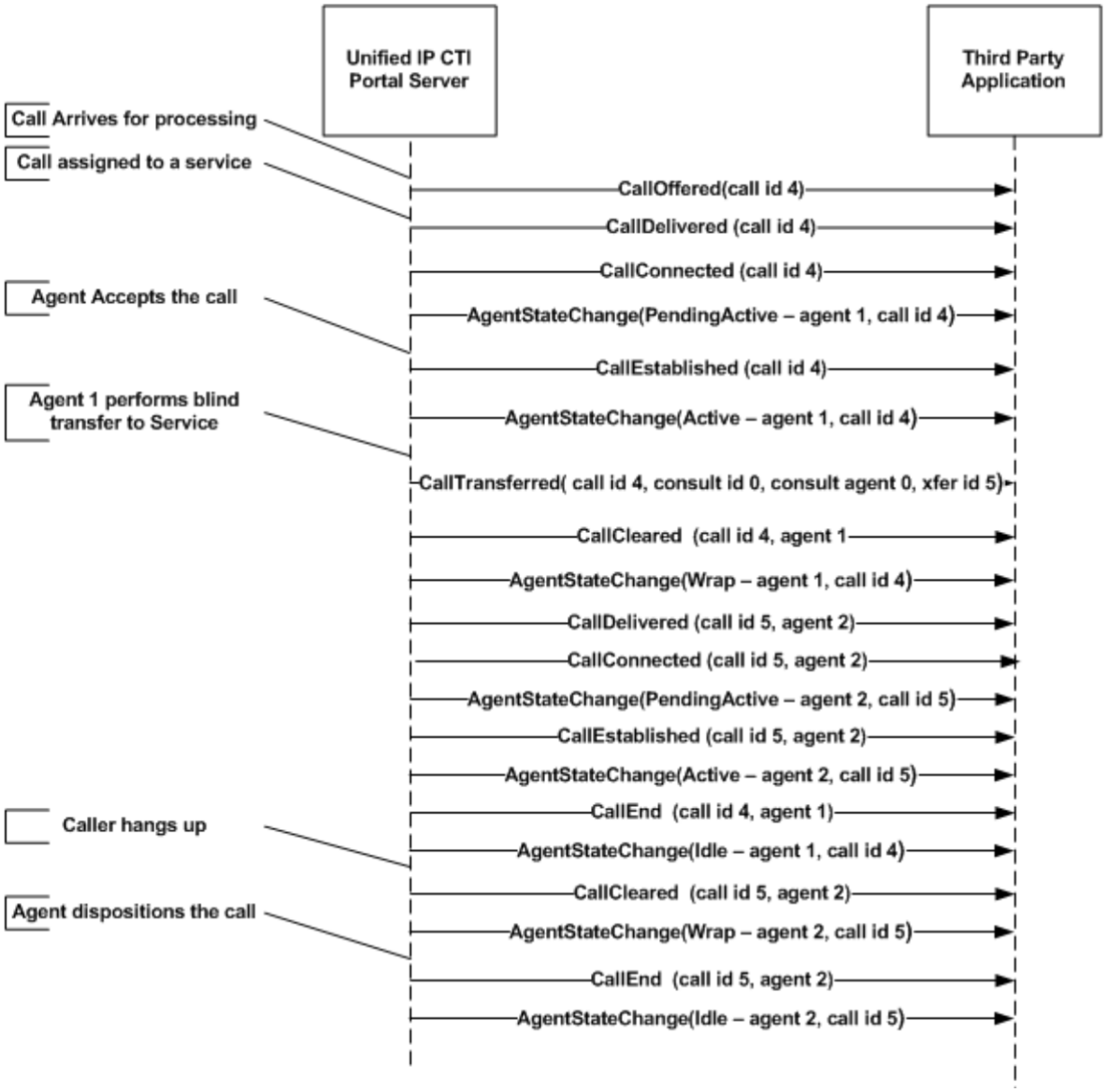
# Blind Transfer to Agent

Inbound call gets routed to an agent. Agent performs blind transfer to another agent.



