CTI Driver for Siebel 7 Reference Guide for Cisco ICM/IPCC Enterprise & Hosted Editions

Cisco CTI OS Release 7.2(1)
May 2007

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

Customer Order Number:

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

CTI Driver for Siebel 7 Reference Guide for Cisco ICM/IPCC Enterprise & Hosted Editions, CTI OS Release 7.2(1)

CONTENTS

About This Guide

Purpose

This manual discusses the specifics regarding integration of Cisco Intelligent Contact Management (ICM) software with the Siebel eBusiness applications. It also describes how to install, test, demonstrate, and administrate the Cisco Driver for Siebel 7, and discusses the product architecture.

Audience

This manual is intended for Cisco, Siebel, and partner system administrators and integrators.

Organization

The following table describes the information contained in each chapter of this guide.

Chapter	Description
Chapter 1, "Introduction"	Provides an overview of the product, including a description of the architecture.
Chapter 2, "Installation"	Provides instructions for installing the Cisco Driver for Siebel 7 and installing the Cisco Data Store.
Chapter 3, "Customization"	Discusses how to tailor and use the elements provided.
Chapter 4, "Broadcast Statistics"	Provides information about broadcast statistics functionality.
Chapter 5, "Troubleshooting and Testing"	Discusses problems that might be encountered and how to deal with them.
Appendix A, "Business Service Script"	Displays a complete business services script that contains available functions. It also provides a sample DEF file that displays the broadcast statistics configuration.
Appendix B, "Configuring Agent State Toggling"	Displays a complete sample DEF file that displays the ChangeBusyState command configuration for toggling the NotReady button between the Ready/NotReady agent states.

Conventions

This manual uses the following conventions:

Format	Example
Boldface type is used for user entries, keys, buttons, and folder and submenu names.	Choose **Script > Call Type Manager**.
Italic type indicates one of the following: • A newly introduced term • For emphasis • A generic syntax item that you must replace with a specific value • A title of a publication	• A *skill group* is a collection of agents who share similar skills. • *Do not* use the numerical naming convention that is used in the predefined templates (for example, **persvc01**). • IF (*condition*, *true-value*, *false-value*) • For more information, see the *Cisco ICM Software Database Schema Handbook*.
An arrow (>) indicates an item from a pull-down menu.	The Save command from the File menu is referenced as **File > Save**.

Other Publications

For additional information about Cisco Intelligent Contact Management (ICM) software, see the Cisco web site listing ICM documentation.

Obtaining Documentation

Cisco documentation and additional literature are available on Cisco.com. Cisco also provides several ways to obtain technical assistance and other technical resources. These sections explain how to obtain technical information from Cisco Systems.

Cisco.com

You can access the most current Cisco documentation at this URL:

http://www.cisco.com/techsupport

You can access the Cisco website at this URL:

http://www.cisco.com

You can access international Cisco websites at this URL:

http://www.cisco.com/public/countries_languages.shtml

Product Documentation DVD

Cisco documentation and additional literature are available in the Product Documentation DVD package, which may have shipped with your product. The Product Documentation DVD is updated regularly and may be more current than printed documentation.

The Product Documentation DVD is a comprehensive library of technical product documentation on portable media. The DVD enables you to access multiple versions of hardware and software installation, configuration, and command guides for Cisco products and to view technical documentation in HTML. With the DVD, you have access to the same documentation that is found on the Cisco website without being connected to the Internet. Certain products also have .pdf versions of the documentation available.

The Product Documentation DVD is available as a single unit or as a subscription. Registered Cisco.com users (Cisco direct customers) can order a Product Documentation DVD (product number DOC-DOCDVD=) from the Ordering tool or Cisco Marketplace.

Cisco Ordering tool:

http://www.cisco.com/en/US/partner/ordering/

Cisco Marketplace:

http://www.cisco.com/go/marketplace/

Ordering Documentation

Beginning June 30, 2005, registered Cisco.com users may order Cisco documentation at the Product Documentation Store in the Cisco Marketplace at this URL:

http://www.cisco.com/go/marketplace/

Cisco will continue to support documentation orders using the Ordering tool:

- Registered Cisco.com users (Cisco direct customers) can order documentation from the Ordering tool:

  http://www.cisco.com/en/US/partner/ordering/

- Instructions for ordering documentation using the Ordering tool are at this URL:

  http://www.cisco.com/univercd/cc/td/doc/es_inpck/pdi.htm

- Nonregistered Cisco.com users can order documentation through a local account representative by calling Cisco Systems Corporate Headquarters (California, USA) at 408 526-7208 or, elsewhere in North America, by calling 1 800 553-NETS (6387).

Documentation Feedback

You can rate and provide feedback about Cisco technical documents by completing the online feedback form that appears with the technical documents on Cisco.com.

You can send comments about Cisco documentation to bug-doc@cisco.com.

You can submit comments by using the response card (if present) behind the front cover of your document or by writing to the following address:

Cisco Systems
Attn: Customer Document Ordering
170 West Tasman Drive
San Jose, CA 95134-9883

We appreciate your comments.

Cisco Product Security Overview

Cisco provides a free online Security Vulnerability Policy portal at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.html

From this site, you can perform these tasks:

- Report security vulnerabilities in Cisco products.
- Obtain assistance with security incidents that involve Cisco products.
- Register to receive security information from Cisco.

A current list of security advisories and notices for Cisco products is available at this URL:

http://www.cisco.com/go/psirt

If you prefer to see advisories and notices as they are updated in real time, you can access a Product Security Incident Response Team Really Simple Syndication (PSIRT RSS) feed from this URL:

http://www.cisco.com/en/US/products/products_psirt_rss_feed.html

Reporting Security Problems in Cisco Products

Cisco is committed to delivering secure products. We test our products internally before we release them, and we strive to correct all vulnerabilities quickly. If you think that you might have identified a vulnerability in a Cisco product, contact PSIRT:

- Emergencies—security-alert@cisco.com

  An emergency is either a condition in which a system is under active attack or a condition for which a severe and urgent security vulnerability should be reported. All other conditions are considered nonemergencies.

- Nonemergencies—psirt@cisco.com

In an emergency, you can also reach PSIRT by telephone:

- 1 877 228-7302
- 1 408 525-6532

Tip We encourage you to use Pretty Good Privacy (PGP) or a compatible product to encrypt any sensitive information that you send to Cisco. PSIRT can work from encrypted information that is compatible with PGP versions 2.*x* through 8.*x*.

Never use a revoked or an expired encryption key. The correct public key to use in your correspondence with PSIRT is the one linked in the Contact Summary section of the Security Vulnerability Policy page at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.htm

The link on this page has the current PGP key ID in use.

Obtaining Technical Assistance

Cisco Technical Support provides 24-hour-a-day award-winning technical assistance. The Cisco Technical Support & Documentation website on Cisco.com features extensive online support resources. In addition, if you have a valid Cisco service contract, Cisco Technical Assistance Center (TAC) engineers provide telephone support. If you do not have a valid Cisco service contract, contact your reseller.

Cisco Technical Support & Documentation Website

The Cisco Technical Support & Documentation website provides online documents and tools for troubleshooting and resolving technical issues with Cisco products and technologies. The website is available 24 hours a day, at this URL:

http://www.cisco.com/techsupport

Access to all tools on the Cisco Technical Support & Documentation website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register at this URL:

http://tools.cisco.com/RPF/register/register.do

Note Use the Cisco Product Identification (CPI) tool to locate your product serial number before submitting a web or phone request for service. You can access the CPI tool from the Cisco Technical Support & Documentation website by clicking the **Tools & Resources** link under Documentation & Tools. Choose **Cisco Product Identification Tool** from the Alphabetical Index drop-down list, or click the **Cisco Product Identification Tool** link under Alerts & RMAs. The CPI tool offers three search options: by product ID or model name; by tree view; or for certain products, by copying and pasting **show** command output. Search results show an illustration of your product with the serial number label location highlighted. Locate the serial number label on your product and record the information before placing a service call.

Submitting a Service Request

Using the online TAC Service Request Tool is the fastest way to open S3 and S4 service requests. (S3 and S4 service requests are those in which your network is minimally impaired or for which you require product information.) After you describe your situation, the TAC Service Request Tool provides recommended solutions. If your issue is not resolved using the recommended resources, your service request is assigned to a Cisco engineer. The TAC Service Request Tool is located at this URL:

http://www.cisco.com/techsupport/servicerequest

For S1 or S2 service requests or if you do not have Internet access, contact the Cisco TAC by telephone. (S1 or S2 service requests are those in which your production network is down or severely degraded.) Cisco engineers are assigned immediately to S1 and S2 service requests to help keep your business operations running smoothly.

To open a service request by telephone, use one of the following numbers:

Asia-Pacific: +61 2 8446 7411 (Australia: 1 800 805 227)
EMEA: +32 2 704 55 55
USA: 1 800 553-2447

For a complete list of Cisco TAC contacts, go to this URL:

http://www.cisco.com/techsupport/contacts

Definitions of Service Request Severity

To ensure that all service requests are reported in a standard format, Cisco has established severity definitions.

Severity 1 (S1)—Your network is "down," or there is a critical impact to your business operations. You and Cisco will commit all necessary resources around the clock to resolve the situation.

Severity 2 (S2)—Operation of an existing network is severely degraded, or significant aspects of your business operation are negatively affected by inadequate performance of Cisco products. You and Cisco will commit full-time resources during normal business hours to resolve the situation.

Severity 3 (S3)—Operational performance of your network is impaired, but most business operations remain functional. You and Cisco will commit resources during normal business hours to restore service to satisfactory levels.

Severity 4 (S4)—You require information or assistance with Cisco product capabilities, installation, or configuration. There is little or no effect on your business operations.

Obtaining Additional Publications and Information

Information about Cisco products, technologies, and network solutions is available from various online and printed sources.

- Cisco Marketplace provides a variety of Cisco books, reference guides, documentation, and logo merchandise. Visit Cisco Marketplace, the company store, at this URL:

  http://www.cisco.com/go/marketplace/

- *Cisco Press* publishes a wide range of general networking, training and certification titles. Both new and experienced users will benefit from these publications. For current Cisco Press titles and other information, go to Cisco Press at this URL:

  http://www.ciscopress.com

- *Packet* magazine is the Cisco Systems technical user magazine for maximizing Internet and networking investments. Each quarter, Packet delivers coverage of the latest industry trends, technology breakthroughs, and Cisco products and solutions, as well as network deployment and troubleshooting tips, configuration examples, customer case studies, certification and training information, and links to scores of in-depth online resources. You can access Packet magazine at this URL:

  http://www.cisco.com/packet

- *iQ Magazine* is the quarterly publication from Cisco Systems designed to help growing companies learn how they can use technology to increase revenue, streamline their business, and expand services. The publication identifies the challenges facing these companies and the technologies to help solve them, using real-world case studies and business strategies to help readers make sound technology investment decisions. You can access iQ Magazine at this URL:

  http://www.cisco.com/go/iqmagazine

  or view the digital edition at this URL:

  http://ciscoiq.texterity.com/ciscoiq/sample/

- *Internet Protocol Journal* is a quarterly journal published by Cisco Systems for engineering professionals involved in designing, developing, and operating public and private internets and intranets. You can access the Internet Protocol Journal at this URL:

  http://www.cisco.com/ipj

- Networking products offered by Cisco Systems, as well as customer support services, can be obtained at this URL:

  http://www.cisco.com/en/US/products/index.html

- Networking Professionals Connection is an interactive website for networking professionals to share questions, suggestions, and information about networking products and technologies with Cisco experts and other networking professionals. Join a discussion at this URL:

  http://www.cisco.com/discuss/networking

- World-class networking training is available from Cisco. You can view current offerings at this URL:

  http://www.cisco.com/en/US/learning/index.html

CHAPTER 1

Introduction

The Cisco Driver for Siebel 7 interfaces the Siebel Enterprise Relationship Management application with Cisco's CTI products. It is designed to process work items, defined as customer interactions, like the telephone, email, and other interaction media to the agent's desktop using the Siebel application.

The Cisco Driver for Siebel 7:

- Provides seamless integration between Cisco and Siebel products.
- Supports Siebel toolbar, commands, and Siebel VB, escript. Agents can place, receive, conference, and transfer calls, including full context call transfer.
- Supports Siebel 7 and greater releases which use the Siebel Communications server, including the Siebel Web client and mobile Web client.
- Passes data to the Siebel application for screen pop and Siebel VB, including ANI, DNIS, caller entered digits, call type, call placement, call variables, including expanded call context (ECC) variables.
- Enables Siebel developers to use Cisco commands and events to develop and enhance their applications.
- Provides integration with Cisco's Data Store.
- Provides agent and skill group statistics from a CTI OS Server with a broadcast display in Siebel.

The Cisco Driver for Siebel 7 requires Siebel 7 or greater. Throughout this document, the terms *driver* or *Cisco driver* are used interchangeably when referring to the Cisco Driver for Siebel 7.

Introduction to Cisco ICM/CTI OS

Cisco's Intelligent Contact Management (ICM) software is central to Cisco's overall call center routing solution. The Cisco CTI OS Server is the heart of Cisco's CTI solutions. Together, ICM software and CTI OS provide agent state and call event information to client applications from TDM or IP ACD, PBX, or IVR switches.

When a customer places an inbound toll-free call, the PSTN queries the customer's Cisco ICM software for the routing destination or label. As the network is delivering the call, a pre-route notification is simultaneously passed to the destination site. A destination site can use this event to gather appropriate database information. (The database can include the Siebel database.)

The information gathered is used in conjunction with routing scripts, which ultimately determine the best skill group to handle the call based on a large amount of available information. The information collected during call queuing is known as *call data* and is passed with the call to an agent using the Siebel application.

A Siebel Certified Engineer uses the Siebel application to map how call data appears in the appropriate Siebel screen pop. Additional information on how to customize the Siebel application is provided in Chapter 3, "Customization."

Introduction to Siebel 7 Architecture

Siebel 7 supports multiple client interfaces, ranging from desktop-installed applications to thin browser-based and wireless interfaces. Each interface has its own strengths and advantages, ranging from high-performance response to ease of installation. The same user is not limited to a single interface connection to Siebel and might decide to connect from different systems with different interfaces at different times.

The exact performance of any given interface depends on multiple factors, from the bandwidth and quality of the connection to the load on the Siebel system.

Figure 1-1 identifies the basic components diagram of the Siebel 7 architecture. The diagram shows two possible client connections: the Web client and a connected Web client.

- The thin client uses a browser Java/JavaScript interface to connect to a web server (IIS or Apache).
- The web server connects to the Siebel Object Manager (SOM), relaying both requests and state to and from the browser.

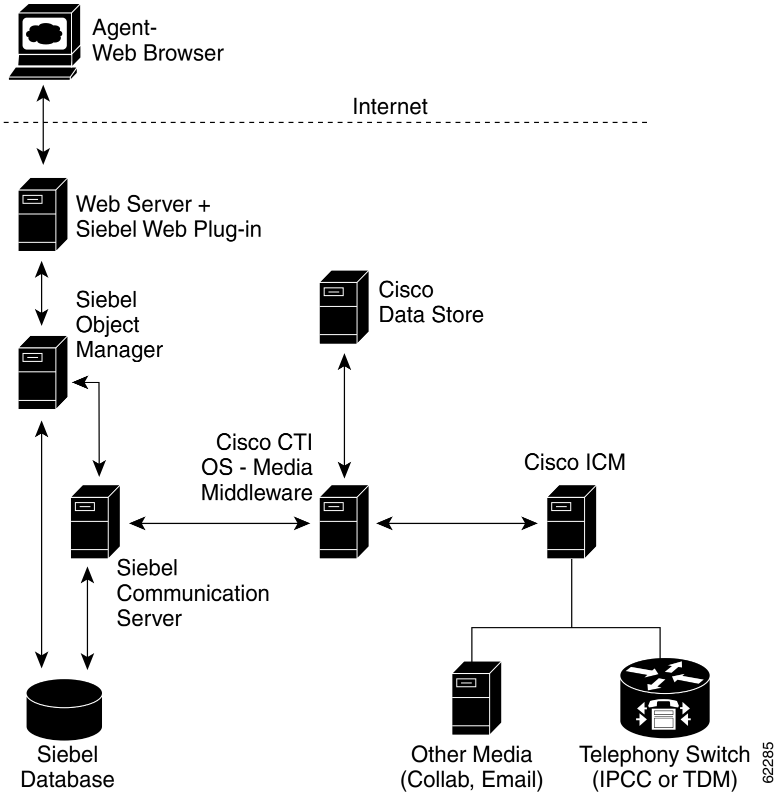
Siebel 7 components include:

- **Siebel Database**. There is only one single database for an enterprise, regardless of the number of sites for an enterprise. (Refer to the Siebel documentation for supported databases.)
- **Siebel Object Manager**. Supports business rules and state information for a client session. The SOM is essentially the Siebel application. It presents and validates forms and manages state and navigation across a session.
- **Siebel Communication Server**. Provides an infrastructure to support several kinds of communications activities for Siebel application users. The Cisco Driver for Siebel 7 uses an Adaptive Communications API. The Siebel Communication Server uses a session-based/interactive communications component, which supports multichannel interactive communications for call or contact center agents who use the communication toolbar to make or receive voice calls through computer telephony integration (supported by Cisco drivers) and other features (Cisco does not provides drivers for these features).

Note Refer to the Siebel documentation for detailed information about the Siebel Communication Server.

Cisco CTI OS connects to the Siebel Communication Server to enable multiple media events, including telephone events.

Figure 1-1 Architectural Block Diagram of Siebel 7



Siebel 7 Multiple Site Architecture

Siebel 7 supports two models for multiple sites. Sites either share a single Siebel database, as shown in Figure 1-2, or they maintain their own database and might use database replication to share data. The choice of database design among multiple sites depends on customer requirements and involves trade-offs.

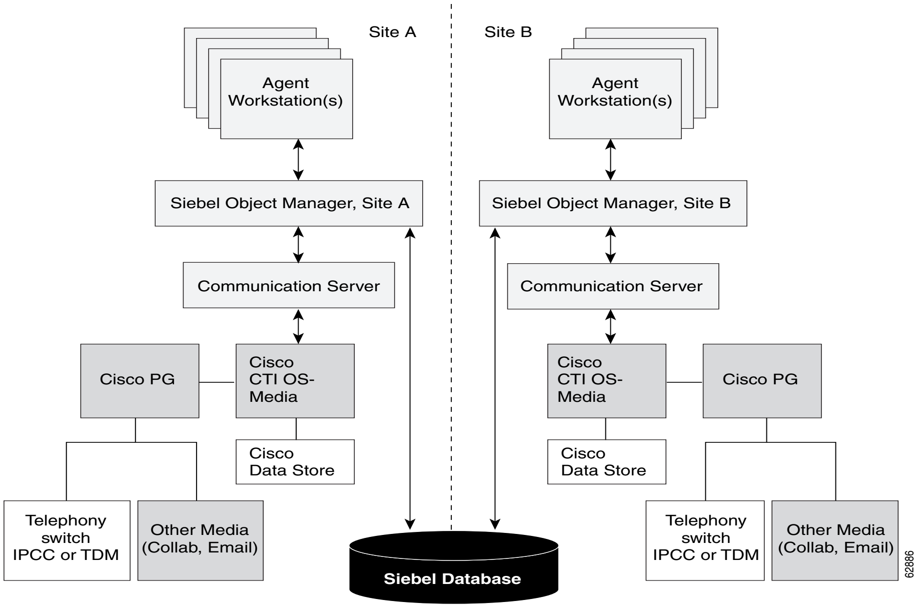
In order to maximize the ability to transfer call context and share data, sites must share a common database. This is not always practical or desirable and customer needs should prevail. Table 1-1 outlines the advantages and trade-offs of each architecture choice.

