Introduction

This chapter provides an introduction to Computer Telephony Integration (CTI) and describes how CTI can enhance the value of contact center applications. This chapter also introduces the Computer Telephony Integration Object Server (CTI OS) product and discusses the advantages of using CTI OS to develop custom CTI enabled applications.

- Introduction to CTI, page 1
- CTI-Enabled Applications, page 1
- Events and Requests Within CTI Environment, page 2
- Overview of CTI OS, page 4

Introduction to CTI

The workflow of a modern contact center is based on two main areas: the media for communicating with the customer and the platform for servicing customer requests.

CTI is the integration of the communications media (phone, email, or web) with the customer service platform (customer databases, transaction processing systems, or CRM (customer relationship management) software packages).

Integrating communications media with the customer service platform helps agents service customers better and faster in the following two ways:

- It enables the agent to leverage the information and events provided by the media to direct workflow.
- It increases the depth and breadth of customer information presented to the agent when the customer's contact arrives at the workstation.

CTI-Enabled Applications

A CTI-enabled application is one in which the software an agent uses to service a customer request is driven by information generated by the presentation of the customer contact.

Screen Pop

The most common CTI application is a screen pop. In a screen pop, the customer service platform is provided with customer information at the arrival of a phone call and begins processing the customer's transaction at the same time as the communication begins between the customer and the agent. This transfer of customer information is called the call context information: a rich set of customer-specific data that travels with the call throughout the enterprise.

For example, a phone call can trigger a screen pop application for a cellular telephone company. It uses the customer ANI (automated number identification, or calling line ID) to do a database look up to retrieve the customer's account information and displays this customer record for the agent. By the time the agent can say “Thank you for calling ABC Telephony Company,” the account record is on the screen and the agent is ready to service the customer's request.

Agent State Control

Similar to a screen pop, CTI application control of agent state is a way to improve the agent's workflow by integrating the service delivery platform with the communications media. A CTI application enabled for agent state can set the agent's current work state according to the type of work being performed.

For example, a sales application can automatically send an agent to a wrap-up or after-call work state when the customer contact terminates. The agent can then enter wrap up data about that transaction or customer inquiry and (subject to a timer) change the state automatically back to available when the wrap up work is complete.

Third-Party Call Control

The most advanced CTI integration projects seek a total integration of the customer service platform with the communications media. In third-party call control applications, the actual control over the teleset or other media is initiated via the software application and coordinated with application screens or views.

For example, a financial services application can transfer a phone call to a speed-dial number designated by the application itself. In this scenario, the agent can click a button to determine the appropriate destination for the transfer, save the application's customer context, and transfer the call to the other agent.

Events and Requests Within CTI Environment

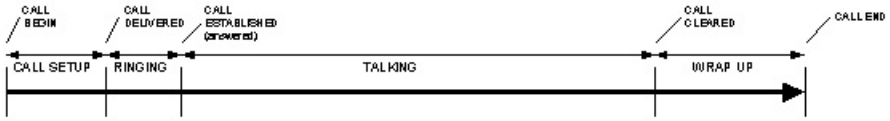
The first step to developing a CTI-enabled application is to understand the events and requests that are at play within the CTI environment. Asynchronous events are messages sent to applications that indicate an event to which the application can respond (for example, CallBeginEvent). Requests are the mechanism that the application uses to request that a desired behavior happen (for example, TransferCall).

Asynchronous Events

The CTI environment is one of diverse servers and applications communicating over a network. This naturally leads to asynchronous, or unsolicited events – events that arrive based on some stimulus external to the user's application. The main source of events in the CTI environment is the communications media.

The following figure depicts the stages of a typical inbound telephone call and its associated events.

Figure 1: Typical Inbound Call Events Flow



The following events are generated, based on the state of the call:

- OnCallBegin: Indicates that the call has entered the setup phase.
- OnCallDelivered: Generated when the call starts ringing.
- OnCallEstablished: Generated when the call is answered.
- OnCallCleared: Generated when the voice connection is terminated (e.g. call hung up).
- OnCallEnd: Generated when the logical call appearance (including call data) is complete.

In addition to the events and states shown in the figure above, the following are typical call events that CTI applications use:

- OnCallHeld: Generated when the call transitions from the active to held state.
- OnCallRetrieved: Generated when the call is removed from hold.
- OnCallTransferred: Indicates that the call has been transferred to another party.
- OnCallConferenced: Indicates that a new party has been added to the call.

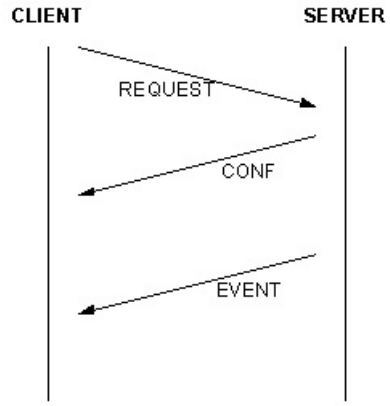
The foregoing is only a brief sample of the events available via CTI OS. The complete set of events available for CTI developers is detailed in later chapters in this guide.

Request-Response Paradigm

In addition to responding to asynchronous events, a CTI enabled application can make programmatic requests for services via the CTI interface. Specifically, the CTI application uses the request-response mechanism to perform agent state and third-party call control, and to set call context data.