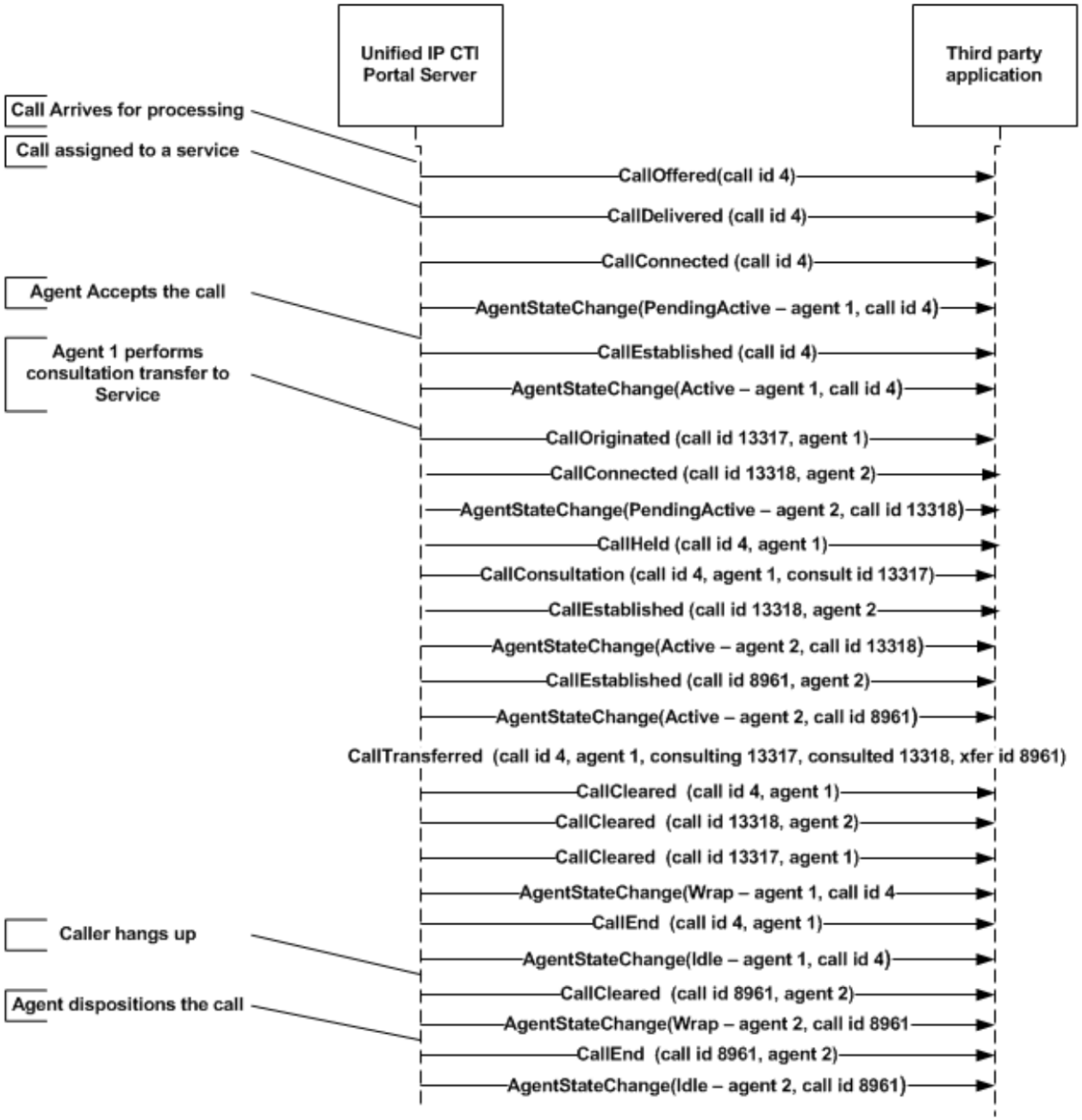
# Blind Transfer to Valid Service

Inbound call gets routed to an agent. Agent performs blind transfer to a valid service.