Regardless of the Siebel architecture implemented, each site contains the following Cisco components:

- **PG**. Dual Cisco Peripheral Gateway, including PIM.
- **CTI Server**. Dual Cisco CTI Servers.
- **CTI OS Server**. Dual Cisco CTI OS Servers.
- **Cisco Driver for Siebel 7**. One instance of the driver per Siebel Communication Server.

Note **Important!** Although ICM/CTI OS can transfer calls between agents across sites within an enterprise, the Siebel call context can only be transferred between agents that share a common database.

Figure 1-2 Multiple Site Architecture with Siebel 7



Note The diagram shown above is a sample of multiple site architecture with Siebel 7. It is also possible for a single Cisco Driver for Siebel 7 to communicate with multiple CTI OS servers at the same time. However, a separate CTI OS server must be installed for every ACD.

Table 1-1 Data Availability and Call Context Transfer Rules

Call Center Architecture	Data Availability	Call Context
Single shared database	Data available to any agent, regardless of call center or site	Call context can be transferred between agents regardless of call center or site
Multiple independent databases *with* data replication	Data available, but might be out-of-date Only current as of last replication	No call context transfer across sites
Multiple independent databases *without* data replication	Data never shared	No call context transfer across sites

Difference Between Siebel 6.x and Siebel 7.x Drivers

Siebel 7 supports multiple client interfaces, ranging from desktop-installed applications to thin browser-based and wireless interfaces. In addition, Siebel 7 provides support for screen pop on multiple media events, such as from email, collaboration, and calls.

- Previously, the Cisco CTI Driver for Siebel was installed on each desktop. The new Cisco Driver for Siebel 7 is installed once on the Siebel Communication Server.
- The Cisco Driver for Siebel 6 cannot be used with Siebel 7. Cisco Driver for Siebel 7 does not work with Siebel 6.

- The Cisco Driver for Siebel 7 only runs on a Siebel Communications server running on Windows server. For additional information, refer to the appropriate release of the *Hardware & System Software Spec. (Bill of Materials) for Cisco ICM/IPCC Enterprise & Hosted Editions* available at: http://www.cisco.com/en/US/products/sw/custcosw/ps1001/products_user_guide_list.html.
- All CTI OS-supported switches are supported, including IPCC. Media Termination for IPCC on Siebel thin-client interface is not supported. (See Chapter 5, "Troubleshooting and Testing" for a list of supported switches.)
- Fault tolerance is supported on CTI OS; for example, if CTI OS Server A shuts down, the system switches over to CTI OS Server B. If there is no CTI OS Server B, an error message displays and the client is reconnected automatically as soon as the CTI OS server starts running again.

Supported Features

Cisco Driver for Siebel 7 supports the following features:

- All call information from ICM (CTI Server) is available to a Siebel application. Siebel applications can use any data parameter for screen pop, including but not limited to ANI, DNIS, call variables, and extended call variables.
- Current call context, that is, the current Siebel open forms and context are transferred during call transfer or conference.
- Statistics on Siebel application.
- Outbound Option functionality.
- Siebel agent calls can be silently monitored with Call Manager 6.0's silent monitor implementation.

Minimum Software Requirements

For details about the software requirements for CTI OS in the 7.0(0) Release, refer to the appropriate release of the *Hardware & System Software Spec. (Bill of Materials) for Cisco ICM/IPCC Enterprise & Hosted Editions* available at:
http://www.cisco.com/en/US/products/sw/custcosw/ps1001/products_user_guide_list.html.

CHAPTER 2

Installation

This chapter discusses installation and configuration of:

- Cisco Driver for Siebel 7
- Cisco Data Store

This chapter also discusses configuration guidelines.

Installation Instructions

This section provides the installation instructions for the Cisco Driver for Siebel 7.

The Cisco CTI Driver for Siebel 7 versions on the CD supports the Siebel Release 7.5.x or later.

Note The ICM 7.0 Release of CTI OS does not provide support to Siebel Release 7.0.x. For more details, refer to the the *Cisco Compatibility Matrix.*

Install the Cisco CTI Driver for Siebel 7 from Siebel75plus directory by running setup.exe (use all defaults).

Step 1 Install CTI OS Server, preferably in a duplex configuration.

Note Refer to the *CTI OS System Manager's Guide for Cisco ICM/IPCC Enterprise & Hosted Editions* for CTI OS Server installation instructions.

Step 2 On the Communication Administration screen in Siebel 7, define a new Configuration and Profile object for each CTI OS server.

Step 3 Install the Cisco Driver for Siebel 7 in the Communication Server. You must enter the full path name in the “LibraryName” parameter (see the configuration file, below).

Note If the Driver is installed in the BIN subdirectory of the Siebel Comm Server installation, then the LibraryName parameter can contain only the DLL file name and does not need to use the full path name.

If you accept all defaults during the Driver installation, the Driver is installed into the C:\Program Files\Cisco Systems\SiebelDriver directory. To ensure Siebel knows the location of the Driver, in the DEF file Driver "Library Name" parameter, add the location where the Driver is located. Be sure to add the complete path to the Sieb7CiscoCTI.dll by specifying the following in the DEF file:

```
[Driver:Cisco Driver]
      Library Name   = "C:\Program Files\Cisco Systems\SiebelDriver\Sieb7CiscoCTI.dll"
      Outbound Flag  = "Y"
      Channel String = "CISCO Phone"
      Channel Type   = "Voice"
      Icon File      = "voice.gif"
      Interactive    = "Y"
      Inbound Flag   = "Y"
```

Note If the Library Name = "Sieb7CiscoCTI.dll," then Siebel tries to load the DLL from the SiebSrvr/BIN directory. Refer to the Siebel Bookshelf for more details.

Step 4 Edit the sample configuration file, cisco_default_ENU.def (shown below), distributed on the Cisco CTI Driver for Siebel 7 CD.

Note Only *part* of the sample configuration file, cisco_default_ENU.def, displays below.

Driver parameters can be overridden by profile parameters.

```
[Profile:CiscoProfile_forIPCC]
   Driver:SideAPort        = "42028"
   Driver:PeripheralID     = "1"
   Driver:SideBHost        = "CTIOSServerHostNameB"
   Driver:SideAHost        = "CTIOSServerHostNameA"
   Driver:SideBPort        = "42028"
   Driver                  = "Cisco Driver"

   [Driver:Cisco Driver]
      Library Name   = "C:\Program Files\Cisco
        Systems\SiebDriver7\Sieb7CiscoCTI.dll"
      Outbound Flag  = "Y"
      Channel String = "CISCO Phone"
      Channel Type   = "Voice"
      Icon File      = "voice.gif"
      Interactive    = "Y"
      Inbound Flag   = "Y"

      [Driver Parameters:Cisco Driver]
         Required:Driver:SideAPort        = "42028"
         Driver:DriverLogFile             = "ctc.log"
         Driver:IsEasySim                 = "FALSE"
         Service:SelectDN                 = "{@SelectedDN}"
         Service:ACDDNList                = "{@ACDDNList}"
         Required:Driver:PeripheralID     = "5000"
         Service:MaxLogKB                 = "128"
         Driver:SideBHost                 = "CTIOSServerBHost"
         Driver:DataServerName            = "CiscoDataStoreHostName"
         Driver:DataServerPort            = "42027"
         Service:HasForward               = "FALSE"
         Required:Service:DNList          = "{@DNList}"
         Required:Service:IsQueueRequired = "TRUE"
         Required:Driver:SideAHost        = "CTIOSServerAHost"
         Required:Driver:NetworkType      = "ncacn_ip_tcp"
         Required:Driver:LibraryName      = "Sieb7CiscoCTI.dll"
```

```
Service:ServiceLogFile          = "ctc_{@Username}.log"
Required:Name                   = "Cisco Driver"
Service:BroadcastStatistics     = "FALSE"
Driver:SideBPort                = "42028"
Driver:LogDebug                 = "TRUE"
Driver:ViewBmkCookie            = "CallVariable10"
```

The following table provides the definitions for the Cisco Driver parameters.

Cisco Driver Parameter	Definition
SideAPort	CTI OS Server port number for Side A
SideBPort	CTI OS Server port number for Side B
SideAHost	CTI OS Server name or IP address for Side A
SideBHost	CTI OS Server name or IP address for Side B
DataServerName	Name/IP address of the machine where Cisco Data Store is installed
DataServerPort	Cisco Data Store Listen port number
ViewBmkCookie	Call variable used for transferring SiebelView bookmark. During a transfer or conference, a view bookmark is exchanged between members of the call. The allowed values are: CallVariable1, CallVariable2, … CallVariable10. The default is CallVariable10. Note that if a call variable is being used for this purpose, it cannot be used for any other purpose.

The following table provides the definitions for the Cisco Service parameters.

Cisco Service Parameter	Definition
Service:BroadcastStatistics	Enables Agent and SkillGroupStatistics broadcast. The following values are available for this parameter: • True: Enables Agent and SkillGroupStatistics broadcast. • False (default): Disables Agent and SkillGroupStatistics broadcast.
Service:AutoLogout	Automatically logs the agent out of the ACD queue when the agent logs out of the Siebel session. The Siebel contact center administrator can use this parameter to configure the AutoLogout behavior for all agents that belong to a particular Siebel 7 configuration. The following values are available for this parameter: • True: Agent is automatically logged out of the ACD queue when the agent logs out of the Siebel session. **Note** This setting applies whether the agent specifically exited the Siebel application or the Web browser, or if the agent's application session times out. • False (default): Agent is not automatically logged out of the ACD queue when the agent logs out of the Siebel session. **Note** See Chapter 5, "Troubleshooting and Testing," for more information about this parameter.
Service:AutoLogoutReasonCode	Reason code for AutoLogout request in integer format. (Default is 0.) **Note** This is an optional parameter. If omitted, the default value is 0.

Step 5 The Driver log file is generated with a default file name in the directory where the corresponding Siebel executable that loads the driver DLL resides. For example, the Siebel Thin Client log file resides in the SiebSrvr\BIN directory; so the Dedicated Siebel client log file will be generated in the SiebClient\BIN directory. The name and location of the Driver Log file can be overridden using the following Registry key before the actual driver DLL is loaded (it can not be changed dynamically):

After the driver installation, the registry for the Trace Settings is:

HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems, Inc.\CTIOS Tracing

- "TraceFileName"="CtiosClientLog"
- "TraceMask"=dword:40000307
- "MaxDaysBeforeExpire"=dword:00000007
- "MaxFiles"=dword:00000005
- "MaxFileSizeKb"=dword:00000800
- "FlushIntervalSeconds"=dword:0000001e
- "TraceServer"="C:\\PROGRA~1\\CISCOS~1\\SIEBEL~1\\CTIOST~1.EXE"

For example, a possible value for this key would be:

C:\DriverLogs\CiscoDriverLog

Step 6 Import the configuration (.def) file into Siebel once all edit changes are completed.

Step 7 Add a new CTI Teleset with the appropriate extensions, and add a new CTI user, giving that user an ACD login and password, if necessary.

```
[Command:LoginToPBX]
   Profile       = ""
   DeviceCommand = "LogIn"
   MenuPosition  = "20.1"
   Comments      = ""
   Title         = "Log In (Phone)"
   CmdData       = "LoginToPBX"

   [CmdData:LoginToPBX]
      Param.AgentWorkMode = "0"
      Param.UserId        = "{@UserName}"
      Comments            = ""
      Param.Password      = "{@AgentPin}"
      Param.ACDQueue      = "{@QueueList}"
      Param.AgentId       = "{@AgentId}"
```

Step 8 For the Siebel Dedicated (thick) client make sure that CTI is enabled in Siebel and other related Siebel flags are set as desired (refer to the Siebel doc for more info) by updating the Communication section of the uagent.cfg file. See the sample below.

```
[Communication]
CommEnable             = TRUE
CommSimulate           = FALSE
CommLocalDriver        = TRUE
CommLogDebug           = TRUE
CommReleaseLogHandle   = TRUE
CommConfigManager      = FALSE
CommReqTimeout         = 600
CommLogFile            = SComm.log
```

You can also modify appropriate parameters on the corresponding Siebel Server components for Siebel thin clients.

Step 9 For a screen pop, make sure you have appropriate settings in the User Preferences Siebel screen.

Refer to the *Siebel Call Center User Guide* for information on how to configure a call center, and define agents, telesets, and extensions.

Note In CTI OS Releases 6.0 and later, updates to Cisco CTI OS software (Engineering Specials, Service Releases and Maintenance Releases) are installed with Patch Manager. Once installation completes you cannot move any CTI OS files from the directories in which they are installed, or Patch Manager will be unable to perform CTI OS software updates correctly.

Note Cisco Siebel driver doesn't work if CTI OS Server has security turned ON. The security has to be turned OFF on CTI OS Server in order for CTI OS Clients, using Siebel Driver, to connect.

Uninstalling Cisco Driver for Siebel 7

To uninstall the Cisco Driver for Siebel 7, run Add/Remove programs from the Windows Control Panel and select the Cisco Driver for Siebel 7 Uninstall.

Note If Release 6.0 software patches for any CTI OS components are installed on your machine, you must uninstall these patches before you uninstall the Cisco Driver for Siebel 7 software.

Cisco Data Store: Installing

The Cisco Data Store is used to store the Siebel application call context data during conferences, transfers, and internal calls. This section presents the information that is necessary to install the Cisco Data Store.

Cisco Data Store Installation

The Cisco Data Store must be installed on a machine accessible by any server running the Siebel Driver. The Cisco Data Store's IP address and port number are not blocked by any firewall or security software. For the minimum operating system requirements, refer to the appropriate release of the *Hardware & System Software Spec. (Bill of Materials) for Cisco ICM/IPCC Enterprise & Hosted Editions* available at: http://www.cisco.com/en/US/products/sw/custcosw/ps1001/products_user_guide_list.html.

Note We do not recommend installing the Cisco Data Store on any of the Siebel COM servers (or any other machine that might become COM intensive).

How to install Cisco Data Store

Step 1 Locate and run the program **Setup.exe** in the CiscoDataStore folder of the CD. The License Agreement screen appears.

Step 2 Click the **Yes** button. A destination screen appears.

Step 3 Click the **Next** button to accept the defaults and begin the installation process.

Step 4 After installation has completed, click the **Finish** button.

During installation, the following registry keys are added that Cisco Data Store uses:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems\Ctios\ObjectStoreServerSocket]
[HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems\Ctios\ObjectStoreServerSocket\Connections]
"HeartbeatIntervalMs"=dword:0000ea60
"HeartbeatRetrys"=dword:00000005
"ListenPort"=dword:0000a42d
[HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems\Ctios\ObjectStoreServerSocket\Logging]
"TraceFileName"="ObjectStoreServerSocketLog"
"TraceMask"=dword:00000007
```

Observe the following guidelines when making changes to the default values:

- **ListenPort** (default is 42029). When changing this value, make sure that the port number does not conflict with other values and there is no other application listening on the same port number. Also, make sure that the ListenPort value is used by the driver site as a value for the DataServerPort Driver parameter.
- **TraceFileName**. Provide a different name or the complete path. For troubleshooting purposes you can change a TraceMask (see Chapter 5, “Troubleshooting and Testing,” for more details).

Note Important! Do not change the HeartbeatIntervalMs and HeartbeatRetrys values.

How to start Cisco Data Store

Cisco Data Store is installed as a service, so it needs to be running in order for the Driver to connect to it.

Step 1 Start Cisco Data Store from the ICM Service Control application by selecting **Start > Programs > Cisco Systems CTI OS > ICM Service Control**.

Step 2 Select **ctiosdatastore** and click the **Start** button.

After Cisco Data Store starts, make sure it listens on the correct port.

Note In CTI OS Releases 6.0 and later, updates to Cisco CTI OS software (Engineering Specials, Service Releases and Maintenance Releases) are installed with Patch Manager. Once installation completes you cannot move any CTI OS files from the directories in which they are installed, or Patch Manager will be unable to perform CTI OS software updates correctly.

How to Uninstall Cisco Data Store

To uninstall the Cisco Data Store, run Add/Remove programs from the Windows Control Panel and select the Cisco Data Store Uninstall.

Note If Release 6.0 software patches for any CTI OS components are installed on your machine, you must uninstall these patches before you uninstall the Cisco Data Store software.

Guidelines for Configuring the CTI Driver for Siebel 7

Profiles

A *profile* is a logical instance in Siebel that basically points to a driver (DLL) name/location for a particular media type. (Profile parameters can override Driver parameters.)

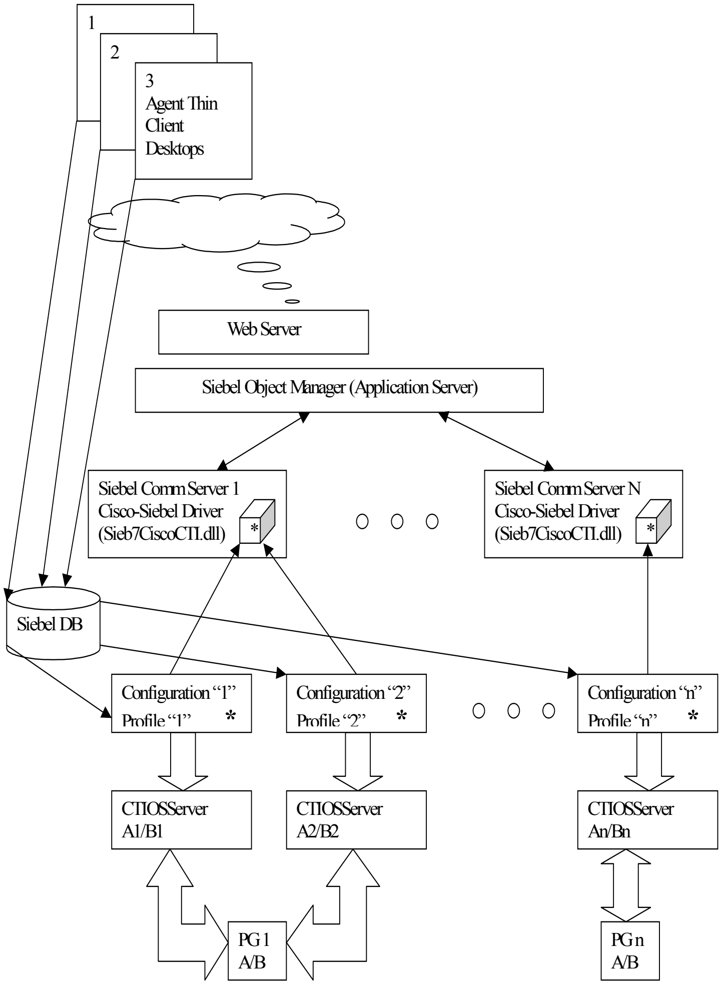
Every Siebel user belongs to a particular Siebel configuration.

Multiple profiles can be associated with a configuration, but only one profile is allowed for voice type media with the Cisco DLL name and should be listed under the configuration.

Note Make sure you create one configuration and one profile for each CTI OS Server pair.

Figure 2-1 displays the step-by-step process occurring underneath when Siebel agents log in to CTI with the Cisco CTI Driver for Siebel 7 installed on all involved Siebel Comm Servers at the same location.

Figure 2-1 Step-By-Step Process of Siebel Agents Logging in to CTI



1. Siebel Thin Agent logs in via the browser to a call center.
2. Agent receives all data from the Siebel database; in particular, the configuration data for that agent and all profile data along with the Driver parameters (list of all possible parameters is found in the sample Cisco DEF file). (Those parameters include the CTI OS Server name and the port number for Sites A and B.) The Peripheral ID is received as a parameter along with the driver (DLL) name and location.
3. A corresponding Siebel Comm Server loads a Cisco CTI Driver for Siebel 7 instance <DLL> (in case it was not loaded yet).
4. The driver creates a driver/service object for this agent and connects to a corresponding CTI OS Server(s) (specified via DEF file) above.
5. The same driver can handle connections to different CTI OS Servers at the same time.

Configuration Guidelines

Observe the following when configuring the CTI Driver for Siebel 7:

- In a DEF file the Driver:SideAHost, Driver:SideBHost, Driver:SideAPort, and Driver:SideBPort parameters should not be set to blank (""). If there is only one CTI OS Server installed (site A), the parameters for site B must still be set to the same values and should not be left blank.
- The DEF file Required:Driver:PeripheralID = "5006" Driver parameter should match the Peripheral ID on the CTI OS Server.
- Make sure that the Peripheral Type for this Peripheral ID was chosen correctly during the CTI OS Server installation (can be verified/changed in the Registry on the CTI OS Server machine).

CHAPTER 3

Customization

The Siebel application can be customized in many different ways to meet the needs of a specific client. There is no fully featured out of the box Siebel application offered by Cisco with the Cisco Siebel Driver. The Cisco Driver for Siebel 7 provides an out of the box solution to screen pop calls using any customer-selected variable, such as ANI, DNIS, Call or ECC variables.

The customization process involves setting up rules in the Siebel application that takes events created by the Cisco Driver for Siebel 7 and maps them to screen-pop operations in the Siebel application. For example, different screen pops might be required when a call arrives from a new versus an existing customer, when the call is placed to a different DNIS, when the call comes from an outside caller, or when it is queued to a specific skill group.

Customization requires mapping Cisco ICM/CTI OS events and commands in the Siebel application. In order to perform well, customization requires knowledge of the ICM CTI events with a thorough working knowledge of the Siebel product. A Siebel Certified Engineer or partner with prior Siebel/CTI experience is required for product customization.

Note **Important!** The Cisco Driver for Siebel 7 provides an out-of-the-box solution to screen pop calls using any customer-selected variable, such as ANI, DNIS, Call, or ECC variable.

Siebel Customization Basics

When the Cisco Driver receives call and agent state change events, it creates a corresponding Siebel event. This triggers a set of actions defined within the *Siebel database*, such as a screen pop performed on an incoming call.

The Siebel application is highly customizable. You can use the Siebel Visual Basic-like or Java-like scripting language for additional customization. The customizable Siebel database tables can be imported or exported to plain text DEF files. An example of a DEF file is shown in Example 3-1.

A Siebel event consists of a Siebel event name followed by one or more parameters. These parameters can be used in conjunction with the customization code to determine what action to take on any event. Table 3-1 maps the Cisco Driver for Siebel 7 events to Siebel event names. The parameters passed with every call event are listed in Table 3-3.

Typical arguments passed with an event, for example, OnCallEstablished (the equivalent Siebel event is EventAnswer), would include ANI, DNIS, CallStatus, Call Variables, and more.

In order to understand the Siebel customization process better, consider the issues involved when a call arrives and is answered by an agent. For this example (see Example 3-1), assume screen pop uses the customer account number passed in CallVariable1:

- If an inbound call is received on the primary agent's extension, a screen pop should be performed using the customer account number, passed in this example in CallVariable1.
- If the call does not have an account number, a new customer form should screen pop. (Some fields might automatically be pre-populated with information from other call variables.)
- If the call is a transfer from another agent, the agent should go to the NotReady state and receive a transfer of the current agent's screen context, that is, the forms that the agent had been using in handling this call. They might already contain pre-populated information with the customer's name, address, preferences, and order history.

Example 3-1 Sample Siebel DEF File for Event Handling