The typical request-response flow for CTI uses the model shown in the following figure:

Figure 2: Sample Request-Response Message Flow

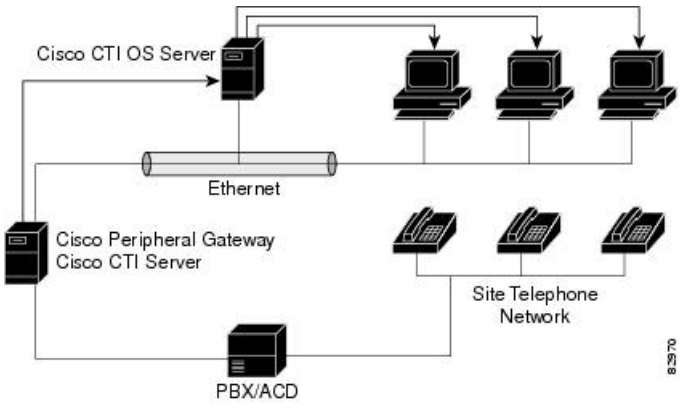


A request generated by the CTI-enabled application (CLIENT) is sent to the CTI service (SERVER), and a response message (CONF) is generated to indicate that the request was received. In most cases if the request is successful, a follow-on event is received indicating that the desired behavior has occurred. Detailed descriptions of this kind of request-response-event message flow are detailed in later chapters in this guide.

Overview of CTI OS

The Computer Telephony Integration Object Server (CTI OS) is Cisco's next generation customer contact integration platform. CTI OS combines a powerful, feature-rich server and an object-oriented software development toolkit to enable rapid development and deployment of complex CTI applications. Together with the Cisco CTI Server Interface, CTI OS and Client Interface Library (CIL) creates a high performance, scalable, fault-tolerant three-tiered CTI architecture, as illustrated in the figure below.

Figure 3: CTI OS Three-Tiered Architecture Topology



The CTI OS application architecture employs three tiers:

- The CIL is the first tier and provides an application-level interface to developers.

- The CTI OS Server is the second tier and provides the bulk of the event and request processing and enabling the object services of the CTI OS system.
- The Cisco CTI Server is the third tier and provides the event source and the back-end handling of telephony requests.

Advantages of CTI OS as a CTI Development Interface

CTI OS brings several major advances to developing custom CTI integration solutions. The CIL provides an object-oriented and event driven application programming interface (API), while the CTI OS server does all the heavy-lifting of the CTI integration: updating call context information, determining which buttons to enable on softphones, providing easy access to supervisor features, and automatically recovering from failover scenarios.

- **Rapid integration**. Developing CTI applications with CTI OS is significantly easier and faster than any previously available Cisco CTI integration platform. The same object oriented interface is used across programming languages, enabling rapid integrations in .NET, and C++, Visual Basic, or any Microsoft COM compliant container environment. Developers can use CTI OS to create a screen pop application in as little as five minutes. The only custom-development effort required is within the homegrown application to which CTI is being added.
- **Complex solutions made simple**. CTI OS enables complex server-to-server integrations and multiple agent monitoring-type applications. The CIL provides a single object-oriented interface that you can use in two modes: Agent Mode and Monitor Mode. See CTI OS Client Interface Library Architecture for an explanation of these two modes.
- **Fault tolerant**. CTI OS is built upon the Unified ICM NodeManager fault-tolerance platform, which automatically detects process failure and restarts the process, enabling work to continue. Upon recovery from a failure, CTI OS initiates a complete, system-wide snapshot of all agents, calls, and supervisors and propagates updates to all client-side objects.

Key Benefits of CTI OS for CTI Application Developers

The CTI OS Client Interface Library (CIL) provides programmers with the tools to rapidly develop high-quality CTI enabled applications, taking advantage of the rich features of the CTI OS server. Every feature of CTI OS was designed with ease of integration in mind, to remove the traditional barriers to entry for CTI integrations.

- **Object-oriented interactions**. CTI OS provides an object-oriented CTI interface by defining objects for all call center interactions. Programmers interface directly with Session, Agent, SkillGroup, and Call objects to perform all functions. CIL objects are thin proxies for the server-side objects, where all the heavy-lifting is done. The Session object manages all objects within the CIL. A UniqueObjectID identifies each object. Programmers can access an object by its UniqueObjectID or by iterating through the object collections.
- **Connection and session management**. The CTI OS CIL provides out-of-the-box connection and session management with the CTI OS Server, hiding all of the details of the TCP/IP sockets connection. The CIL also provides an out-of-the-box failover recovery: upon recovery from a failure, the CIL automatically reconnects to another CTI OS (or reconnects to the same CTI OS after restart), re-establishes the session, and recovers all objects for that session.

- **All parameters are key-value pairs**. The CTI OS CIL provides helper classes treat all event and request parameters as simply a set of key-value pairs. All properties on the CTI OS objects are accessible by name via a simple Value = GetValue (“key”) mechanism. Client programmers can add values of any type to the CTI OS Arguments structure, using the enumerated CTI OS keywords, or their own string keywords (for example, AddItem[“DialedNumber”, “1234”]). This provides for future enhancement of the interface without requiring any changes to the method signatures.
- **Simple event subscription model**. The CTI OS CIL implements a publisher-subscriber design pattern to enable easy subscription to event interfaces. Programmers can subscribe to the appropriate event interface that suits their needs, or use the AllInOne interface to subscribe for all events. C++ and COM contain subclassable event adapter classes. These classes enable programmers to subscribe to event interfaces; they add only minimal custom code for the events they use and no code at all for events they do not use.

Illustrative Code Fragments

Throughout this manual, illustrative code fragments are provided both to clarify usage and as examples. These fragments are written in several languages, including . Though .NET (and therefore VB .NET) is supported, note that the VB code fragments are written using VB 6 syntax.