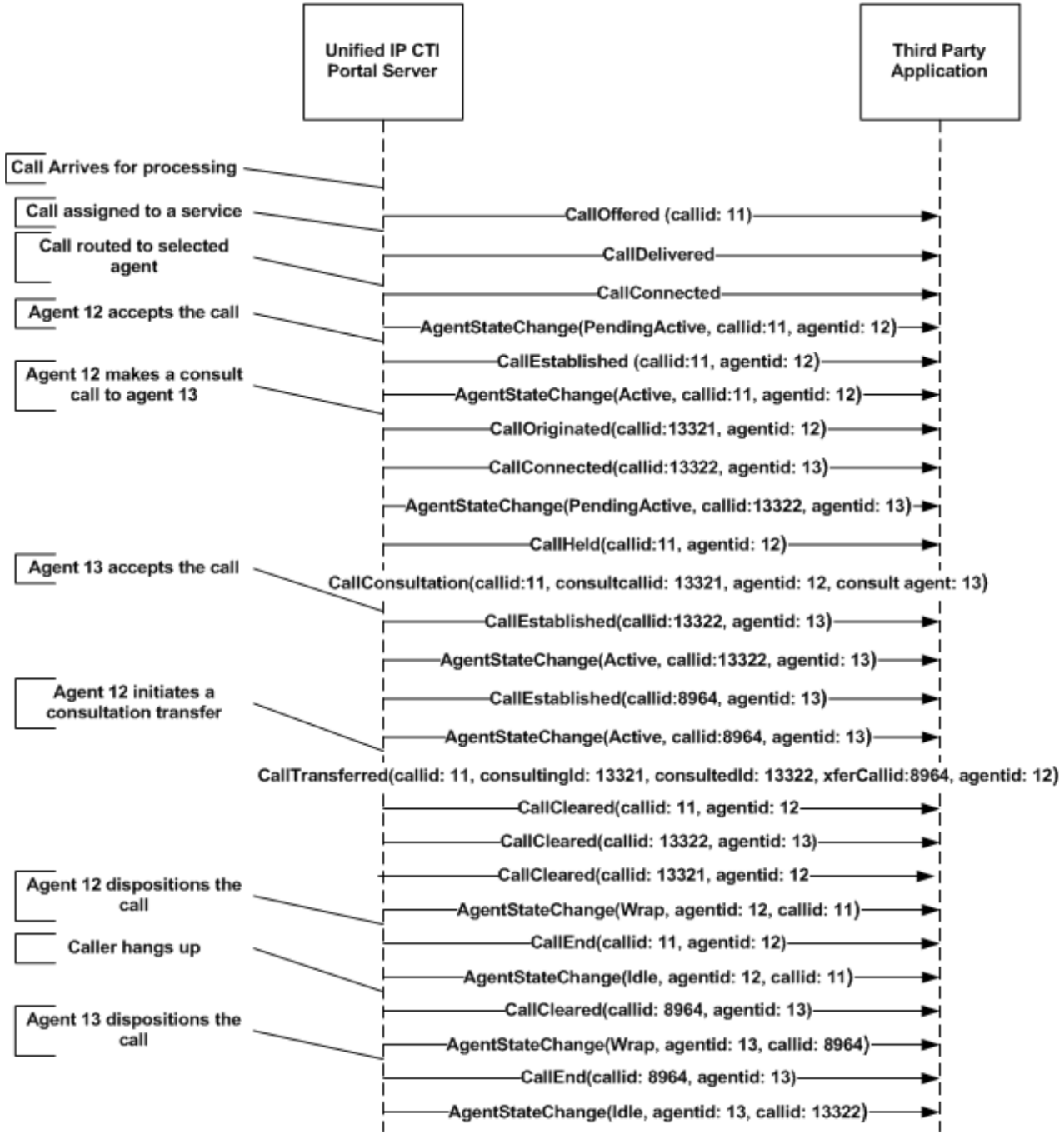
# Consult Transfer to Service

Inbound call gets routed to an agent. Agent performs consultation transfer to a valid service.