```
; --- Internal Agent to Agent Call ---
[EventHandler:InsideCallReceivedFromQueue]
Order = "1"
DeviceEvent          = "EventAnswer"
Response             = "OnInsideCallReceived"
Filter.SiebelCall    = "?*"

; --- Handle inbound customer call ---
[EventHandler:InboundConsumerCall]
Order = "2"
DeviceEvent          = "EventAnswer"
Response             = "InboundConsumerCall"
Filter.CallVariable1 = "?*"
Filter.CallStatus    = "ACDIn"

[EventResponse:InboundConsumerCall]
QueryBusObj          = "Consumer"
QueryBusComp         = "Consumer"
QuerySpec            = "[CSN]= '{CallVariable1}'"
SingleView           = "Consumer Detail View"
FindDialog           = "Consumer"
FindField.CSN        = "{CallVariable1}"
SingleLog            = "LogIncomingCallConsumerFound"
Log                  = "LogIncomingCallConsumerNotFound"

[EventLog:LogIncomingCallConsumerNotFound]
         AfterWork.'ACD Call Duration' = "{@WorkDuration}"
         Comments                      = ""
         LogField.Description          = "Unknown Caller ({CallVariable1})"
         LogField.Comment              = "Demo:Inbound Call Consumer Not Found"
         BusObj                        = "Action"
         BusComp                       = "Action"
         LogField.Type                 = "Call - Inbound"

      [EventLog:LogIncomingCallConsumerFound]
         AfterWork.'ACD Call Duration' = "{@WorkDuration}"
         AfterWork.'Planned'           = "{@WorkStartTime}"
         AfterWork.'Planned Completion'= "{@Now}"
         Comments                      = ""
         Display                       = "TRUE"
         LogField.Description          = "Call - Inbound"
         LogField.Comment              = "Consumer: {CallVariable1} {CSN}"
         BusObj                        = "Action"
         BusComp                       = "Action"
         LogField.Type                 = "Call - Inbound"
```

Note Sample DEF files, cisco_default_ENU.def, and cisco_agentANDskillgroupSTATISTICS_sample_ENU.def, are distributed on the product CD.

Reading the DEF File

Note A DEF file consists of bracket-delineated sections. Shown in this example are EventHandler and EventResponse sections. EventHandlers are executed when Siebel receives an event. Each EventHandler describes a set of specific arguments that must be matched. The Siebel application evaluates each EventHandler based on the order parameter.

In Example 3-1, two EventHandler sections are defined. (The order parameter determines evaluation preference.) The first of the two EventHandler entries filters out inside calls. The SiebelCall parameter is used by Siebel to store the current call context. The ="?*" syntax specifies that the parameter has a non-empty value. If this condition is satisfied, the Siebel application looks for an EventResponse called "OnInsideCallReceived," which is not shown for simplicity.

The second EventHandler filters on both call status and call variable 1. The type of call must be ACDIn, a parameter value created by the Cisco Driver to Siebel, and call variable 1 must not be empty. If these conditions are met, the Siebel application looks for an EventResponse called "InboundConsumerCall," which is shown. Call arguments, call types, and other special values are listed in Table 3-3.

The "InboundConsumerCall" event response performs several steps:

- Defines which business object is affected
- Decides which database query to perform and on which field
- Defines a Find dialog box and how to populate it with data
- Logs entries into the Siebel log for success or failure of the response

In this case, CallVariable1 is used to query the Consumer business object and to populate the Find dialog box.

Note In Siebel, the [ ] and { } syntax have special meaning. Items within square brackets ([ ]) refer to a business object field. Items surrounded by curly braces ({ }) refer to variables passed with the event.

Refer to the *Siebel Call Center User Guide* for more details.

Siebel definition files often contain other more complex information. The DEF file is defined and executed by Siebel. The exact syntax and complete description of arguments and options available is defined in Siebel documentation.

Note **Important!** The default DEF file provided with the Cisco CTI Driver for Siebel 7 is only used for demonstration purposes. It should not be used in a production environment.

Table 3-1 Cisco Driver for Siebel 7 Events

Siebel Event	Arguments Passed
EventAgentStatistics	Receives and broadcasts agent statistics Arguments: See Table 3-2 Parameters Passed with Broadcast Statistics Events CTI OS Event: QueryAgentStatisticsConf CTI Server Event: QUERY_AGENT_STATISTICS_CONF Observe the following: • A corresponding EventLog will be used to update a Broadcast table (corresponding record). • A placeholder record with "—" as the body, but a corresponding Abstract field and a corresponding recipients list must be created prior to processing this event using Siebel scripts. See Example 3-2 for more details. Chapter 4, "Broadcast Statistics," provides more information.
EventSkillGroupStatistics	Receives and broadcasts skill group statistics Arguments: See Table 3-2 Parameters Passed with Broadcast Statistics Events CTI OS Event: QuerySkillGroupStatistics CTI Server Event: QUERY_SKILL_GROUP_STATISTICS Observe the following: • A corresponding EventLog will be used to update a Broadcast table (corresponding record). • A placeholder record with "—" as the body, but a corresponding Abstract field and a corresponding recipients list must be created prior to processing this event using Siebel scripts. See Example 3-3 for more details. Chapter 4, "Broadcast Statistics," provides more information.

Table 3-1 Cisco Driver for Siebel 7 Events (continued)

Siebel Event	Arguments Passed
EventSkillGroupRemove	Special event that uses the skill group number information as a parameter (provided to Siebel) that invokes the RemoveUserFromBroadcast Siebel service method to update a recipient list (removes a corresponding agent from a recipient list) **Note** This event is sent when the agent logs out or when the user closes a browser with the Siebel application. CTI Server Event: AGENT_STATE_EVENT to logout See Example 3-4 for more details. Chapter 4, "Broadcast Statistics," provides more information.
EventSkillGroupAdd	Special event that provides Siebel with the agent's skill group number information as a parameter. The AddUserToBroadcast Cisco business service method can be invoked to create a record holder with a corresponding Abstract field in the Siebel Broadcast table (if not yet created) and to update a recipients list on it. CTI Server Event: AGENT_STATE_EVENT when agent logs into an ACD (at this point the skill group information is already known) See Example 3-5 for more details. Chapter 4, "Broadcast Statistics," provides more information.
EventDialing	CTI OS Event: OnCallOriginated CTI Server Event: CALL_ORIGINATED_EVENT
EventRinging	Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallDelivered CTI Server Event: CALL_DELIVERED_EVENT
EventAnswer	Issued to Siebel by the Driver if InboundCall (alerting call on agent device) is connected Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallEstablished CTI Server Event: CALL_ESTABLISHED_EVENT

Table 3-1 Cisco Driver for Siebel 7 Events (continued)

Siebel Event	Arguments Passed
EventEstablished	Issued to Siebel by the Driver if OutboundCall (initiated call on agent device) is connected Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallEstablished CTI Server Event: CALL_ESTABLISHED_EVENT
EventHeld	Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event name: OnCallHeld CTI Server Event: CALL_HELD_EVENT
EventRetrieved	Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallRetrieved CTI Server Event: CALL_RETRIEVED_EVENT
EventReleased	Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallCleared CTI Server Event: CALL_CLEARED_EVENT
EventUpdateCurCallData	Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallDataUpdate for current call CTI Server Event: CALL_DATA_UPDATE_EVENT
EventTransferred	Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallTransferred CTI Server Event: CALL_TRANSFERRED_EVENT
EventConferenced	Arguments: See Table 3-3 Parameters Passed with Call Events CTI OS Event: OnCallConferenced CTI Server Event: CALL_CONFERENCED_EVENT
EventAgentLogin	Cause: Agent logged into ACD CTI OS Event: OnAgentStateChange CTI Server Event: AGENT_STATE_EVENT and new agent state is Login
EventAgentLogout	Cause: Agent logged out from ACD CTI OS Event: OnAgentStateChange CTI Server Event: AGENT_STATE_EVENT and new agent state is Logout

Table 3-1 Cisco Driver for Siebel 7 Events (continued)

Siebel Event	Arguments Passed
EventAgentReady	Cause: Agent state set to Ready CTI OS Event: OnAgentStateChange CTI Server Event: AGENT_STATE_EVENT and new agent state is Ready
EventAgentNotReady	Cause: Agent state set to NotReady CTI OS Event: OnAgentStateChange CTI Server Event: AGENT_STATE_EVENT and new agent state is Not Ready
EventAgentBusy	Cause: Agent state set to WorkReady CTI OS Event: OnAgentStateChange CTI Server Event: AGENT_STATE_EVENT and new agent state is Busy (WorkReady)
EventAgentNotBusy	Cause: Agent state set to WorkNotReady CTI OS Event: OnAgentStateChange CTI Server Event: AGENT_STATE_EVENT and new agent state is Not Busy (WorkNotReady)
EventServerUnavailable	CTI OS Event: OnCtiOSFailure CTI Server Event:Connection to CTI Server broken
EventError	CTI OS Event: Error generated on last request
EventUserMessage	CTI OS Event: USER_MESSAGE_EVENT

Example 3-2 Using EventAgentStatistics Device Event to Update Broadcast Table

```
[EventHandler:EventAgentStatistics]
   Filter.Body = "?*"
   Profile     = ""
   Comments    = ""
   Order       = "15"
   Response    = "OnEventAgentStatistics"
   DeviceEvent = "EventAgentStatistics"

   [EventResponse:OnEventAgentStatistics]
      Comments = "Agent Statistics Information to be translated into Broadcast Bar Message"
      Log      = "LogEventAgentStatistics"

      [EventLog:LogEventAgentStatistics]
         QuerySpec                        = "[Abstract]='CTIAgentStat_{@UserName}'"
         LogField.'All'                   = "N"
         LogField.'Activation Date/Time'  = "{Activation Date/Time}"
         Comments                         = "Takes Agent Statistic information and inserts/updates Message
Broadcast table"
         LogField.'Expiration Date/Time'  = "{Expiration Date/Time}"
         BusObj                           = "Broadcast Message"
         LogField.'Body'                  = "{Body}"
         LogField.'Type'                  = "{Type}"
         BusComp                          = "Broadcast Message"
         LogField.'Abstract'              = "CTIAgentStat_{@UserName}"
```

Example 3-3 Using EventSkillGroupStatistics Device Event to Update Broadcast Table

```
[EventHandler:EventSkillGroupStatistics]
   Filter.Body = "?*"
   Profile     = ""
   Comments    = ""
   Order       = "16"
   Response    = "OnEventSkillStatistics"
   DeviceEvent = "EventSkillGroupStatistics"

   [EventResponse:OnEventSkillStatistics]
      Comments = "Skill Statistics Information to be translated into Broadcast Bar Message"
      Log      = "LogEventSkillStatistics"

      [EventLog:LogEventSkillStatistics]
         QuerySpec                        = "[Abstract]='CTISkillStat_{SkillGroupNumber}'"
         LogField.'All'                   = "N"
         LogField.'Activation Date/Time' = "{Activation Date/Time}"
         Comments                         = "Takes Skill Statistic information and inserts/updates Message
Broadcast table"
         LogField.'Expiration Date/Time' = "{Expiration Date/Time}"
         BusObj                           = "Broadcast Message"
         LogField.'Body'                  = "{Body}"
         LogField.'Type'                  = "{Type}"
         BusComp                          = "Broadcast Message"
         LogField.'Abstract'              = "CTISkillStat_{SkillGroupNumber}"
```

Example 3-4 Using EventSkillGroupRemove Device Event to Update Broadcast Table

```
[EventHandler:EventSkillGroupRemove]
   Filter.SkillGroupNumber = "?*"
   Profile     = ""
   Comments    = ""
   Order       = "1"
   Response    = "OnEventSkillGroupRemove"
   DeviceEvent = "EventSkillGroupRemove"

   [EventResponse:OnEventSkillGroupRemove]
      Comments              = ""
      ServiceMethod = "Cisco Broadcast Stats.RemoveUserFromBroadcast"
      ServiceParam.UserName = "{@UserName}"
      ServiceParam.Abstract = "CTISkillStat_{SkillGroupNumber}"
```

Example 3-5 Using EventSkillGroupAdd Device Event to Update Broadcast Table

```
[EventHandler:EventSkillGroupAdd]
   Filter.SkillGroupNumber = "?*"
   Profile     = ""
   Comments    = ""
   Order       = "1"
   Response    = "OnEventSkillGroupAdd"
   DeviceEvent = "EventSkillGroupAdd"
```