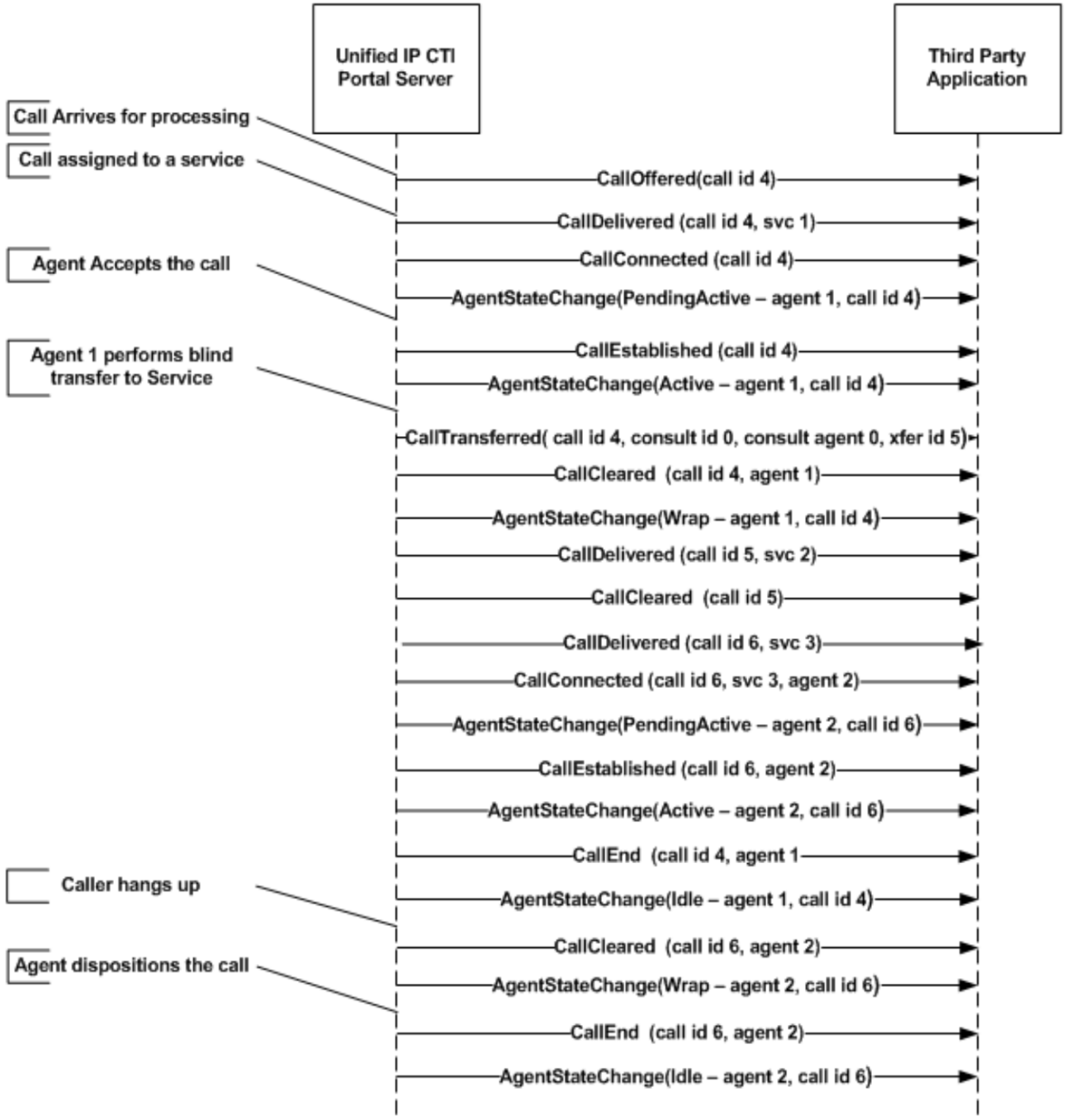
# Consult Transfer Agent

Inbound call gets routed to an agent. Agent performs consultation transfer to a valid agent.



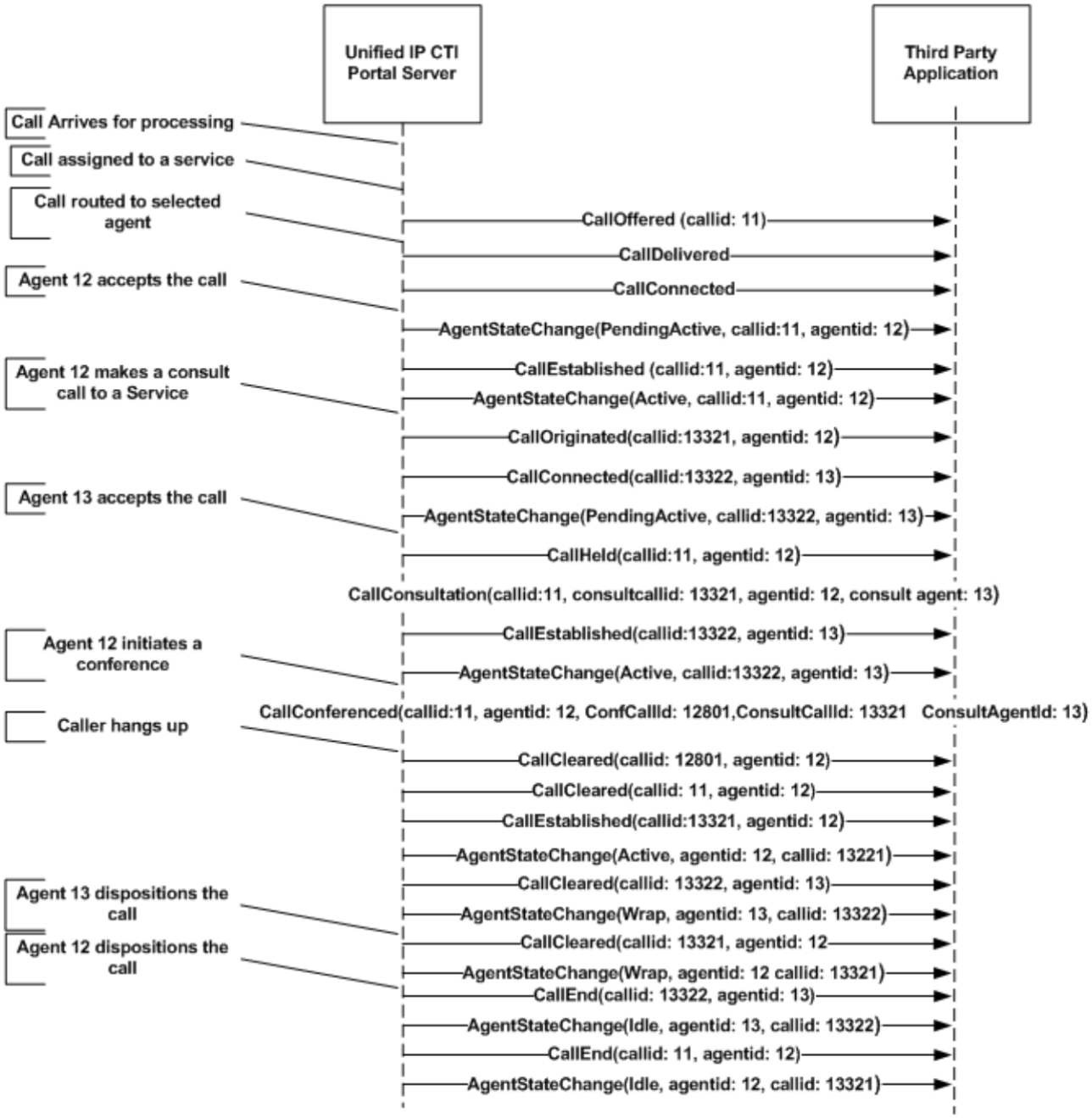
# Blind Transfer to Unmanned Service

Inbound call gets routed to an agent. Agent performs blind transfer to an unmanned service. Reroute logic is executed for unmanned service.



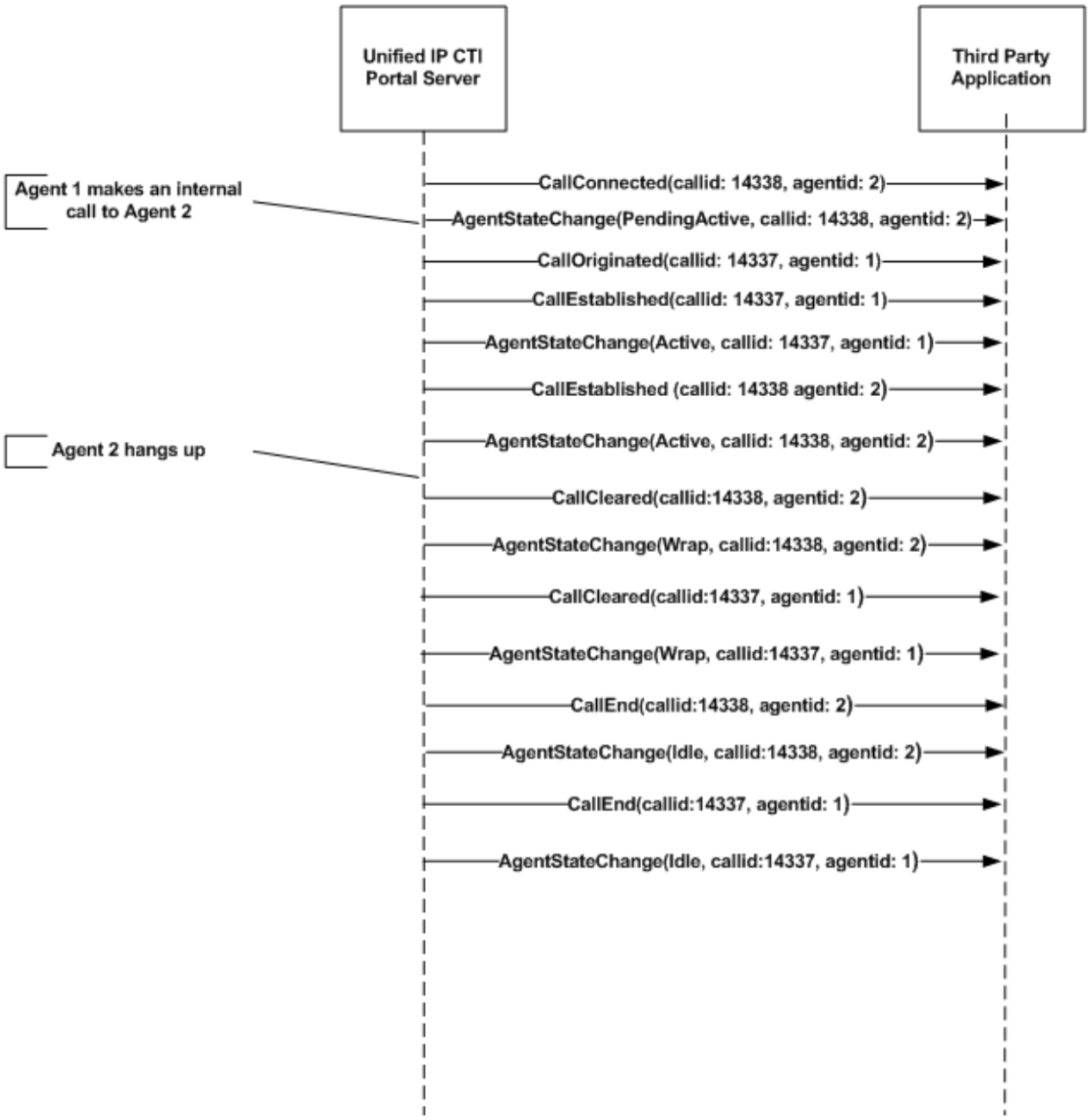
# 3-Way Conference Service

Conference scenario: Inbound call gets routed to an agent. Agent makes a consultation call to a service. Customer hangs up first.