```
[EventResponse:OnEventSkillGroupAdd]
   Comments              = ""
   ServiceMethod = "Cisco Broadcast Stats.AddUserToBroadcast"
   ServiceParam.UserName = "{@UserName}"
   ServiceParam.Abstract = "CTISkillStat_{SkillGroupNumber}"
```

Table 3-2 Parameters Passed with Broadcast Statistics Events

Parameter Name	Arguments Passed
Abstract	Contains "CTIAgentStat_" prefix and a Siebel agent name for the AgentStatsitics broadcast record or "CTISkillStat_" prefix and a skill group number. **Note** The broadcast message's Abstract field determines whether it is used to display agent statistics or skill group statistics.
Activation Date/Time	Activation date and time to start a broadcast
All	Indicates list of recipients. Values are "Y" or "N." ("Y" broadcasts to everyone.)
Body	Body of Broadcast message that contains the agent or skill group number and the statistics name/value pair for configured statistics.
Expiration Date/Time	Expiration date and time to stop a broadcast.
Type	Broadcast message type. **Note** Refer to the Siebel documentation for more details.

Table 3-3 Parameters Passed with Call Events

Parameter Name	Arguments Passed
ANI	Automatic Number Identification, *string*
CallID	Call identifier, *string*
CallType	Call type. Values are listed in Table 3-4 Additional Parameters for the Event Transferred Device Event, *string*
CallVariable1, ... CallVariable10	Call-related variable data, *string*
CallWrapupData	Call-related wrap-up data, *string*
CED	Caller Entered Digits in response to IVR prompting, *string*
DialedNumber	The number dialed, *string*
DNIS	Dialed Number Identification Service, *string*
LineType	Line type. Values are listed in Table 3-6 Line Types, *string*

Table 3-3 Parameters Passed with Call Events (continued)

Parameter Name	Arguments Passed
UserToUserInfo	ISDN user-to-user information, *string*
VarName1, ... VarNameN VarArray[0], ... VarArray[n]	The value contained in the named Expanded Call Context variable. Expanded Call Context variables might consist of an arbitrary number of named variable fields, subject only to a combined total limit of 2000 bytes. These variables are available only if the Expanded Call Context feature is explicitly enabled in ICM software. For specifics on enabling, creating, and naming the ECC variables, refer to the *ICM Scripting and Media Routing Guide for Cisco ICM/IPCC Enterprise & Hosted Editions* .

In addition to the parameters for the DeviceEvent listed in Table 3-3, the Device Event EventTransferred uses the parameters listed in Table 3-4:

Table 3-4 Additional Parameters for the Event Transferred Device Event

Parameter Name	Arguments Passed
PrimaryDeviceID	The connection identifier of the primary call connection
SecondaryDeviceID	The connection identifier of the secondary call connection
TransferringDeviceID	The device identifier of the device that transferred the call
TransferredDeviceID	The device identifier of the device to which the call was transferred
NumParties	The number of active connections associated with this conference call. This value also indicates the number of ConnectedPartyCallID, ConnectedPartyDeviceIDType and ConnectedPartyDeviceID floating fields present in the floating part of the message.
ConnectedPartyCallID1 ... ConnectedPartyCallID6	The Call ID value assigned to one of the call parties. There may be more than one ConnectedPartyCallID field in the message (see NumParties, above).

Table 3-4 Additional Parameters for the Event Transferred Device Event (continued)

Parameter Name	Arguments Passed
ConnectedPartyDeviceIDType1 ... ConnectedPartyDeviceIDType6	Indicates the type of the device identifier supplied in the following ConnectedPartyDeviceID floating field. There may be more than one ConnectedPartyDeviceIDType field in the message (see NumParties, above).
ConnectedPartyDeviceID1 ... ConnectedPartyDeviceID6	The device identifier of one of the call parties. There may be more than one ConnectedPartyDeviceID field in the message (see NumParties, above).

Likewise, in addition to the parameters for the DeviceEvent listed in Table 3-3, the Device Event EventConferenced uses the parameters listed in Table 3-5:

Table 3-5 Additional Parameters for the Event Conferenced Device Event

Parameter Name	Arguments Passed
PrimaryDeviceID	The connection identifier of the primary call connection
SecondaryDeviceID	The connection identifier of the secondary call connection
ControllerDeviceID	The device identifier of the conference controller device
AddedPartyDeviceID	The device identifier of the device added to the call
NumParties	The number of active connections associated with this conference call. This value also indicates the number of ConnectedPartyCallID, ConnectedPartyDeviceIDType and ConnectedPartyDeviceID floating fields present in the floating part of the message.
ConnectedPartyCallID1 ... ConnectedPartyCallID6	The Call ID value assigned to one of the call parties. There may be more than one ConnectedPartyCallID field in the message (see NumParties, above).

Table 3-5 Additional Parameters for the Event Conferenced Device Event (continued)

Parameter Name	Arguments Passed
ConnectedPartyDeviceIDType1 ... ConnectedPartyDeviceIDType6	Indicates the type of the device identifier supplied in the following ConnectedPartyDeviceID floating field. There may be more than one ConnectedPartyDeviceIDType field in the message (see NumParties, above).
ConnectedPartyDeviceID1 ... ConnectedPartyDeviceID6	The device identifier of one of the call parties. There may be more than one ConnectedPartyDeviceID field in the message (see NumParties, above).

Table 3-6 Line Types

Parameter Name	Arguments Passed
DID	Direct inward call
Help	Assistance call
InboundACD	Inbound ACD call
Inside	Inside call
Message	Voice message call
OutboundACD	Outbound ACD call
Supervisor	Supervisor call
Unknown	Any purpose call

Siebel Commands

Siebel commands can be customized to translate to Driver commands through the DEF file. Commands are mapped using Device command entries. These provide a mechanism for mapping commands, tool tips, hotkeys, and passing command parameters. It is possible to use the CmdData entries to pop up dialog boxes to query for arguments before sending the command to the driver.

Example 3-6 displays a portion of a sample DEF file. The Siebel CTI toolbar button for Login needs to be invoked as a sub-command of the SignOnGroupInMenu command (refer to the Siebel documentation for more details). The entry also defines several parameters to pass with the login command.

Note The example only demonstrates the Login command.

Example 3-6 Sample DEF File

```
[Command:SignOnGroupInMenu]
   SubCommand_1 = "LoginToPBX"
   Profile      = ""
   MenuPosition = "20"
   Comments     = ""
   Title        = "Log In"
   Description  = "Log in"
   CmdData      = ""

[Command:LoginToPBX]
   Profile       = ""
   DeviceCommand = "LogIn"
   MenuPosition  = "20.1"
   Comments      = ""
   Title         = "Log In (Phone)"
   CmdData       = "LoginToPBX"

   [CmdData:LoginToPBX]
      Param.AgentWorkMode = "0"
      Param.UserId        = "{@UserName}"
      Comments            = ""
      Param.Password      = "{@AgentPin}"
      Param.ACDQueue      = "{@QueueList}"
      Param.AgentId       = "{@AgentId}"
```

Table 3-7 Cisco Driver for Siebel 7 Commands

Siebel Command	Arguments Required
AnswerCall	*No arguments*
ChangeBusyState (See Example 3-7)	*Optional:* ReasonCode **Note** Does the following to the corresponding CTI Toolbar agent state button: • If the current agent state is NotReady or WorkReady, changes agent state to Ready. • If the current agent state is Ready, changes the agent state to NotReady. On the Nortel Meridian, the command activates the button to toggle only between the Ready and the WorkReady states since there is no NotReady state. The corresponding CTI Toolbar agent state button looks pressed when the agent state is NotReady.
ChangeReadyState	*Optional:* ReasonCode AgentWorkMode* **Note** Changes agent state to Ready.
ChangeNotReadyState	*Optional:* ReasonCode **Note** Changes agent state to NotReady.
ChangeWorkReadyState	*Optional:* ReasonCode **Note** Changes agent state to WorkReady.
ConferenceComplete	*No arguments*

Table 3-7 Cisco Driver for Siebel 7 Commands (continued)

Siebel Command	Arguments Required
ConferenceInit	*Required:* Dialed number *Optional:* CallVariable1 … CallVariable10 UserToUserInfo CallWrapupData AuthorizationCode AccountCode CallMannerType* CallPlacementType* FacilityType* AlertRings CallOption* PostRoute* Priority* AnsweringMachine* *ECCVarName1 … ECCVarNameN* CallNotifyText[1]
HoldCall	*No arguments*
LogIn	*Required:* AgentID Password *Optional:* Extension Instrument NumSkillGroups SkillGroupNumber1 … SkillGroupNumberN SkillGroupName1 … SkillGroupNameN SkillPriority1 … SkillPriorityN ReasonCode AgentWorkMode*
LogOut	*Optional:* ReasonCode

Table 3-7 Cisco Driver for Siebel 7 Commands (continued)

Siebel Command	Arguments Required
MakeCall	*Required:* Dialed number *Optional:* CallVariable1 … CallVariable10 UserToUserInfo CallWrapupData AuthorizationCode AccountCode CallMannerType* CallPlacementType* FacilityType* AlertRings CallOption* PostRoute* Priority* AnsweringMachine* *ECCVarName1 … ECCVarNameN* CallNotifyText[1]
ReleaseCall	*No arguments*
ResumeSelectedCall (see Example 3-8)	TrackingID, which is driver trackingID for the selected item in WorkItemList Siebel combo box
SuspendDeselectedCall (see Example 3-8)	TrackingID, which is driver trackingID for the deselected (previously selected) item in WorkItemList Siebel combo box
ReconnectCall	TrackingID, using Siebel special variable @SelectedWorkItem:DriverWorkTrackID Combined action that clears an active call and then retrieves an existing held call
RetrieveCall	TrackingID, using Siebel special variable @SelectedWorkItem:DriverWorkTrackID High-level compound action that places an active call on hold and then either retrieves a previously held call or answers an alerting call at the same device
SendDTMFSignal	Requests ACD to send a sequence of DTMF tones *Required:* DTMFString *Optional:* ToneDuration PauseDuration
SendMessage	*Required:* AgentID MessageText
TransferComplete	*No arguments*

Table 3-7 Cisco Driver for Siebel 7 Commands (continued)

Siebel Command	Arguments Required
TransferInit	*Required:* Dialed number *Optional:* CallVariable1 … CallVariable10 UserToUserInfo CallWrapupData AuthorizationCode AccountCode CallMannerType* CallPlacementType* FacilityType* AlertRings CallOption* PostRoute* Priority* AnsweringMachine* *ECCVarName1 … ECCVarNameN* CallNotifyText[1]
TransferMute	*Required:* Dialed number *Optional:* CallVariable1 … CallVariable10 UserToUserInfo CallWrapupData FacilityCode AuthorizationCode AccountCode CallMannerType* CallPlacementType* FacilityType* AlertRings CallOption* PostRoute* Priority* AnsweringMachine* *ECCVarName1 … ECCVarNameN* CallNotifyText[1]
UnHoldCall	*No arguments*
UpdateCurCallData	*Optional:* CallVariable1 … CallVariable10 CED DialedNumber UserToUserInfo CallWrapupData *ECCVarName1 … ECCVarNameN*

* See Table 3-8 on page 3-19 for a list of available values to use with this command parameter.

1. The ECC variable "user.SiebelCNT" must be configured in ICM using the Configuration Manager in order for Param.CallNotifyText (set on call sending side) gets propagated to the call receiving side. Siebel CallNotifyText is a parameter that can be used to notify about incoming call. This parameter's value is displayed on the Siebel status bar (where all the agent /call states are displayed) upon call arrival. CallNotifyText can be provided via a DEF file (using the Param.CallNotifyText command data parameter for the MakeCall, ConferenceInit, TrnsferInit, and TransferMute device commands) or it can be set in a router script directly into the user.SiebelCNT ECC variable. Siebel CallNotifyText is not restored for recovered calls on failover.

Example 3-7 Sample for Toggling Between the Ready and NotReady States and Ready and WorkReady States Using the ChangeBusyState Command

```
;//////////////////////////////////////////////////////////////////////
;// For "NotReady" button
;//
[Command:NotReadyGroup]
   SubCommand_1 = "NotReadyGroupSetToNotReady"
   SubCommand_2 = "NotReadyGroupSetToReady"
   Profile      = ""
   Comments     = ""
   Description  = "Change ready state"
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

   [Command:NotReadyGroupSetToNotReady]
   ExecuteAll   = "TRUE"
   SubCommand_1 = "NotReadyForPhoneSetToNotReady"
   SubCommand_2 = "NotReadyForPhoneSetToReadyNoPopup"
   Profile      = ""
   Comments     = ""
   ExecuteAll   = "TRUE"
   Description  = "Set to not ready"
   Hidden       = "TRUE"

   [Command:NotReadyGroupSetToReady]
   SubCommand_1 = "NotReadyForPhoneSetToReady"
   Profile      = ""
   Comments     = ""
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

; ////////////Not Ready commands for Phone
[Command:NotReadyForPhoneGroup]
   SubCommand_1 = "NotReadyForPhoneSetToReady"
   SubCommand_2 = "NotReadyForPhoneSetToNotReady"
   SubCommand_3 = "NotReadyForPhoneSetToReadyNoPopup"
   Profile      = ""
   Comments     = ""
   Description  = "Change ready state for phone"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:NotReadyForPhoneSetToReady]
   CmdData       = ""
   Description   = "Phone: set to ready"
   DeviceCommand = "ChangeBusyState"
   FilterSpec    = "[$GetCommandStatus(ChangeBusyState)] = 'Checked'"
   Hidden        = "TRUE"
   Profile       = ""
   Comments      = ""
   CmdData       = ""
```

```
[Command:NotReadyForPhoneSetToNotReady]
   Profile       = ""
   FilterSpec    = "[$GetCommandStatus(ChangeReadyState)] = 'Disabled' AND
[$GetCommandStatus(ChangeNotReadyState)]= 'Enabled'"
   DeviceCommand = "ChangeBusyState"
   Comments      = ""
   Description   = "Phone: set to not ready"
   CmdData       = "NotReadyWithPopup"
   Hidden        = "TRUE"

   [CmdData:NotReadyWithPopup]
      Param.ReasonCode = "[Value]"
      SelectBusComp    = "List Of Values"
      SelectParam      = "TRUE"
      Comments         = ""
      SelectBusObj     = "List Of Values"
      SelectApplet     = "Transfer Multiple LOV Popup Applet"
      SelectTitle      = "Please select the reason for changing status to Not-Ready"
      SelectQuerySpec  = "[Type] = 'REASON_CODE' AND [Active] = 'Y'"

[Command:NotReadyForPhoneSetToReadyNoPopup]
   Profile       = ""
   FilterSpec    = "[$GetCommandStatus(ChangeReadyState)] = 'Enabled' AND
[$GetCommandStatus(ChangeNotReadyState)]= 'Enabled'"
   DeviceCommand = "ChangeBusyState"
   Comments      = ""
   Description   = "Phone: set to not ready"
   CmdData       = "NotReadyNoPopup"
   Hidden        = "TRUE"

   [CmdData:NotReadyNoPopup]
      Param.ReasonCode = "[Value]"
      Comments         = ""
```

Example 3-8 Sample for Configuring Switch Between Multiple Calls Using Siebel WorkItemList Control and Cisco CTI Driver

```
[Command:WorkItemList]
   SubCommand_1 = "SuspendDeselectedCall"
   ExecuteAll   = "TRUE"
   CmdData      = "WorkItemList"
   SubCommand_2 = "ResumeSelectedCall"
   Hidden       = "TRUE"

[Command:SuspendDeselectedCall]
   DeviceCommand = "SuspendDeselectedCall"
   FilterSpec    = "[@DeselectedWorkItem:WorkState] = 'Active'"
   CmdData       = "SuspendDeselectedCall"
   Hidden        = "TRUE"

   [CmdData:SuspendDeselectedCall]
      Param.TrackingID = "{@DeselectedWorkItem:DriverWorkTrackID}"

[Command:ResumeSelectedCall]
   FilterSpec    = "[@SelectedWorkItem:WorkState] = 'Suspended'"
   DeviceCommand = "ResumeSelectedCall"
   CmdData       = "ResumeSelectedCall"
   Hidden        = "TRUE"

   [CmdData:ResumeSelectedCall]
      Param.TrackingID = "{@SelectedWorkItem:DriverWorkTrackID}"
```

- The ResumeSelectedCall and SuspendDeselectedCall commands should be used as a sub-command only in the context of WorkItemList Siebel control.
- The ResumeSelectedCall and SuspendDeselectedCall commands are enabled/available only if there is more than one active/connected call.
- The best way to determine a trackingID, the required parameter for these commands, is to use the @DeselectedWorkItem and @SelectedWorkItem Siebel macros, as shown in Example 3-8 (attribute is DeviceWorkTrackID).

The following table lists available values to use with command parameters.

Note All values are string type and not case-sensitive; for example, using "unspecified" or "Unspecified" is allowed.

Table 3-8 Valid Values for Command Parameters

Command Parameter	Valid Values
AgentWorkMode	Unspecified (default) AutoIn ManualIn
AnsweringMachine	Unspecified (default) Connect Disconnect None NoneNoModem ConnectNoModem
CallMannerType	Unspecified (default) Polite Belligerent SemiPolite Reserved
CallOption	Unspecified (default) CallingAgentOnline CallingAgentReserved CallingAgentNotReserved CallingAgentBuzzBase CallingAgentBeepHset ServiceCircuitOn
CallPlacementType	Unspecified (default) LineCall Outbound OutboundNoAccessCode DirectPosition DirectAgent SupervisorAssist
FacilityType	Unspecified (default) TrunkGroup SkillGroup

Table 3-8 Valid Values for Command Parameters (continued)

Command Parameter	Valid Values
PostRoute	False (default) True
Priority	False (default) True

Client Tracing Control

Table 3-9 describes the registry values used to control client tracing. Add these values to the HKLM\SOFTWARE\Cisco Systems, Inc.\CTIOS Tracing registry key.

Note Restart any clients after adding the registry values. Once registry values are created they can be modified at any time.

Table 3-9 Client Tracing Control Registry Values

Registry Value	Description
MaxFileSizeKb	Maximum size of a single trace file. (Default is 2048.)
MaxFiles	Maximum number of trace files before trace server starts deleting the oldest one. (Default is 4.)
MaxDaysBeforeExpire	Maximum number of days before the log file is rolled over to a new file regardless of size. (Default is 1.)
FlushIntervalSeconds	Number of seconds before trace server flushes output to the log file. (Default is 30.)

Note These registry settings are optional and if omitted, the Trace Server uses the default settings.

Outbound Option Support

The Cisco Outbound Option (formerly known as Cisco Blended Agent) is supported with the Cisco Driver for Siebel 7 starting the ICM Release 7.0(0).

A set of sample DEF files demonstrate how to provide access to the Outbound Option related functionality from the Siebel desktop. The Outbound Option has a special set of ECC variables to ferry requests and responses between agent desktop and the Dialer.

The Driver provides set of Device commands to generate Outbound Option responses. The following table lists the Siebel Device Commands that are exclusively for the Outbound Option.

Siebel Command	Arguments Required
OOAcceptCall	BAResponse
OOSkipCall	BAResponse
OORejectCall	BAResponse
OOCallback	BAResponse
OOSkipNext	BAResponse
OOVoice	BAResponse
OOFax	BAResponse
OOAnsMach	BAResponse
OOInvalid	BAResponse
OORejectClose	BAResponse
OOSkipClose	BAResponse

BAResponse is the required OO ECC variable that has to be sent along with the OO request. Any other Call or ECC variables can be sent along with it.

An Outbound Option configuration can have the following modes:

1. Progressive
2. Predictive
3. Preview
4. Direct Preview

Specifically, there are 3 sample DEF files; one that is used for Predictive or the Progressive mode (since the functionality for both modes is identical from the agent perspective), a second for Preview mode and a third DEF file for Direct Preview mode. These DEF files incorporate the Outbound Option functionality in the form of menu options on the Siebel Communications Menu.

When imported into a Siebel configuration for a chosen mode, the sample DEF file provides support for handling Outbound related requests and appropriate event handling. For a sample for the OO commands in the DEF file, refer to the OO DEF files samples that provided on Driver CD.

Example 3-9 Sample for the usage of the OO commands via the DEF file

```
[Command:OutboundOptionGroupInMenu]
   SubCommand_1  = "Accept"
   SubCommand_2  = "Reject"
   SubCommand_3  = "RejectClose"
   SubCommand_4  = "Skip"
   SubCommand_5  = "SkipClose"
   SubCommand_6  = "Voice"
   SubCommand_7  = "Fax"
   SubCommand_8  = "AnsMach"
   SubCommand_9  = "Invalid"
   SubCommand_10 = "CallBack"
   Profile       = ""
   MenuPosition  = "0"
   Comments      = ""
   Title         = "Outbound Option"
   Description   = "Outbound Option"
   CmdData       = ""

[Command:Accept]
   Description       = "Accept Preview Call"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOAcceptCall"
   MenuPosition  = "0.1"
   Comments      = ""
   HotKey        = "Ctrl+F11"
   CmdData       = "AcceptData"

   [CmdData:AcceptData]
      Comments            = ""
      Param.BAResponse = "Accept"

[Command:Reject]
   Description       = "Reject Preview Call"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OORejectCall"
   MenuPosition  = "0.2"
   Comments      = ""
   CmdData       = "RejectData"

   [CmdData:RejectData]
      Comments            = ""
      Param.BAResponse = "Reject"

[Command:RejectClose]
   Description       = "Rejects and Close Preview Call"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OORejectClose"
   MenuPosition  = "0.3"
   Comments      = ""
   CmdData       = "RejectCloseData"