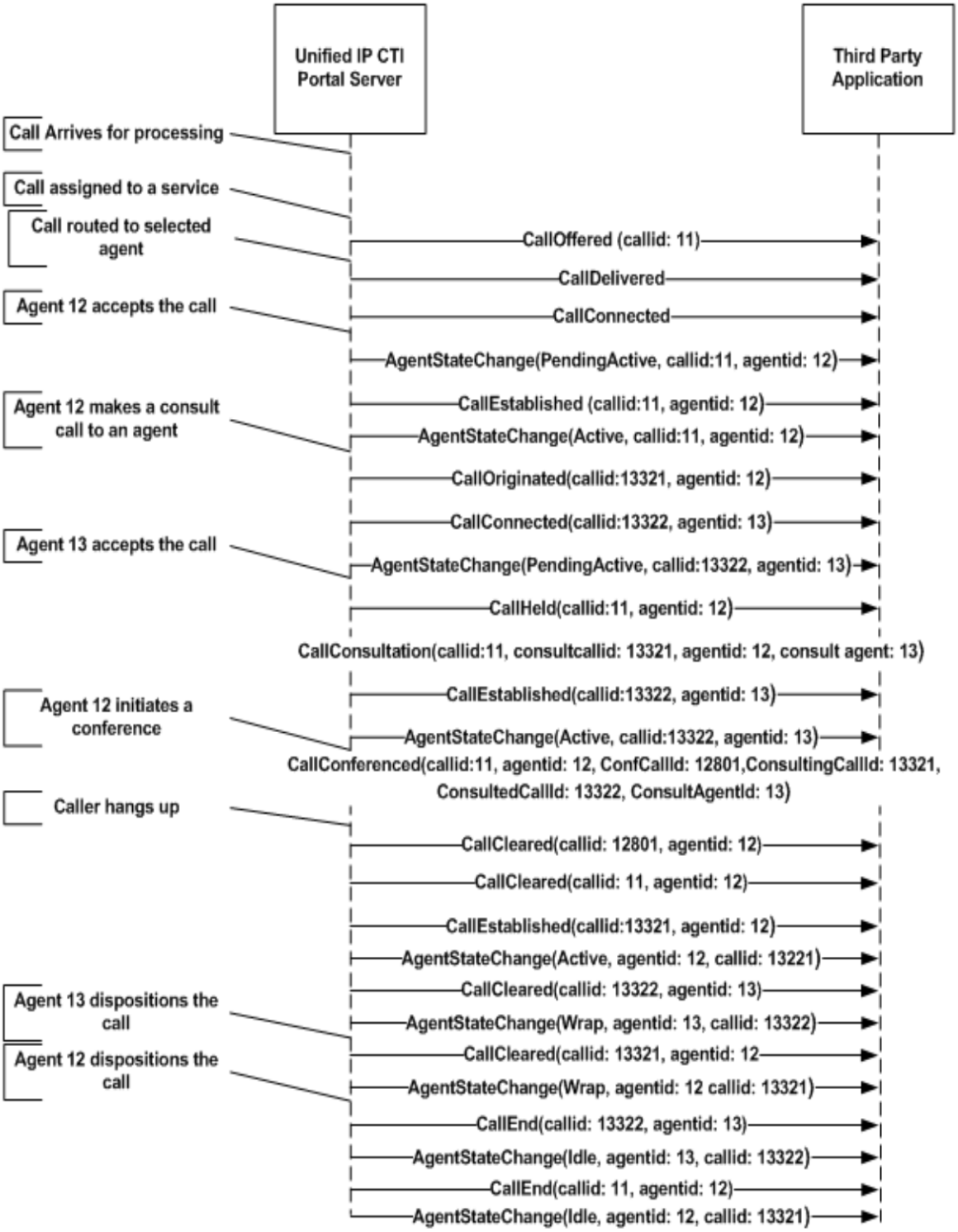
# Agent to Agent

Internal call scenario: Agent dials another agent.