   [CmdData:RejectCloseData]
      Comments            = ""
      Param.BAResponse = "RejectClose"

[Command:Skip]
   Description       = "Skip Preview Call"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOSkipCall"
   MenuPosition  = "0.4"
   Comments      = ""
   CmdData       = "SkipData"

   [CmdData:SkipData]
      Comments            = ""
      Param.BAResponse = "Skip"

[Command:SkipClose]
   Description       = "Skip and Close Preview Call"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOSkipClose"
   MenuPosition  = "0.5"
   Comments      = ""
   CmdData       = "SkipCloseData"
   [CmdData:SkipCloseData]
      Comments            = ""
      Param.BAResponse = "SkipClose"

[Command:Voice]
   Description       = "Reclassify as Voice"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOVoice"
   MenuPosition  = "0.6"
   Comments      = ""
   CmdData       = "VoiceData"

   [CmdData:VoiceData]
      Comments            = ""
      Param.BAResponse = "REX_VOICE"

[Command:Fax]
   Description       = "Reclassify as Fax"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOFax"
   MenuPosition  = "0.7"
   Comments      = ""
   CmdData       = "FaxData"

   [CmdData:FaxData]
      Comments            = ""
      Param.BAResponse = "REX_FAX"

[Command:AnsMach]
   Description       = "Reclassify as Ans Machine"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOAnsMach"
   MenuPosition  = "0.8"
   Comments      = ""
   CmdData       = "AnsMachData"

   [CmdData:AnsMachData]
      Comments            = ""
      Param.BAResponse = "REX_ANS_MACHINE"

[Command:Invalid]
   Description       = "Reclassify as Invalid"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOInvalid"
   MenuPosition  = "0.9"
   Comments      = ""
   CmdData       = "InvalidData"

   [CmdData:InvalidData]
      Comments            = ""
      Param.BAResponse = "REX_INVALID"

[Command:CallBack]
   Description       = "Schedule Callback"
   CmdChannelOnFocus = "TRUE"
   Profile       = ""
   DeviceCommand = "OOCallback"
   MenuPosition  = "0.10"
   Comments      = ""
   CmdData       = "CallbackData"
```

```
    [CmdData:CallbackData]
       Comments           = ""
       Param.BAResponse = "Callback 06022005 10:00"
     ;// Change the value of the timestamp in the line to meet you application
     ;//needs. It is hard coded here but possible variations:
         ;// Indicates to the dialer that should schedule a callback on the date and time
specified                        ;Param.BAResponse = "Callback MMddyyyy HH:mm"
         ;//Indicates to the dialer that it should cancel a scheduled callback
         ;Param.BAResponse = "Callback Cancel"
    ;//Indicates to the dialer to change the number to dial when the callback schedule
;Param.BAResponse = "P#" + "callbackPhoneNumber"
```

Supported Combinations

Following are the supported versions of the ICM and and the CTI OS Releases in combination with the Siebel versions:

- ICM 6.0 and CTI OS 6.0, Siebel versions 7.5.3 and 7.7
- ICM 7.0 and CTI OS 7.0, Siebel versions 7.5.3 and 7.7

Note Support for the Outbound Option feature for Siebel customers is limited to Avaya and IPCC.

For additional information about the Outbound Option functioanlity with Siebel, refer to the *Outbound Option User Guide for Cisco ICM/IPCC Enterprise & IPCC Hosted Editions* and *Outbound Option Setup and Configuration Guide for Cisco ICM/IPCC Enterprise & IPCC Hosted Editions.*

CCM Silent Monitor Support

In 7.2(1), CTI OS supports Call Manager 6.0's silent monitor implementation, called CCM Silent Monitor. With this type of Silent Monitor configured in CTI OS Server, Siebel agents can be silent monitored by the supervisor that uses a standard CTI OS Supervisor Desktop. Please note the following requirements, versions and limitations for this functionality:

- Siebel agent desktop will not receive any notification that call is being monitored.
- Agent desktop does not see call events related to the silent monitor call.
- Siebel agent should use a hard phone in addition to Siebel CTI. Required phone types are listed below.
- CTIOS Silent Monitor type is not supported with Siebel agents.
- CTI OS Server has to be configured for CCM Silent Monitor type. It can be done by running CTI OS Server setup. The following registry key indicates that CCM Silent Monitor type is configured:
  - HKKLM\SOFTWARE\Cisco Systems, Inc.\Ctios_<*Instance*>\CTIOS1\EnterpriseDesktopSettings\All Desktops\IPCCSilentMonitor\Name\Settings\CCMBasedSilentMonitor
  - This field is used to toggle the silent monitor type:

    0 configures CTI OS-based silent monitor

    1 configures CCM-based silent monitor

- CCM Silent Monitor is supported for the following versions
  - Call Manager: Version 6.0 or higher
  - IPCC: Version 7.2(1) or higher
  - Supervisor Desktop: CTIOS Supervisor Desktop Application Version 7.2 (1)

    Type: C++ , Java, and .NET; No device (hard phone) required on Supervisor desktop
  - Agent Desktop: Siebel CTI with Siebel Driver version 7.0 or higher
  - Agent Desktop Device: Type 79x1 Cisco phone (7941, 7961, or 7971)
- Because silent monitor calls are estimated to be twice as expensive as standard call, using CCM silent monitor reduces the total BHCA that can be supported by IPCC. The impact to performance is listed in the *Hardware & System Software Spec. (Bill of Materials) for Cisco ICM/IPCC Enterprise & Hosted Editions* located at: http://www.cisco.com/en/US/products/sw/custcosw/ps1001/products_user_guide_list.html.
- Refer to the following guides to get more detail information on CCM Silent Monitor configuration and troubleshooting:
  - CTI OS System Manager's guide
  - CTI OS Supervisor Desktop Guide
  - CTI OS Developer's Guide
  - CTI OS Troubleshooting Guide

CHAPTER 4

Broadcast Statistics

This chapter discusses the broadcast statistics used with the CTI Driver for Siebel 7.

Introduction

In general, broadcast statistics functionality provides agent and skill group statistics from a CTI OS Server with a broadcast display in Siebel.

Broadcast statistics include the following:

- Manage enabling/disabling agent and/or skill group statistics using a DEF file
- Configuring statistics required for broadcast
- Dynamically associating/disassociating the agent with the broadcast by creating/removing a special position name only for broadcast functionality and adding it to a recipient list for a corresponding broadcast message
- Clean all broadcast association when the agent logs out or when the Siebel application is closed.

Note

- Broadcast statistics can only be achieved by using a combination of the Cisco Driver Device commands and events along with Siebel scripts.
- The current Driver version supports skill group statistics if the Driver only talks to one CTI OS Server (one peripheral).
- Detection of Skill Group Numbers can be configured.

Configuring Broadcast Statistics

There are many agent and skill group statistics available (see list of available agent and skill group statistics, below) for broadcasting purposes.

Note Every Siebel employee belongs to a particular configuration.

Statistics to be broadcasted to Siebel agents or a group of agents can be configured in two ways:

- Enabling the broadcast message (agent or skill group statistics) for an agent or for all agents under a particular configuration. This can be done using the following parameters:

- Driver:BroadcastStatistics (enable/disable broadcast on a driver level).
- Param.SkillGroupBroadcastStat and Param.AgentBroadcastStat (enable/disable broadcast agent level) in Login Device command. (See Appendix A, “Business Service Script,” for more details.)

Note The *CTI OS System Manager's Guide for Cisco ICM/IPCC Enterprise & Hosted Editions* discusses how to configure statistics on the CTI OS Server site.

- Specifying particular statistics for broadcast within a DEF file, so not all statistics that come from the CTI OS Server OnAgentSatistics/SkillGroupStatistics events will be broadcasted to Siebel. This can be done using the Service parameters with the "AgentStat" prefix for agent statistics and the Driver parameters with the "SkillGrpStat" prefix for skill group statistics. For example:

```
Driver:SkillGrpStat1            = "NotReadyTimeToday"
Driver:SkillGrpStat2            = "HandledCallsToday"
Driver:SkillGrpStat3            = "AvailTimeToday"

Service:AgentStat1                = "AvailTimeToday"
Service:AgentStat2                = "LoggedOnTimeToday"
```

Note If none of these parameters (AgentStat, SkillGrpStat) are specified in a DEF file, then all statistics configured on the CTI OS Server will be broadcasted to Siebel. If these parameters are specified, then an additional filtering is done in the CTI Driver for Siebel 7 and only those statistics will appear on broadcast messages in Siebel.

Agent Statistics

The following agent statistics (for Protocol 8) can be used to configure agent broadcast messages.

Refer to the *Cisco ICM Software CTI Server Message Reference Guide (Protocol Version 9)* for more details.

- AvailTimeSession
- LoggedOnTimeSession
- NotReadyTimeSession
- AgentOutCallsSession
- AgentOutCallsTalkTimeSession
- AgentOutCallsTimeSession
- AgentOutCallsHeldSession
- AgentOutCallsHeldTimeSession
- HandledCallsSession
- HandledCallsTalkTimeSession
- HandledCallsAfterCallTimeSession
- HandledCallsTimeSession
- IncomingCallsHeldSession
- IncomingCallsHeldTimeSession

- InternalCallsSession
- InternalCallsTimeSession
- InternalCallsRcvdSession
- InternalCallsRcvdTimeSession
- InternalCallsHeldSession
- InternalCallsHeldTimeSession
- AvailTimeToday
- LoggedOnTimeToday
- NotReadyTimeToday
- AgentOutCallsToday
- AgentOutCallsTalkTimeToday
- AgentOutCallsTimeToday
- AgentOutCallsHeldToday
- AgentOutCallsHeldTimeToday
- HandledCallsToday
- HandledCallsTalkTimeToday
- HandledCallsAfterCallTimeToday
- HandledCallsTimeToday
- IncomingCallsHeldToday
- IncomingCallsHeldTimeToday
- InternalCallsToday
- InternalCallsTimeToday
- InternalCallsRcvdToday
- InternalCallsRcvdTimeToday
- InternalCallsHeldToday
- InternalCallsHeldTimeToday
- AvgTalkTimeToday
- CallsHandledToday
- TimeLoggedInToday
- TimeTalkingToday
- TimeHoldingToday
- TimeReadyToday
- TimeNotReadyToday
- AvgHoldTimeToday
- AvgHandleTimeToday
- AvgIdleTimeToday
- PercentUtilizationToday

Skill Group Statistics

The following skill group statistics (for Protocol 8) can be used to configure skill group broadcast messages.

Refer to the *Cisco ICM Software CTI Server Message Reference Guide (Protocol Version 9)* for more details.

- CallsQNow
- CallsQTimeNow
- LongestCallQNow
- AvailTimeToHalf
- LoggedOnTimeToHalf
- NotReadyTimeToHalf
- AgentOutCallsToHalf
- AgentOutCallsTalkTimeToHalf
- AgentOutCallsTimeToHalf
- AgentOutCallsHeldToHalf
- AgentOutCallsHeldTimeToHalf
- HandledCallsToHalf
- HandledCallsTalkTimeToHalf
- HandledCallsAfterCallTimeToHalf
- HandledCallsTimeToHalf
- IncomingCallsHeldToHalf
- IncomingCallsHeldTimeToHalf
- InternalCallsRcvdToHalf
- InternalCallsRcvdTimeToHalf
- InternalCallsHeldToHalf
- InternalCallsHeldTimeToHalf
- CallsQHalf
- CallsQTimeHalf
- LongestCallQHalf
- AvailTimeToday
- LoggedOnTimeToday
- NotReadyTimeToday
- AgentOutCallsToday
- AgentOutCallsTalkTimeToday
- AgentOutCallsTimeToday
- AgentOutCallsHeldToday
- AgentOutCallsHeldTimeToday

- HandledCallsToday
- HandledCallsTalkTimeToday
- HandledCallsAfterCallTimeToday
- HandledCallsTimeToday
- IncomingCallsHeldToday
- IncomingCallsHeldTimeToday
- InternalCallsRcvdToday
- InternalCallsRcvdTimeToday
- InternalCallsHeldToday
- InternalCallsHeldTimeToday
- CallsQToday
- CallsQTimeToday
- LongestCallQToday
- AvgCallsQTimeNow
- AvgAgentOutCallsTalkTimeToHalf
- AvgAgentOutCallsTimeToHalf
- AvgAgentOutCallsHeldTimeToHalf
- AvgHandledCallsTalkTimeToHalf
- AvgHandledCallsAfterCallTimeToHalf
- AvgHandledCallsTimeToHalf
- AvgIncomingCallsHeldTimeToHalf
- AvgInternalCallsRcvdTimeToHalf
- AvgInternalCallsHeldTimeToHalf
- AvgCallsQTimeHalf
- AvgAgentOutCallsTalkTimeToday
- AvgAgentOutCallsTimeToday
- AvgAgentOutCallsHeldTimeToday
- AvgHandledCallsTalkTimeToday
- AvgHandledCallsAfterCallTimeToday
- AvgHandledCallsTimeToday
- AvgIncomingCallsHeldTimeToday
- AvgInternalCallsRcvdTimeToday
- AvgInternalCallsHeldTimeToday
- AvgCallsQTimeToday
- MeasuredCallQTime

The following skill group statistics were added in Protocol 8:

- AgentsLoggedOn
- AgentsNotReady

- AgentsReady
- AgentsTalkingIn
- AgentsTalkingOut
- AgentsTalkingOther
- AgentsWorkNotReady
- AgentsWorkReady
- AgentsBusyOther
- AgentsReserved
- AgentsHold
- RouterCallsQNow
- LongestRouterCallQNow

Detection of Skill Group Numbers

Let us assume that the Siebel Agent has ACD Queues configured via **Administration-Communications->Agent General Profile -> ACD Queues** tab and the DEF file has the following entry for Login command data:

```
Command:LoginToPBXDevice]
   Profile       = ""
   DeviceCommand = "LogIn"
   Comments      = ""
   CmdData       = "LoginToPBXDevice"
   Hidden        = "TRUE"

   [CmdData:LoginToPBXDevice]
      Param.AgentWorkMode           = "0"
      Param.UserId                  = "{@UserName}"
      Comments                      = ""
      Param.SkillGroupBroadcastStat = "TRUE"
      Param.Password                = "{@AgentPin}"
      Param.ACDQueue                = "{@QueueList}"
      Param.AgentId                 = "{@AgentId}"
```

Then the Siebel Driver will start the Skill Group Statistics Broadcast for all the ACD Queue values configured, treating them as the SkillGroup Number that the agent belongs to. This method can be used if the agent statically belongs to a particular skill group set, and the association doesn't change dynamically. The same method can also be used on some of the skill groups as a sub-mask for a Broadcast Skill Group Statistics.

Otherwise (if nothing is configured in the ACD Queue), the Siebel Driver automatically uses the OnQueryAgentStateConf event to detect ALL skill groups that the agent belongs to, on Login, and start the Skill Group Statistics Broadcast for all skill groups.

Business Services

A new business service needs to be created to encapsulate broadcast-related methods.

A business service can be created using the Siebel tools and compiling it into a Siebel repository, or by saving this object directly to the Siebel database. This business service will implement a number of methods that are critical in order to make a complete Broadcast Statistics solution work as expected.

How to create a Siebel business service for message broadcast

The following instructions describe how to create a Siebel business service for a message broadcast without requiring a new Siebel repository.

Note If the Siebel tools are used, the Siebel repository must be recompiled.

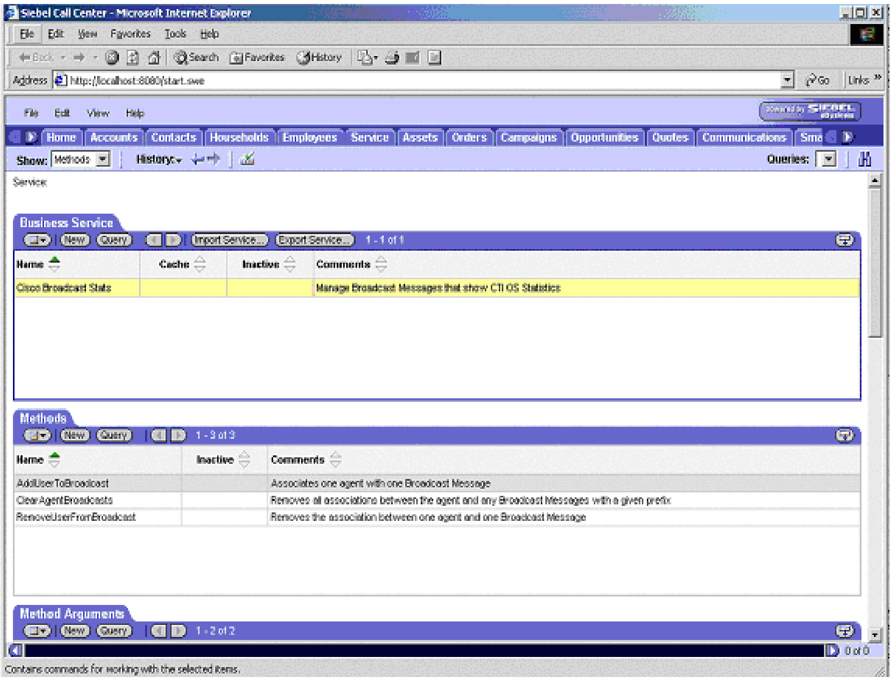
Step 1 Log in to the Siebel Client Application as a Siebel Administrator.

Step 2 Create a Siebel Business Object for the message broadcast by selecting **View > Site Map > Business Service Administration > Business Service Details**.

Step 3 In the Business Service menu, clicking the **Import Service** button and browse to the CiscoBroadcastStats.xml file found in the Siebel7/Documentation directory or the Siebel7.5/Documentation directory.

The following business service methods are created with corresponding arguments and scripts:

- AddUserToBroadcast
- ClearAgentBroadcasts
- RemoveUserFromBroadcast



Siebel Call Center - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost:8080/start.swe
Go Links
File Edit View Help
Home Accounts Contacts Households Employees Service Assets Orders Campaigns Opportunities Quotes Communications Sma
Show: Methods
History
Queries:
Service:
Business Service
New Query Import Service... Export Service... 1 - 1 of 1

Name	Cache	Inactive	Comments
Cisco Broadcast Stats			Manage Broadcast Messages that show CTI OS Statistics

Methods
New Query 1 - 3 of 3

Name	Inactive	Comments
AddUserToBroadcast		Associates one agent with one Broadcast Message
ClearAgentBroadcasts		Removes all associations between the agent and any Broadcast Messages with a given prefix
RemoveUserFromBroadcast		Removes the association between one agent and one Broadcast Message

Contains commands for working with the selected items.

This new business service is successfully entered in the Siebel database and can be used with the DEF file as explained in the "Invoking Service Methods" section.

Invoking Service Methods

This section describes each service method that will be invoked using a DEF file.

AddUserToBroadcast Function *addUserToBroadcast(strUserName, strMsgAbstract*)

This service method must be used in the Login command to associate the employee having the given Siebel login name with the broadcast message used to send statistics for the given agent group or skill group.

```
function AddUserToBroadcast(strUserName, strMsgAbstract)
{
        var strPositionId  = FindPositionId(strUserName);
        if (strPositionId == "")
    {
        ShowStatusText("Can't find or create Position for login name " + strUserName, true);
        return;
    }
        var boBroadcastMessage = null;
    var bcBroadcastMessage = null;

    GetBroadcastMsg(strMsgAbstract, boBroadcastMessage, bcBroadcastMessage);

    if (bcBroadcastMessage == null)
    {
        ShowStatusText("Unable to find or create Broadcast Message " + strMsgAbstract, true);
        return null;
    }

    if (AssociateChildRecord(bcBroadcastMessage, "Position Id", "Id", strPositionId) == "")
    {
        ShowStatusText("Unable to associate Position with Broadcast Message", true);
    }

    bcBroadcastMessage = null;
    boBroadcastMessage = null;
}
```

Example 4-1 Sample DEF File for Invoking AddUserToBroadcast Service Method

```
[Command:LoginToPBX]
   SubCommand_1 = "LoginToPBXDevice"
   SubCommand_2 = "InsertAgentStatBroadcast"
   Profile      = ""
   FilterSpec   = "[$GetCommandStatus(LoginToPBXDevice)]='Enabled'"
   MenuPosition = "20.1"
   Comments     = ""
   Title        = "Log In (Phone)"
   ExecuteAll   = "TRUE"
   CmdData      = ""

 [Command:InsertAgentStatBroadcast]
   Profile       = ""
   Comments      = "Add a Broadcast Message record for this agent/position"
   CmdData       = "InsertAgentStatBroadcast"
   ServiceMethod = "Cisco Broadcast Stats.AddUserToBroadcast"
   Hidden        = "TRUE"
```

```
[CmdData:InsertAgentStatBroadcast]
   Comments              = ""
   ServiceParam.UserName = "{@UserName}"
   ServiceParam.Abstract = CTIAgentStat_{@UserName}
```

ClearAgentBroadcasts Function *ClearAgentBroadcasts(strUserName, strPrefix)*

This service method must be used in the Login command to remove any associations between the agent having the given Siebel login name and all broadcast messages whose abstracts begin with “strPrefix.”

Note Use “CTISkillStat_” as the prefix for skill group messages, “CTIAgentStat_” as the prefix for agent messages, and “CTI*Stat_” as the prefix to match both.

This service method should be called when the agent first logs into Siebel, so that the agent will initially not receive any skill group statistics.

```
function ClearAgentBroadcasts(strUserName, strPrefix)
{
    // Find the Position used to send statistics to this agent.

    var strPositionId = FindPositionId(strUserName);

    if (strPositionId == "")
    {
        ShowStatusText("Can't find or create Position for login name " + strUserName, true);
        return;
    }

    // Iterate through all Broadcast Messages starting with
    // the prefix strPrefix and remove any association between this agent's
    // Position and those messages.

    var boBroadcastMessage = TheApplication().GetBusObject("Broadcast Message");

    if (boBroadcastMessage == null)
    {
        ShowStatusText("Unable to create Broadcast Message business object", true);
        return null;
    }

    var bcBroadcastMessage=boBroadcastMessage.GetBusComp("Broadcast Message");

    if (bcBroadcastMessage == null)
    {
        boBroadcastMessage = null;
        ShowStatusText("Unable to create Broadcast Message business component", true);
        return null;
    }

    with (bcBroadcastMessage)
    {
        ClearToQuery();
        SetViewMode( AllView );
        ActivateField("Abstract");
        ActivateField("Position Id");

        SetSearchSpec("Abstract", "LIKE '" + strPrefix + "*'");
        ExecuteQuery( ForwardOnly );
     }


    var hasRecord = bcBroadcastMessage.FirstRecord();
    while (hasRecord)
    {
        var bcMVG = bcBroadcastMessage.GetMVGBusComp("Position Id");

        if (bcMVG == null)
        {
            ShowStatusText("Unable to create MVG BusComp for Position Id", true);
            bcBroadcastMessage = null;
            boBroadcastMessage = null;
            return;
        }

        // Search through the existing Position associations with
        // this Broadcast Message for one whose Row Id
        // matches the agent's statistics Position Id.  If one
        // is found, remove it.

        bcMVG.SetSearchSpec("Id", strPositionId);
        bcMVG.ExecuteQuery( ForwardOnly );

        while (bcMVG.FirstRecord())
        {
            bcMVG.DeleteRecord();
        }

        bcMVG = null;

        hasRecord = bcBroadcastMessage.NextRecord();
    }

    bcBroadcastMessage = null;
    boBroadcastMessage = null;
}
```

Example 4-2 Sample DEF File for Invoking ClearAgentBroadcasts Service Method

```
[Command:LoginToPBX]
   SubCommand_1 = "LoginToPBXDevice"
   SubCommand_2 = "InsertAgentStatBroadcast"
   SubCommand_3 = "ClearAgentSkillGroups"
   Profile      = ""
   FilterSpec   = "[$GetCommandStatus(LoginToPBXDevice)]='Enabled'"
   MenuPosition = "20.1"
   Comments     = ""
   Title        = "Log In (Phone)"
   ExecuteAll   = "TRUE"
   CmdData      = ""

[Command:ClearAgentSkillGroups]
   Profile       = ""
   Comments      = "Add a Broadcast Message record for this agent/position"
   CmdData       = "ClearAgentSkillGroups"
   ServiceMethod = "Cisco Broadcast Stats.ClearAgentBroadcasts"
   Hidden        = "TRUE"

   [CmdData:ClearAgentSkillGroups]
      Comments              = ""
      ServiceParam.UserName = "{@UserName}"
      ServiceParam.Prefix    = "CTISkillStat_"
```

RemoveUserFromBroadcast Function *RemoveUserFromBroadcast(strUserName, strMsgAbstract)*

This service method must be used to remove any association between the employee having the given Siebel login name and the broadcast message used to send agent or skill group statistics based on the strMsgAbstract parameter that will specify the broadcast message.

This service method has to be called on the OnEventSkillGroupRemove event to remove an association between an agent and skill group statistics broadcast message and as a subcommand for the Logout command to remove an association between an agent and an agent statsitics broadcast message.

```
function RemoveUserFromBroadcast(strUserName, strMsgAbstract)
{
    // RemoveUserFromBroadcast:
    //
    // Removes any association between the Employee having the given
    // Siebel login name and the Broadcast Message used to send
    // agent or skill group statistics.

    // Find the Position used to send statistics to this user.

    var strPositionId = FindPositionId(strUserName);

    if (strPositionId == "")
    {
        ShowStatusText("Can't find or create Position for login name " + strUserName, true);
        return;
    }

    // Find the Broadcast Message

    var boBroadcastMessage = null;
    var bcBroadcastMessage = null;

    GetBroadcastMsg(strMsgAbstract, boBroadcastMessage, bcBroadcastMessage);

    if (bcBroadcastMessage == null) {
        ShowStatusText("Unable to find or create Broadcast Message " + strMsgAbstract, true);
        return null;
    }

    var bcMVG = bcBroadcastMessage.GetMVGBusComp("Position Id");

    if (bcMVG == null)
    {
        ShowStatusText("Unable to create MVG BusComp for Position Id", true);
        bcBroadcastMessage = null;
        boBroadcastMessage = null;
        return;
    }

    // Search through the existing Position associations with
    // this Broadcast Message for one whose Row Id
    // matches the user's statistics Position Id.  If one
    // is found, remove it.

    bcMVG.SetSearchSpec("Id", strPositionId);
    bcMVG.ExecuteQuery( ForwardOnly );
    while (bcMVG.FirstRecord())
    {
        bcMVG.DeleteRecord();
    }

    bcMVG = null;
    bcBroadcastMessage = null;
    boBroadcastMessage = null;
}
```

Example 4-3 Sample DEF File for Invoking RemoveUserFromBroadcast Service Method for Agent Statistics

```
[Command:LogoutFromPBX]
   SubCommand_1 = "LogoutFromPBXDevice"
   SubCommand_2 = "RemoveAgentStatBroadcast"
   Profile      = ""
   FilterSpec   = "[$GetCommandStatus(LogoutFromPBXDevice)]='Enabled'"
   MenuPosition = "22.1"
   Comments     = ""
   Title         = "Log Out (Phone)"
   ExecuteAll   = "TRUE"
   CmdData      = ""

[Command:LogoutFromPBXDevice]
   Profile       = ""
   DeviceCommand = "LogOut"
   Comments      = ""
   CmdData       = "LogoutFromPBXDevice"

   [CmdData:LogoutFromPBXDevice]
      Param.ReasonCode = "1"
      Comments         = ""

[Command:RemoveAgentStatBroadcast]
   Profile       = ""
   Comments      = "Remove association Broadcast Message record for this agent/position"
   CmdData       = "RemoveAgentStatBroadcast"
   ServiceMethod = "Cisco Broadcast Stats.RemoveUserFromBroadcast"
   Hidden        = "TRUE"

   [CmdData:RemoveAgentStatBroadcast]
      Comments              = ""
      ServiceParam.UserName = "{@UserName}"
      ServiceParam.Abstract = CTIAgentStat_{@UserName}
```

Example 4-4 Sample DEF File for Invoking RemoveUserFromBroadcast Service Method for Skill Group Statistics

```
[EventHandler:EventSkillGroupRemove]
   Filter.SkillGroupNumber = "?*"
   Profile     = ""
   Comments    = ""
   Order       = "1"
   Response    = "OnEventSkillGroupRemove"
   DeviceEvent = "EventSkillGroupRemove"

   [EventResponse:OnEventSkillGroupRemove]
      Comments              = ""
      ServiceMethod = "Cisco Broadcast Stats.RemoveUserFromBroadcast"
      ServiceParam.UserName = "{@UserName}"
      ServiceParam.Abstract = "CTISkillStat_{SkillGroupNumber}"
```

CHAPTER 5

Troubleshooting and Testing

This chapter discusses problems that might be encountered and how to deal with them, and includes switch-specific information.

Set Trace Level

- Setting trace levels: Trace levels for client processes, such as the AgentDesktop phone, can be found in the registry under **HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems, Inc.\CTIOS Tracing**. (Refer to the *CTI OS Developer's Guide for Cisco ICM/IPCC Enterprise & Hosted Editions* for more information.)

Warning **The default value for the trace masks is 0x40000307 (4 is for threading trace). Changing this value can have a serious impact on client performance. It should only be modified by experienced field personnel or at the request of Cisco support personnel.**

- Set value for TraceMask to 7ff for maximum tracing. The Trace level for clients is located in the registry in **HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems, Inc.\CTIOS Tracing**.
- Set value for TraceMask to 40000307 for thread name to be printed. The Trace level for clients is located in the registry in **HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems, Inc.\CTIOS Tracing**.
- CTI OS client logs: The trace log name and location for client processes can be found under the following registry keys: **HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems, Inc.\CTIOS Tracing\TraceFileName**. The default filename is **CtiosClientLog**. Log files are created using the convention <*TraceFileName*>.<*LogonUserName*>.mmdd.hhmmss.log. To store the files in a different location, provide a fully qualified path for the TraceFileName. For example, setting the value to “C:\Temp\CtiosClientLog” would put the log files in the directory “C:\Temp” using the naming convention CtiosClientLog.mmdd.hhmmss.log. Client trace files are simple ASCII text and can be opened with a conventional text editor, such as Notepad.

- The following instructions describe how to increase Driver log file sizes and save them for a configured amount of time, so they will not be overwritten:
  - Change the following registry values to the **HKEY_LOCAL_MACHINE\SOFTWARE\Cisco Systems, Inc.\CTIOS Tracing** registry key for additional control of client tracing.

Registry Value	Description
DWORD "MaxFileSizeKb"	Maximum size of a single trace file (default is 2048)
DWORD "MaxFiles"	Maximum number of trace files before the Trace Server starts deleting the oldest file (default is 5)
DWORD "MaxDaysBeforeExpire"	Maximum number of days before log file is rolled over to a new file regardless of size (default is 7)
DWORD "FlushIntervalSeconds"	Number of seconds before the Trace Server transfers output to the log file (default is 30)

Service:AutoLogout Parameter

The Service:AutoLogout parameter, if enabled, sends a logout request automatically every time an agent logs out of Siebel, if the AgentState is not already set to Logout. The logout request fails if the current agent state does not allow transition to the Logout state. In this case, a ControlFailureConf error displays in the Driver log as a response to the AutoLogout request.

Use the following guidelines to troubleshoot the ControlFailureConf error when using the Service:AutoLogout parameter:

- On Aspect, IPCC, and Spectrum, the agent should be in the NotReady state in order to log out; otherwise, when the Siebel application closes a ControlFailureConf error occurs.
- On most switches, the agent can not log out while in the Talking state.
- AutoLogout might fail if the AutoLogoutReasonCode is not provided or was entered incorrectly.

Note The above cases where ControlFailureConf occurs should be considered normal behavior.

Locating Siebel Client Logs

Each Siebel CTI user has a separate Siebel SCOMM log with the naming convention SCOMM_<*Siebel User Name*>.log. These logs are very useful for troubleshooting purposes.

The location of these logs depends on which type of Siebel client you are using. (Refer to the Siebel documentation set for more details.)

- Siebel Dedicated client: SCOMM logs are located in the <SiebelRoot>/web client/LOG directory.
- Siebel Thin client: SCOMM client logs are located in the <SiebelRoot>/Siebsrvr/LOG directory.

Using Siebel Macros

Using Siebel Macros on Login

If the Siebel @UserName macro is specified on the Login command, it is easier to read the Driver log because most of the lines have the Siebel user name printed in the front; otherwise, you need to search for it by AgentID/Extension.

The following is an example of the Login command that uses the Siebel macro mentioned above:

```
[Command:LoginToPBXDevice]
   Profile       = ""
   DeviceCommand = "LogIn"
   Comments      = ""
   CmdData       = "LoginToPBXDevice"
   Hidden        = "TRUE"

   [CmdData:LoginToPBXDevice]
      Param.AgentWorkMode            = "0"
      Param.UserId                   = "{@UserName}"
      Comments                       = ""
      Param.SkillGroupBroadcastStat  = "TRUE"
      Param.Password                 = "{@AgentPin}"
      Param.ACDQueue                 = "{@QueueList}"
      Param.AgentBroadcastStat       = "FALSE"
      Param.AgentId                  = "{@AgentId}"
```

The following is an example of a Driver log that uses the Siebel macro mentioned above:

```
11/25/02 16:36:44.711  1688  siebmtshmw  [DRIVER <ASP7 -- 282>] EVENT: OnCallBegin: (PeripheralID:5002
PeripheralType:1
    ConnectionCallID:789 CallType:10 ServiceNumber:-1 ServiceID:-1
    SkillGroupPriority:0 NumNamedVariables:0 NumNamedArrays:0
    CallVariable2:9782758589 CallVariable10:SiebelDataStore.Access.Fail
    RouterCallKeyDay:0 RouterCallKeyCallID:0 DeviceID:22
    UniqueObjectID:call.5002.789.22 CallStatus:65535 MessageID:eCallBeginEvent
    DeviceUniqueObjectID:device.5002.22)

11/25/02 16:36:44.881  1688  siebmtshmw  [call.5002.789.20] eCallDataUpdateEvent
11/25/02 16:36:44.891  1688  siebmtshmw  [DRIVER <ASP5 -- 280>] EVENT: OnCallDataUpdate: (PeripheralID:5002
    PeripheralType:1 ConnectionCallID:789 CallType:10 ServiceNumber:-1
    ServiceID:-1 SkillGroupID:5059 SkillGroupPriority:0 NumNamedVariables:0
    NumNamedArrays:0 CallVariable2:9782758589
    CallVariable10:SiebelDataStore.Access.Fail RouterCallKeyDay:0
    RouterCallKeyCallID:0 EnablementMask:0x180002 DeviceID:20
    UniqueObjectID:call.5002.789.20 CallStatus:1 MessageID:eCallDataUpdateEvent
    DeviceUniqueObjectID:device.5002.20)
```

Note ASP5 and ASP7 represent the Siebel user names and 280 and 282 are their Agent IDs.

Supported Switches

The Cisco Driver for Siebel 7 supports the following switches:

- IPCC
- Alcatel

- Aspect
- Avaya (DEFINITY ECS)
- Nortel Meridian
- Nortel Symposium
- Rockwell Spectrum
- Siemens HiCom (North America Only)

Switch-specific Information

Alcatel

- Requires that an agent must provide a position ID (that is, the extension of type A in the Siebel UI) during login, but an extension (that is, the extension of type S in the Siebel UI) is not required. The Siebel UI does not allow the agent to leave the extension blank. The Cisco Driver for Siebel 7 properly logs in the agent, if the agent provides the proper position ID and *any* extension that is not equal to the position ID.

Aspect

- Agents must be in the NotReady state to log out.
- Consultation calls: Place the outside line call on hold and make another inside line call. The Cisco Driver for Siebel 7 enables Siebel CTI toolbar buttons to initiate transfer/conference both ways:
  - Using Hold and MakeCall buttons (as described above): Hold the current call and make another call
  - Using TransferInit/ConferenceInit buttons
  - Blind Transfer (only available during transfer to the Call Connection Table)
- Conference calls: After the conference call is complete, the conference originator can still make calls (with a different call ID), which is treated as a single conference call. (This behavior on a soft phone is similar to the Aspect phone.)

Nortel Meridian

- One step blind transfers (<Fast Transfer>) corresponding with the DeviceCommand TransferMute are not supported for off-switch calls.

Note This is a Nortel limitation—refer to the Nortel documentation for more information.

Nortel Symposium

- Off-switch calls are not supported.

Note This is a Nortel limitation—refer to the Nortel documentation for more information.

Rockwell Spectrum

- The PositionID is required for login. This value is the Logical Workstation Number configured on an ACD for each agent. The PositionID needs to be configured for every agent in the “All Telesets” Siebel view as an extension of type “A” in addition to the normal “S” extension type.
- Agents must be in the NotReady state to log out.
- Agents must be in the NotReady state in order to make a call.
- Agents need to enter a WorkReady request while talking in order to remain in the NotReady state after the call is finished.
- Agent can receive a direct call if they are in the NotReady state, but the agent will not get a Ringing event (such as BeginCall or CallDelivered, or any others) until the call is answered (on a hard phone).
- Consultation calls: Place the outside line call on hold and make another inside line call. The Cisco Driver for Siebel 7 enables Siebel CTI toolbar buttons to initiate transfer/conference both ways:
  - Using Hold and MakeCall buttons (as described above): Hold the current call and make another call
  - Using TransferInit/ConferenceInit buttons
  - Blind Transfer is available
  - Single-step transfer is available for ACD calls
- Conference calls:
  - After the conference call is complete, the conference originator can still make calls (with a different call ID), which is treated as a single conference call. (This behavior on a soft phone is similar to the Spectrum hard phone.)
  - The conference originator (after the conference call is complete), can do the following when there are two calls present (similar to Spectrum hard phone behavior):

    Hold/Retrieve holds/retrieves both calls simultaneously (similar to a Spectrum hard phone).

    ReleaseCall releases both calls and stops the conference. To release the call, one party from the conference must put another party on Hold first (similar to Spectrum hard phone behavior).

Siemens HiCom (North America Only)

- The **Release** and **Hold/Retrieve** buttons are enabled when there is only one call on the device. The **Release** button is enabled when that call’s state is Initiating, Talking, Failed, or OnHold. The **Hold** button is enabled when that call’s state is Talking, and the **Retrieve** button is enabled when the call’s state is OnHold.
- When there are two calls at the device, including a consult call, either call can only be disconnected using the **Reconnect** button. Holding and retrieving the calls is accomplished using the **Alternate** button. To disconnect the held call, it is necessary to click **Alternate** first before clicking **Reconnect**.

- If there are two calls at the device, and the other party on the Talking call disconnects, the Held call can only be retrieved using the **Reconnect** button.
- Neither Single Step Transfer nor Single Step Conference are supported; however, Blind Transfer (completing the transfer before the consulted agent answers) is supported.
- The SendDTMFSignal request is only valid for trunk calls and will fail if called using an inside call.
- If an agent has placed another agent on hold, the second agent can not initiate a transfer or a conference until the first agent retrieves the call. For example:
  - Agent1 is talking to Agent2 and Agent1 clicks **Hold**.
  - Agent2's Transfer_Init and Conference_Init buttons are disabled until Agent1 clicks **Retrieve**. (Agent1's Transfer_Init and Conference_Init buttons are also disabled because these buttons are not allowed for a held call.) When Agent1 clicks **Retrieve**, the **Transfer_Init** and **Conference_Init** buttons will re-enable on Agent1's and Agent2's softphone.

Troubleshooting Problems

Symptom No screen context transfer occurs during transfer/conference call

Possible Cause This problem has no connection to Cisco Data Store and is not an ACD-specific problem. The driver log indicates that the connection to Cisco Data Store was not established successfully in the first place (from the very beginning of the log):