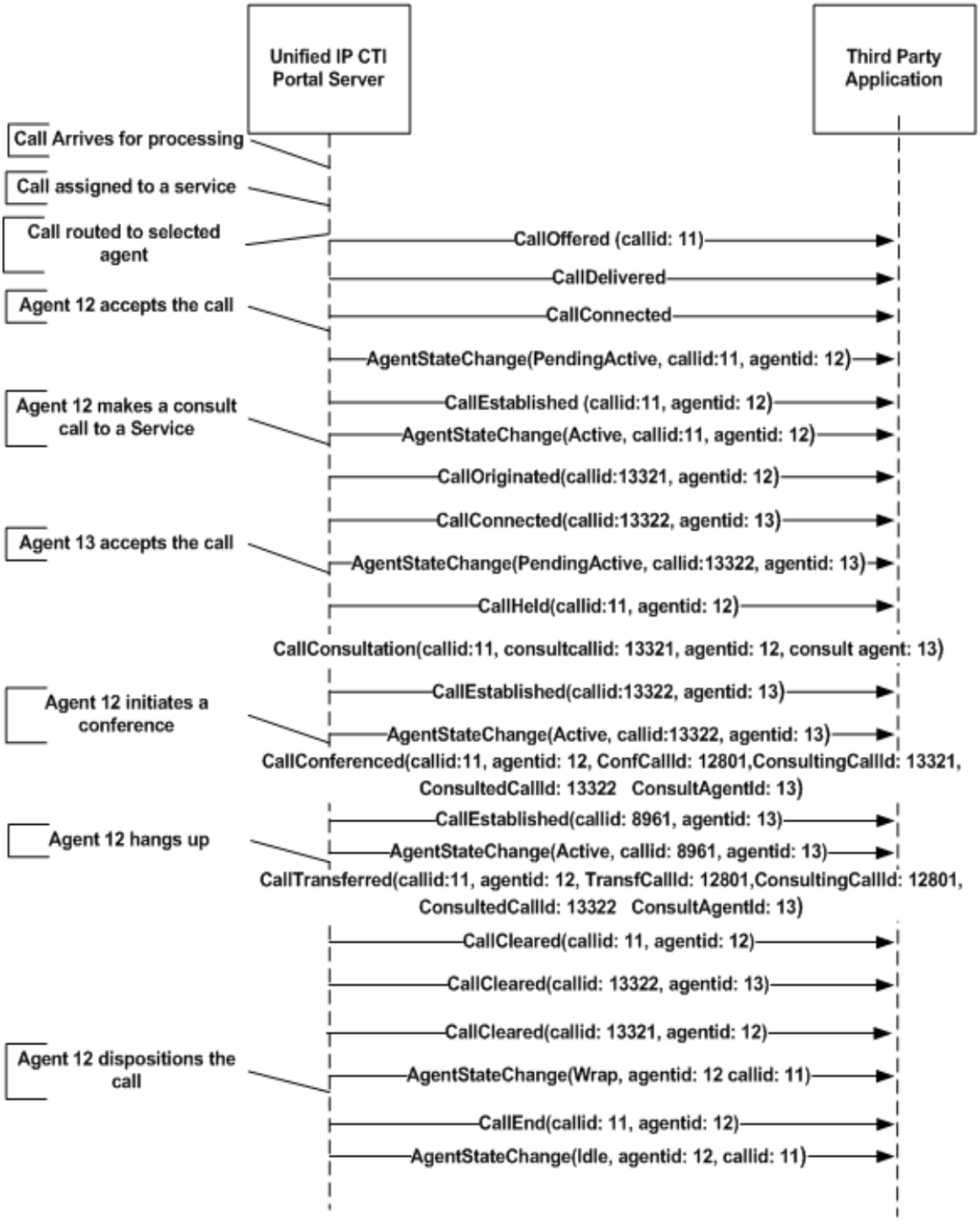
# 3-Way Conference Agent

Conference scenario: Customer hangs up first.



# Conference Initiator Hang Up

Conference scenario. Conference initiator hangs up.



# Index

## A

about this guide, i
Agent Management Requests
  GetAgentState, 3-1
  LoginAgent, 3-1
  LogoutAgent, 3-1
  overview, 3-1
  PanicLogout, 3-1
  SetAgentState, 3-1
  SetDisposition, 3-1
AgentEvents
  AgentLoginRequestFailed, 4-1
  AgentLoginStateChange, 4-1
  AgentRequestFailed, 4-1
  AgentStateChange, 4-1
  AudiopathStateChange, 4-1
  EnterPasscode, 4-1
  overview, 4-1
  ServiceStateChange, 4-1
API
  agent management and call control interface, 1-1
  notification interface, 1-1
audience, i

## C

Call Control Request
  ClearCall, 3-1
  ConferenceCall, 3-1
  ConsultationCall, 3-1
  GetCallData, 3-1
  HoldCall, 3-1
  MakeCall, 3-1
  RetrieveCall, 3-1
  SendDTMFDigits, 3-1
  SetCallData, 3-1
  TransferCall, 3-1
Call Control Requests, 3-1
  AcceptCall, 3-1
Call Route Events
  overview, 4-1
  RouteRejected, 4-1
  RouteRequest, 4-1
  RouteRequestFailed, 4-1
  RouteUsed, 4-1
Call Routing Request
  overview, 3-1
  RouteSelect, 3-1
CallEvents
  CallClearedEvent, 4-1
  CallCompleted, 4-1
  CallConferenced, 4-1
  CallConnected, 4-1
  CallConsultation, 4-1
  CallDataUpdate, 4-1
  CallDelivered, 4-1
  CallEnd, 4-1
  CallEstablished, 4-1
  CallHeld, 4-1
  CallOffered, 4-1
  CallOriginated, 4-1
  CallRequestFailed, 4-1
  CallRetrieved, 4-1
  CallStarted, 4-1
  CallTransferred, 4-1
  DigitsDialed, 4-1
  overview, 4-1
Client application
  startup and shutdown sequence, 5-1
client connections, 1-1
configuration and install, 1-1
configuring CTIPS
  adding a new portal server, B-1
  deleting an existing portal server, B-1
  modifying an existing portal server, B-1
  with Server Configurator, B-1
CTIPS
  client connections, 1-1
  failover, 1-1
  high availability, 1-1
  implementation, 1-1
  introduction, 1-1
  overview, 1-1
  security, 1-1

# D

data types
CTIAgentCapabilities, A-1
CTIAgentLoginState, A-1
CTIAgentState, A-1
CTIAudioPathState, A-1
CTICallDataItem, A-1
CTICallDataUpdateType, A-1
CTICallInfo, A-1
CTICallState, A-1
CTICallType, A-1
CTIClearingPartyType, A-1
CTIComponentState, A-1
CTIComponentType, A-1
CTIDestinationType, A-1
CTIEntityType, A-1
CTIEventCategory, A-1
CTIEventType, A-1
CTIFailureType, A-1
CTIFieldType, A-1
CTIMediaType, A-1
CTIMessageClass, A-1
CTIRejectReason, A-1
CTIRouteStatusType, A-1
CTIRouteType, A-1
CTIServiceInfo, A-1
CTIServiceState, A-1
CTISnapShotAgentData, A-1
CTISnapShotCallData, A-1
CTISnapShotServiceData, A-1
CTITransferType, A-1
User credentials, A-1

# E

Event Categories
CTIEventCategory, 4-1
Event examples, E-1
3-Way Conference Service, E-1
Agent to Agent, E-1
Blind Transfer to Agent, E-1
Blind Transfer to Unmanned Service, E-1
Blind Transfer to Valid Service, E-1
call arrives to inbound service, E-1
Conference Initiator Hang Up, E-1
Consult Transfer Agent, E-1
Consult Transfer to Service, E-1
Route Request - Route Select to Invalid Destination, E-1
Route Request - Route Select to Valid Service, E-1
Event Examples
3-Way Conferencing Agent, E-1
Event Messages
overview, 2-5
Event Type
EventMessage Type, 4-1
Events
description, 4-1
event header, 4-1
examples, E-1
inbound telephone call, 2-1
Int HeartbeatMessage(), 4-1
Int StartSession, 4-1
messages overview, 4-1
overview, 4-1
registering an event, 5-1
registration, 5-1
RequestId, 4-1
voidPublishCTIEvent(CTIEvent), 4-1
events, requests, and responses
overview, 2-1

# F

failover, 1-1
filters
adding to a session, 5-1

# G

GetAgentState, 3-1

# H

high availability, 1-1

# I

implementation, 1-1
inbound telephone call events, 2-1
installation and configuration, 1-1

# L

LoginAgent request, 3-1
LogoutAgent, 3-1

# O

organization of this guide, i

# P

PanicLogout, 3-1

# R

registering
for agent events, 5-1
for call events, 5-1
for routing events, 5-1

for specific agent events, 5-1
for specific service Ids, 5-1
registration
how to register for events, 5-1
request
AddFilter, 3-1
AddMessageClass, 3-1
Call Routing, 3-1
CloseConnection, 3-1
DeleteFilter, 3-1
DeleteMessageClass, 3-1
Offline, 3-1
Online, 3-1
OpenConnection, 3-1
request identifier, agent identifier, and all identifiers, 3-1
Snapshot, 3-1
requests and responses
overview, 2-2
request
agent, 2-4
call control, 2-4
request messages, 2-2
request session and system, 2-2
route, 2-4
response
overview, 2-4

## S

security, 1-1
Server Configurator
configuring CTIPS, B-1
session
adding filters, 5-1
closing with CTIPS, 5-1
identifier, 3-1
opening with CTIPS, 5-1
placing offline, 5-1
placing online, 5-1
registering for events, 5-1
requesting a snapshot, 5-1
session and system requests, 3-1
SetAgentState, 3-1
SetDisposition, 3-1
Snapshot
request, 3-1
requesting, 5-1
SnapShotUpdate Event, 4-1
summary descriptions, C-1, D-1
call control events, C-1
call control responses, C-1
call control services, C-1
SystemEvents
LinkStatus, 4-1
overview, 4-1
SnapshotUpdate, 4-1

## T

tenant identifier, 3-1