```
10/26/02 15:24:08.152  2244  siebmtshmw  [DRIVER] CTIDriver::SetSocketCD: Set CDS for
internal driver object to 0x5c11d10
10/26/02 15:24:08.152  2244  siebmtshmw  [DRIVER] CTIDriver::SetSocketCD: DataStoreName
specified via Driver
    Configuration parameters: R12KSA01
10/26/02 15:24:08.152  2244  siebmtshmw  Attempting to resolve hostname [R12KSA01] to IP
address (gethostbyname).
10/26/02 15:24:08.152  2244  siebmtshmw  ** Attempting to connect to server [R12KSA01
(172.18.20.253) port:42029] **
10/26/02 15:24:09.214  2244  siebmtshmw  CNetPort(05C141C0)::Open, WARNING,, Connection to
Host[], Target machine
    refused connection. (WSAECONNREFUSED), Error Number(10061).
10/26/02 15:24:09.214  2244  siebmtshmw  CCilConnection(05C14458)::OpenConnection, FAILED
10/26/02 15:24:09.214  2244  siebmtshmw  Connection disconnected, host [R12KSA01].
10/26/02 15:24:14.214  2244  siebmtshmw  [DRIVER] CTIDriver::CreateISCDriverInstance.
Connected to Cisco Siebel Data
    Store, Failure[Connection Failed]
```

This indicates that the R12KSA01 host on port 42029 was not accessible, so nobody can use it.

There are two possible reasons why this happens:

- The Driver parameters in a DEF file are missing the DataStore host name and port number (where it listens). Configure these using the following Driver parameters:

  Driver:DataServerPort = "42029"
  Driver:DataServerName = "R12KSA01"

Note This data displays during a connection attempt.

- Cisco Data Store is not installed and running. Possibly the old Cisco Data Store installation did not require it to be running and any first time connection would start Cisco Data Store for the rest of the users. With a socket Cisco Data Store installation, it must be running prior to any user trying to establish a connection. Also, every Cisco Data Store user needs to verify if a connection is alive and if not it is not, it must be reconnected on every transfer attempt if they need to retrieve data from Cisco Data Store.

  The following might happen in a log:

```
10/26/02 15:31:15.816  2244  siebmtshmw  [DRIVER] *** Detected FAILURE for CiscoDataStore
connection, connecting ...
10/26/02 15:31:15.816  2244  siebmtshmw  Attempting to resolve hostname [R12KSA01] to IP
address (gethostbyname).
10/26/02 15:31:15.816  2244  siebmtshmw  [call.5000.314.805] eCallDataUpdateEvent
10/26/02 15:31:15.816  2244  siebmtshmw  ** Attempting to connect to server [R12KSA01
(172.18.20.253) port:42029] **
....
10/26/02 15:31:16.878  2244  siebmtshmw  CNetPort(05C141C0)::Open, WARNING,, Connection to
Host[], Target machine
    refused connection. (WSAECONNREFUSED), Error Number(10061).
10/26/02 15:31:16.878  2244  siebmtshmw  CCilConnection(05C14458)::OpenConnection, FAILED
10/26/02 15:31:16.878  2244  siebmtshmw  Connection disconnected, host [R12KSA01].
10/26/02 15:31:21.879  2244  siebmtshmw  [DRIVER] Unable to connect to CiscoDataStore.
10/26/02 15:31:21.879  2244  siebmtshmw  CTestSession::OnConnectionFailure,
(MessageID:eOnConnectionFailure
    EventTime:27076878 ReasonCode:3 FailedServer:R12KSA01)
```

  This indicates that the connection is maintained automatically whenever Cisco Data Store starts.

  Recommended Action Verify Cisco Data Store status by doing the following:

- Check if Cisco Data Store was installed on R12KSA01 and the default port was chosen as 42029 by checking the Registry on the R21KSA01 machine:

  HKLM\Software\Cisco Systems\ObjectStoreserverSocket\Conections
  "ListenPort"=42029 (in decimal)

- Start Cisco Service Control (via Program Files) on R12KSA01. An entry for a newly installed Cisco Data Store displays; select it and click the **Start** button. The cmd window should display and indicate that Cisco Data Store is running and listening on port 42029.
- Do the following:
  - If the connection to Cisco Data Store is unavailable from a client machine, try pinging the Cisco Data Store machine.
  - Verify if the machine where the driver is installed (Siebel Comm Server).
  - If you do not have the Cisco Data Store installed, see Chapter 2, "Installation," for installation instructions.
  - After performing these items, do the transfer again and check if the connection to Cisco Data Store is re-established and bookmark it.

APPENDIX A

Business Service Script

This appendix displays a complete business services script that contains available functions. It also provides a sample DEF file that displays the broadcast statistics configuration.

Sample Script

The following script contains the functions implemented on this business service that is not exposed to a client, but can be used internally.

Example A-1 Business Service Script with Available Functions

```
function AddUserToBroadcast(strUserName, strMsgAbstract)
{
    // AddUserToBroadcast:
    //
    // Associates the Employee having the given Siebel login name
    // with the Broadcast Message used to send statistics for
    // the given skill group or agent group.  The Abstract field
    // of the Broadcast Message will determine whether it is
    // used to show agent statistics or skill group statistics.

    var strPositionId  = FindPositionId(strUserName);

    if (strPositionId == "")
    {
        ShowStatusText("Can't find or create Position for login name " + strUserName, true);
        return;
    }

    var boBroadcastMessage = null;
    var bcBroadcastMessage = null;

    GetBroadcastMsg(strMsgAbstract, boBroadcastMessage, bcBroadcastMessage);

    if (bcBroadcastMessage == null)
    {
        ShowStatusText("Unable to find or create Broadcast Message " + strMsgAbstract, true);
        return;
    }

    if (AssociateChildRecord(bcBroadcastMessage, "Position Id", "Id", strPositionId) == "")
    {
        ShowStatusText("Unable to associate Position with Broadcast Message", true);
    }
    bcBroadcastMessage = null;
    boBroadcastMessage = null;
}

function AssociateChildRecord(bcParent, strMVFieldName, strDestFieldName, strDestFieldValue)
{
    // AssociateChildRecord:
    //
    // Takes a parent business component, the field
    // name of a Multi-Value Field in that business component
    // and the Row Id of a record in the destination business
    // component of that Multi-Value Field.
    //
    // If an association already exists between the current
    // parent BC record and the record with the given
    // Row Id, then no action is taken.  Otherwise, these two
    // records are associated.
    //
    // If an error occurred while performing these tasks,
    // then ShowStatusText will be used to display an error
    // message, or a Siebel exception will be thrown.
    //
    // If an exception is not thrown, the return value will be
    // either a string containing the Row Id of the associated
    // record, or the empty string if an error occurred.

    if (bcParent == null)
    {
        ShowStatusText("Invalid Broadcast Message BC argument", true);
        return "";
    }

    var bcMVG = bcParent.GetMVGBusComp(strMVFieldName);

    if (bcMVG == null)
    {
        ShowStatusText("Unable to create MVG BusComp for " + strMVFieldName, true);
        return "";
    }

    with (bcMVG)
    {
        ActivateField(strDestFieldName);
        SetSearchSpec(strDestFieldName, strDestFieldValue);
        ExecuteQuery( ForwardOnly );

        if (FirstRecord())
        {
            // This association already exists; do nothing
            // and return its Row Id
                                                                bcMVG = null;
            return GetFieldValue("Id");
        }
    }

    // There is no existing association; create a new one

    var bcAssoc = bcMVG.GetAssocBusComp();

    if (bcAssoc == null)
    {
        bcMVG = null;
        ShowStatusText("Unable to create Association BusComp for " + strMVFieldName, true);
        return "";
    }

    var retVal = "";

    with ( bcAssoc )
    {
        ClearToQuery();
        SetViewMode( AllView );
        ActivateField(strDestFieldName);
        SetSearchSpec(strDestFieldName, strDestFieldValue);
        ExecuteQuery( ForwardOnly );

        if( FirstRecord( ) )
        {
        Associate( 1 );// NewAfter = 1
        retVal = GetFieldValue("Id"); // return Row Id of associated record
        }
        else
        {
            ShowStatusText("Unable to find " + strMVFieldName + " " + strDestFieldValue, true);
            retVal = "";
        }
   }

   bcAssoc = null;
   bcMVG = null;
   return retVal;
}

function ClearAgentBroadcasts(strUserName, strPrefix)
{
    // ClearAgentBroadcasts:
    //
    // Removes any associations between the agent having the given
    // Siebel login name and all Broadcast Messages whose Abstracts
    // begin with strPrefix.  (Use CTISkillStat_ as the prefix for
    // skill group messages, CTIAgentStat_ as the prefix for agent
    // messages, and CTI*Stat_ as the prefix to match both.
    //
    // Should be called when the agent
    // first logs into Siebel so that the agent will initially
    // not receive any skill group statistics.

    // Find the Position used to send statistics to this agent.

    var strPositionId = FindPositionId(strUserName);

    if (strPositionId == "")
    {
        ShowStatusText("Can't find or create Position for login name " + strUserName, true);
        return;
    }

    // Iterate through all Broadcast Messages starting with
    // the prefix strPrefix and remove any association between this agent's
    // Position and those messages.

    var boBroadcastMessage = TheApplication().GetBusObject("Broadcast Message");

    if (boBroadcastMessage == null)
    {
        ShowStatusText("Unable to create Broadcast Message business object", true);
        return;
    }

    var bcBroadcastMessage=boBroadcastMessage.GetBusComp("Broadcast Message");

    if (bcBroadcastMessage == null)
    {
        boBroadcastMessage = null;
        ShowStatusText("Unable to create Broadcast Message business component", true);
        return;
    }

    with (bcBroadcastMessage)
    {
        ClearToQuery();
        SetViewMode( AllView );
        ActivateField("Abstract");
        ActivateField("Position Id");

        SetSearchSpec("Abstract", "LIKE '" + strPrefix + "*'");
        ExecuteQuery( ForwardOnly );
     }


    var hasRecord = bcBroadcastMessage.FirstRecord();
    while (hasRecord)
    {
        var bcMVG = bcBroadcastMessage.GetMVGBusComp("Position Id");

        if (bcMVG == null)
        {
            ShowStatusText("Unable to create MVG BusComp for Position Id", true);
            bcBroadcastMessage = null;
            boBroadcastMessage = null;
            return;
        }

        // Search through the existing Position associations with
        // this Broadcast Message for one whose Row Id
        // matches the agent's statistics Position Id.  If one
        // is found, remove it.

        bcMVG.SetSearchSpec("Id", strPositionId);
        bcMVG.ExecuteQuery( ForwardOnly );

        while (bcMVG.FirstRecord())
        {
            bcMVG.DeleteRecord();
        }

        bcMVG = null;

        hasRecord = bcBroadcastMessage.NextRecord();
    }

    bcBroadcastMessage = null;
    boBroadcastMessage = null;
}

function FindPositionId(strUserName)
{
    // FindPositionId:
    //
    // Takes a Siebel login name (not login ID)
    // and finds or creates a Position specifically
    // used for that agent to receive statistics messages.
    // The Row Id of the Position is returned.
    //
    // The name of the Position will be the Siebel login name
    // with a prefix of "CTIAgentStat_"
    //
    // If an error occurred,the function
    // will call ShowStatusText to display the error
    // and will return the empty string, or in some cases
    // will throw a Siebel exception.
    //
    // Any script that calls this function
    // should check for an empty string as the
    // return value and react accordingly.



    var strPositionName = "CTIAgentStat_" + strUserName;
    var strPositionId = "";

    var boPosition=TheApplication().GetBusObject("Position");

    if (boPosition == null)
    {
        ShowStatusText("Unable to create Position business object", true);
        return "";
    }

    var bcPosition=boPosition.GetBusComp("Position");

    if (bcPosition == null)
    {
        boPosition = null;
        ShowStatusText("Unable to create Position business component", true);
        return "";
    }

    with (bcPosition)
    {
        ClearToQuery();
        SetViewMode( AllView );
        ActivateField("Name");
        ActivateField("Login Name");
        ActivateField("Division");
        SetSearchSpec("Name", strPositionName);
        ExecuteQuery( ForwardOnly );

        if( FirstRecord() == 0 )
        {
            // No Position record exists for this Name;
            // create a new record and set its Name field
            NewRecord( 1 );// NewAfter = 1
            SetFieldValue("Name", strPositionName);
            SetFieldValue("Division", "Default Organization");
            WriteRecord();

            // Associate the newly created Position
            // with the Employee identified by the
            // Siebel login name
            var strEmployeeId = AssociateChildRecord(bcPosition, "Login Name", "Login Name", strUserName);

            if (strEmployeeId == "")
            {
                ShowStatusText("Unable to associate new Position with Employee");
                DeleteRecord( );
                bcPosition = null;
                boPosition = null;
                return "";
            }

            SetFieldValue("Primary Employee Id", strEmployeeId);
            WriteRecord();
        }

        strPositionId = GetFieldValue("Id");
    }

    bcPosition = null;
    boPosition = null;
    return strPositionId;
}

function GetBroadcastMsg(strAbstract, &boBroadcastMessage, &bcBroadcastMessage)
{
    // GetBroadcastMsg:
    //
    // Given a value for the "Abstract" field, looks for a
    // Broadcast Message record whose Abstract field contains
    // that value.  If one is found, it is returned without
    // modification; otherwise a new Broadcast Message record
    // is created and a business component pointing
    // to that record is placed in bcBroadcastMessage.
    //
    // If an error occurs, ShowStatusText will be used to
    // display a descriptive error message, and null will
    // be returned.  The script that calls this function
    // should check for a null bcBroadcastMessage and react
    // accordingly.  It should also take care to set
    // boBroadcastMessage and bcBroadcastMessage to null
    // when they are no longer needed.

    boBroadcastMessage = TheApplication().GetBusObject("Broadcast Message");

    if (boBroadcastMessage == null)
    {
        ShowStatusText("Unable to create Broadcast Message business object", true);
        return;
    }

    bcBroadcastMessage=boBroadcastMessage.GetBusComp("Broadcast Message");

    if (bcBroadcastMessage == null)
    {
        boBroadcastMessage = null;
        ShowStatusText("Unable to create Broadcast Message business component", true);
        return;
    }

    with (bcBroadcastMessage)
    {
        ClearToQuery();
        SetViewMode( AllView );
        ActivateField("Abstract");
        ActivateField("All");
        ActivateField("Expiration Date/Time");
        ActivateField("Body");
        ActivateField("Position Id");
        SetSearchSpec("Abstract", strAbstract);
        ExecuteQuery( ForwardOnly );

        if( FirstRecord() == 0)
        {
            // No Broadcast Message record exists for this Abstract;
            // create a new record and set its Abstract field
            NewRecord( 1 );// NewAfter = 1
            SetFieldValue("Abstract", strAbstract);
            SetFieldValue("Body", "---");// A non-empty value for this field is required
            SetFieldValue("All", "N");
            SetFieldValue("Expiration Date/Time", "01/01/2000");// Already expired
            WriteRecord();
        }
    }
}

function RemoveUserFromBroadcast(strUserName, strMsgAbstract)
{
    // RemoveUserFromBroadcast:
    //
    // Removes any association between the Employee having the given
    // Siebel login name and the Broadcast Message used to send
    // agent or skill group statistics.

    // Find the Position used to send statistics to this user.

    var strPositionId = FindPositionId(strUserName);

    if (strPositionId == "")
    {
        ShowStatusText("Can't find or create Position for login name " + strUserName, true);
        return;
    }

    // Find the Broadcast Message

    var boBroadcastMessage = null;
    var bcBroadcastMessage = null;

    GetBroadcastMsg(strMsgAbstract, boBroadcastMessage, bcBroadcastMessage);

    if (bcBroadcastMessage == null) {
        ShowStatusText("Unable to find or create Broadcast Message " + strMsgAbstract, true);
        return;
    }

    var bcMVG = bcBroadcastMessage.GetMVGBusComp("Position Id");

    if (bcMVG == null)
    {
        ShowStatusText("Unable to create MVG BusComp for Position Id", true);
        bcBroadcastMessage = null;
        boBroadcastMessage = null;
        return;
    }

    // Search through the existing Position associations with
    // this Broadcast Message for one whose Row Id
    // matches the user's statistics Position Id.  If one
    // is found, remove it.

    bcMVG.SetSearchSpec("Id", strPositionId);
    bcMVG.ExecuteQuery( ForwardOnly );

    while (bcMVG.FirstRecord())
    {
        bcMVG.DeleteRecord();
    }

    bcMVG = null;
    bcBroadcastMessage = null;
    boBroadcastMessage = null;
}

function ShowStatusText(strText, bRaiseError)
{
    // ShowStatusText:
    //
    // Shows a text message on the Siebel status bar.
    //
    // This depends on the Communications Client business service
    // being available, and on the Comm toolbar being displayed
    // on the agent's screen.  However, since the functions
    // declared in this business service are only used by
    // a Communications EventResponse, it is a safe assumption
    // that the Comm toolbar will always be available.
    //
    // Preconditions: strText is a valid text string.
    //    bRaiseError is a boolean value (either true or false.)
    //
    // Postconditions: If the Communications Client business
    //    service is available, then this function has called
    //    its method ShowStatusText, passing it the string
    //    argument as the text to be displayed.  The function
    //    will return a boolean "true" value in this case.
    //
    //    If the business service is not available, or if
    //    any other error occurs, the function will either
    //    call RaiseErrorText with strText as its argument
    //    (if bRaiseError is true) or will do nothing and
    //    return a boolean "false" value (if bRaiseError
    //    is false.)

    var retVal = true;

    // try {
        var bsCommClient = TheApplication( ).GetService("Communications Client");

        if (bsCommClient == null)
        {
            retVal = false;
        }
        else
        {
            var psInputs  = TheApplication( ).NewPropertySet( );
            var psOutputs = TheApplication( ).NewPropertySet( );

            psInputs.SetProperty("Text", strText);
            bsCommClient.InvokeMethod("ShowStatusText", psInputs, psOutputs);

            psOutputs = null;
            psInputs = null;
            bsCommClient = null;
            retVal = true;
        }
    /*
    }
    catch(e)
    {
        retVal = false;
    }
    finally
    {
    */
        if (retVal != true) {
            // An error occurred
            if (bRaiseError == true)
            {
                TheApplication( ).RaiseErrorText(strText);
            }
            else
            {
                // Do nothing
                return false;
            }
        }
    // }
}
```

Configuring Broadcast Statistics Within the DEF File

The sample DEF file with complete broadcast statistics configuration resides on the product CD in the Documentation directory under the Siebel directory. The sample DEF file name is cisco_agentANDskillgroupSTATISTICS_sample_ENU.def.

Driver Parameters

The Driver:BroadcastStatistics="TRUE" parameter enables both agent and skill group broadcast statistics unless it is not overridden later in the Login command. For example, to enable/disable broadcast statistics for all agents under a particular configuration, set Driver:BroadcastStatistics to "TRUE"/"FALSE".

This setting can be overridden specifically either for AgentStatisticsBroadcast or SkillGroupStatisticsBroadcast by using the following parameters in the Login command:

- Param.SkillGroupBroadcastStat
- Param.AgentBroadcastStat

Example A-2 Sample Siebel DEF File for Broadcast Statistics Configuration

```
[Command:LoginToPBXDevice]
   Profile       = ""
   DeviceCommand = "LogIn"
   Comments      = ""
   CmdData       = "LoginToPBXDevice"
   Hidden        = "TRUE"

   [CmdData:LoginToPBXDevice]
      Param.AgentWorkMode           = "0"
      Param.UserId                  = "{@UserName}"
      Comments                      = ""
      Param.SkillGroupBroadcastStat = "TRUE"
      Param.Password                = "{@AgentPin}"
      Param.ACDQueue                = "{@QueueList}"
      Param.AgentBroadcastStat      = "FALSE"
      Param.AgentId                 = "{@AgentId}"
```

If, on a driver level, Driver:BroadcastStatistics="FALSE," the skill group statistics broadcast is re-enabled by specifying Param.SkillGroupBroadcastStat="TRUE," etc.

Note These parameters are optional (can use both, only one, or none).

If there are no parameters on Login, then the driver level parameter will be used.

APPENDIX B

Configuring Agent State Toggling

This appendix provides a sample DEF file that displays the ChangeBusyState command configuration for toggling the NotReady button between the Ready/NotReady agent states.

Configuring Agent State Toggling Within the DEF File

The sample DEF file with complete ChangeBusyState command toggling configuration resides on a CD in the Documentation directory under the Siebel directory. The sample DEF file name is cisco_default_enu.def.

Example B-1 Sample Siebel DEF File for ChangeBusyState Command Toggling Configuration

```
[Siebel]
   CommServerVersion = "7.0"

[Configuration Parameters]
   AutoLogin                  = "FALSE"
   AutoLoginCmd               = "SignOnGroup"
   DialingFilter.Rule1        = "6504771->"
   DialingFilter.Rule2        = "650->9"
   DialingFilter.Rule3        = "6175551->0"
   DialingFilter.Rule4        = "->91"
   MaxCommToolbars            = "1"
   MultiTenancy               = "FALSE"
   PreferenceLoginCmd         = "PreferenceLoginCmd"
   PreferenceLogoutCmd        = "PreferenceLogoutCmd"
   RestoreScreenOnWorkResumed = "FALSE"
   UpdateChannelStatusTable = "TRUE"

[Profile:CiscoProfile_forIPCC]
   Driver:SideAPort         = "42028"
   Driver:PeripheralID      = "5000"
   Driver:SideBHost         = "CTIOSsideB"
   Driver:SideAHost         = "CTIOSsideA"
   Driver:SideBPort         = "42028"
   Driver                   = "Cisco Driver"

   [Driver:Cisco Driver]
      Library Name   = "Sieb7CiscoCTI.dll"
      Outbound Flag  = "Y"
      Channel String = "CISCO Phone"
      Channel Type   = "Voice"
      Icon File      = "voice.gif"
      Interactive    = "Y"
      Inbound Flag   = "Y"

      [Driver Parameters:Cisco Driver]
         Required:Driver:SideAPort        = "42028"
         Driver:IsEasySim                 = "FALSE"
         Service:SelectDN                 = "{@SelectedDN}"
         Service:ACDDNList                = "{@ACDDNList}"
         Required:Driver:PeripheralID     = "5009"
         Driver:SideBHost                 = "CTIOSsideA"
         Driver:DataServerName            = "CDSHostName"
         Driver:DataServerPort            = "42029"
         Driver:CiscoLogFileName          = "C:\IPCC"
         Service:HasForward               = "FALSE"
         Required:Service:DNList          = "{@DNList}"
         Required:Service:IsQueueRequired = "TRUE"
         Required:Driver:SideAHost        = "CTIOSsideA"
         Required:Driver:LibraryName      = "Sieb7CiscoCTI.dll"
         Service:ServiceLogFile           = "ctc_{@Username}.log"
         Driver:BroadcastStatistics       = "FALSE"
         Driver:SideBPort                 = "42028"
         Driver:LogDebug                  = "TRUE"
         Driver:ViewBmkCookie             = "CallVariable10"
         Service:AutoLogout               = "FALSE"
         Service:AutoLogoutReasonCode     = "0"

[Command:AgentAfterCallWork]
   Profile       = ""
   DeviceCommand = "ChangeWorkReadyState"
   MenuPosition  = "50.3"
   Comments      = ""
   Title         = "Work Mode : After Call Work"
   Description   = "Set AgentWorkMode to After-Call-Work"
   HotKey        = "Ctrl+F4"
   CmdData       = ""

[Command:AssociateContact]
   Profile       = ""
   DeviceCommand = "@Associate"
   Comments      = ""
   CmdData       = "AssociateContact"
   Hidden        = "TRUE"

   [CmdData:AssociateContact]
      Param."Contact Id" = "{Id}"
      Comments           = ""
      BusComp            = "Contact"

[Command:HoldCall]
   Profile           = ""
   DeviceCommand     = "HoldCall"
   Comments          = ""
   CmdData           = "HoldCall"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:HoldCall]
      Param.TrackingID = "{@SelectedWorkItem:DriverWorkTrackID}"
      Comments         = ""

[Command:MakeCallToCampaignContact]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to campaign contact"
   CmdData       = "MakeCallToCampaignContact"
   Hidden        = "TRUE"

   [CmdData:MakeCallToCampaignContact]
      RequiredField.'Work Phone #' = "?*"
      Comments                     = ""
      Param.ContactId              = "{Contact Id}"
      Param.CampaignId             = "{Campaign Id}"
      Param.CampaignPhone          = "{Work Phone #}"
      BusComp                      = "Campaign List Contact"
      Param.PhoneNumber            = "{Work Phone #:Lookup}"

[Command:MakeCallToSelectedPhone]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to "{@SelectedText}""
   CmdData       = "MakeCallToSelectedPhone"
   Hidden        = "TRUE"

   [CmdData:MakeCallToSelectedPhone]
      AttachContext               = "TRUE"
      Comments                    = ""
      RequiredField.@SelectedText = "?*"
      Param.CallNotifyText        = "Call from {@UserName}..."
      Param.PhoneNumber           = "{@SelectedText:Lookup}"

[Command:PreferenceLogoutCmd]
   Profile       = ""
   DeviceCommand = "LogOut"
   Comments      = ""
   Description   = "Logout from selected ACD queues"
   CmdData       = "PreferenceLogoutCmd"
   Hidden        = "TRUE"

   [CmdData:PreferenceLogoutCmd]
      Comments       = ""
      Param.ACDQueue = "{@SelectedQueue}"

[Command:ReconnectCallPBX]
   Profile           = ""
   DeviceCommand     = "ReconnectCall"
   MenuPosition      = "70.2"
   Comments          = ""
   Title             = "Reconnect (Phone)"
   CmdData           = "ReconnectCallPBX"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ReconnectCallPBX]
      Param.TrackingID = "{@SelectedWorkItem:DriverWorkTrackID}"
      Comments         = ""

[Command:ReleaseWorkGroup]
   SubCommand_1 = "ReleasePhoneCall"
   Profile      = ""
   Comments     = ""
   Description  = "Release work item"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:WorkItemList]
   SubCommand_1 = "SuspendDeselectedCall"
   SubCommand_2 = "ResumeSelectedCall"
   SubCommand_3 = "RefreshDashboardAtCallChange"
   Profile      = ""
   Comments     = ""
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:AssociateConsumer]
   Profile       = ""
   DeviceCommand = "@Associate"
   Comments      = ""
   CmdData       = "AssociateConsumer"
   Hidden        = "TRUE"

   [CmdData:AssociateConsumer]
      Param."Contact Id" = "{Id}"
      Comments           = ""
      BusComp            = "Consumer"

[Command:ConferenceTransferToCurrentPhone]
   Profile           = ""
   DeviceCommand     = "ConferenceInit"
   Comments          = ""
   Description       = "Conference transfer to "{@Phone}""
   CmdData           = "ConferenceTransferToCurrentPhone"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConferenceTransferToCurrentPhone]
      AttachContext        = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Conference transfer from {@UserName} ..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:MakeCallGroup]
   SubCommand_1  = "MakeCallToPhone"
   SubCommand_2  = "MakeCallToSelectedPhone"
   SubCommand_3  = "MakeCallToSRContact"
   SubCommand_4  = "MakeCallToSROwner"
   SubCommand_5  = "MakeCallToEmployeeHome"
   SubCommand_6  = "MakeCallToEmployee"
   SubCommand_7  = "MakeCallToContactHome"
   SubCommand_8  = "MakeCallToContact"
   SubCommand_9  = "MakeCallToAccount"
   SubCommand_10 = "MakeCallToCampaignContactHome"
   SubCommand_11 = "MakeCallToCampaignContact"
   SubCommand_12 = "MakeCallToExtension"
   SubCommand_13 = "MakeCallToActivityContact"
   SubCommand_14 = "MakeCallToCurrentPhone"
   SubCommand_15 = "MakeCallToPopupEmployee"
   Profile       = ""
   Comments      = ""
   CmdData       = ""
   Hidden        = "TRUE"

[Command:MakeCallToEmployee]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to employee"
   CmdData       = "MakeCallToEmployee"
   Hidden        = "TRUE"
   [CmdData:MakeCallToEmployee]
      Comments                 = ""
      Param.CallNotifyText     = "Call from {@UserName}..."
      RequiredField.'Phone #' = "?*"
      BusComp                  = "Employee"
      Param.PhoneNumber        = "{Phone #:Lookup}"

[Command:MakeCallToExtension]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to extension"
   CmdData       = "MakeCallToExtension"
   Hidden        = "TRUE"

   [CmdData:MakeCallToExtension]
      Comments             = ""
      Param.CallNotifyText = "Call from {@UserName}..."
      BusComp              = "Telephone Status"
      Param.PhoneNumber    = "{Extension}"

[Command:MakeCallToSRContact]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to service request contact"
   CmdData       = "MakeCallToSRContact"
   Hidden        = "TRUE"

   [CmdData:MakeCallToSRContact]
      OnField                                = "Contact Business Phone"
      Comments                               = ""
      RequiredField.'Contact Business Phone' = "?*"
      BusComp                                = "Service Request"
      Param.PhoneNumber                      = "{Contact Business Phone:Lookup}"

[Command:SimCallNotFound]
   Profile       = ""
   DeviceCommand = "SimulateCall"
   MenuPosition  = "0.1"
   Comments      = ""
   HotKey        = "Ctrl+F11"
   CmdData       = "SimCallNotFound"

   [CmdData:SimCallNotFound]
      Comments            = ""
      Param.CallVariable1 = "4153218811"

[Command:UpdateDashboardFromActivity]
   Profile       = ""
   Comments      = ""
   CmdData       = "UpdateDashboardFromActivity"
   ServiceMethod = "Persistent Customer Dashboard.Update Dashboard from CTI"
   Hidden        = "TRUE"

   [CmdData:UpdateDashboardFromActivity]
      ServiceParam.Value         = "{Contact Id}"
      RequiredField.'Contact Id' = "?*"
      WorkTrackingObj.ContactId  = "{Contact Id}"
      Comments                   = ""
      ServiceParam.Field         = "Id"
      BusComp                    = "Action"

[Command:InitiateWorkGroup]
```

```
   SubCommand_1  = "MakeCallToPhone"
   SubCommand_2  = "MakeCallToSelectedPhone"
   SubCommand_3  = "MakeCallToSRContact"
   SubCommand_4  = "MakeCallToSROwner"
   SubCommand_5  = "MakeCallToEmployeeHome"
   SubCommand_6  = "MakeCallToEmployee"
   SubCommand_7  = "MakeCallToContactHome"
   SubCommand_8  = "MakeCallToContact"
   SubCommand_9  = "MakeCallToAccount"
   SubCommand_10 = "MakeCallToCampaignContactHome"
   SubCommand_11 = "MakeCallToCampaignContact"
   SubCommand_12 = "MakeCallToExtension"
   SubCommand_13 = "MakeCallToActivityContact"
   SubCommand_14 = "MakeCallDummy"
   Profile       = ""
   Comments      = ""
   Description   = "Initiate work item"
   CmdData       = ""
   Hidden        = "TRUE"

[Command:MakeCallToAccount]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to account"
   CmdData       = "MakeCallToAccount"
   Hidden        = "TRUE"

   [CmdData:MakeCallToAccount]
      Comments                          = ""
      RequiredField.'Main Phone Number' = "?*"
      BusComp                           = "Account"
      Param.PhoneNumber                 = "{Main Phone Number:Lookup}"

[Command:NotReadyForPhoneSetToReadyNoPopup]
   Profile       = ""
   FilterSpec    = "[$GetCommandStatus(ChangeReadyState)] = 'Enabled' AND 
[$GetCommandStatus(ChangeNotReadyState)]= 'Enabled'"
   DeviceCommand = "ChangeBusyState"
   Comments      = ""
   Description   = "Phone: set to not ready"
   CmdData       = "NotReadyNoPopup"
   Hidden        = "TRUE"

   [CmdData:NotReadyNoPopup]
      Param.ReasonCode = "[Value]"
      Comments         = ""

[Command:ConsultativeTransferToSROwner]
   Profile           = ""
   DeviceCommand     = "TransferInit"
   Comments          = ""
   Description       = "Consultative transfer to service request owner"
   CmdData           = "ConsultativeTransferToSROwner"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConsultativeTransferToSROwner]
      RequiredField.'Owner Phone' = "?*"
      AttachContext               = "TRUE"
      Comments                    = ""
      Param.CallNotifyText        = "Consultative transfer from {@UserName} about SR 
{Id}..."
      BusComp                     = "Service Request"
      Param.PhoneNumber           = "{Owner Phone:Lookup}"

[Command:NotReadyForPhoneSetToNotReady]
   Profile       = ""
   FilterSpec    = "[$GetCommandStatus(ChangeReadyState)] = 'Disabled' AND
[$GetCommandStatus(ChangeNotReadyState)]= 'Enabled'"
   DeviceCommand = "ChangeBusyState"
   Comments      = ""
   Description   = "Phone: set to not ready"
   CmdData       = "NotReadyWithPopup"
   Hidden        = "TRUE"

   [CmdData:NotReadyWithPopup]
      Param.ReasonCode = "[Value]"
      SelectBusComp    = "List Of Values"
      SelectParam      = "TRUE"
      Comments         = ""
      SelectBusObj     = "List Of Values"
      SelectApplet     = "Transfer Multiple LOV Popup Applet"
      SelectTitle      = "Please select the reason for changing status to Not-Ready"
      SelectQuerySpec  = "[Type] = 'REASON_CODE' AND [Active] = 'Y'"

[Command:NotReadyForPhoneSetToReady]
   CmdData       = ""
   Description   = "Phone: set to ready"
   DeviceCommand = "ChangeBusyState"
   FilterSpec    = "[$GetCommandStatus(ChangeBusyState)] = 'Checked'"
   Hidden        = "TRUE"
   Profile       = ""
   Comments      = ""
   CmdData       = ""

[Command:SuspendWorkGroup]
   SubCommand_1 = "HoldCall"
   Profile      = ""
   Comments     = ""
   Description  = "Pause work item"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:AltReconnectInMenu]
   SubCommand_1 = "RetrieveCallPBX"
   Profile      = ""
   MenuPosition = "70"
   Comments     = ""
   Title        = "AlternateCall"
   Description  = "AlternateCall"
   CmdData      = ""

[Command:ConferenceTransferToPhone]
   Profile            = ""
   DeviceCommand      = "ConferenceInit"
   Comments           = ""
   Description        = "Conference transfer to "{@Phone}""
   CmdData            = "ConferenceTransferToPhone"
   OnEditControl      = "TRUE"
   Hidden             = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConferenceTransferToPhone]
      AttachContext        = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Conference transfer from {@UserName} ..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:ConsultativeTransferGroup]
   SubCommand_1 = "CompleteConsultativeTransferToPhone"
   SubCommand_2 = "ConsultativeTransferToPhone"
   SubCommand_3 = "ConsultativeTransferToSROwner"
   SubCommand_4 = "ConsultativeTransferToEmployee"
   SubCommand_5 = "ConsultativeTransferToExtension"
   SubCommand_6 = "ConsultativeTransferToCurrentPhone"
   SubCommand_7 = "ConsultativeTransferToPopupEmployee"
   Profile      = ""
   Comments     = ""
   Description  = "Consultative transfer work item"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:PreferenceLoginCmd]
   Profile       = ""
   DeviceCommand = "LogIn"
   Comments      = ""
   Description   = "Login to CTI"
   CmdData       = "PreferenceLoginCmd"
   Hidden        = "TRUE"

   [CmdData:PreferenceLoginCmd]
      Comments       = ""
      Param.Password = "{@AgentPin}"
      Param.ACDQueue = "{@SelectedQueue}"
      Param.AgentId  = "{@AgentId}"

[Command:ConferenceTransferGroupInLocalMenu]
   SubCommand_1 = "ConferenceTransferToSROwner"
   SubCommand_2 = "ConferenceTransferToEmployee"
   SubCommand_3 = "ConferenceTransferToExtension"
   SubCommand_4 = "ConferenceTransferToCurrentPhone"
   Profile      = ""
   MenuPosition = "3"
   Comments     = ""
   Title        = "Conference Transfer"
   LocalMenu    = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:LoginToPBX]
   Profile       = ""
   DeviceCommand = "LogIn"
   MenuPosition  = "20.1"
   Comments      = ""
   Title         = "Log In (Phone)"
   CmdData       = "LoginToPBX"

   [CmdData:LoginToPBX]
      Param.AgentWorkMode = "0"
      Param.UserId        = "{@UserName}"
      Comments            = ""
      Param.Password      = "{@AgentPin}"
      Param.ACDQueue      = "{@QueueList}"
      Param.AgentId       = "{@AgentId}"

[Command:NotReadyGroup]
   SubCommand_1 = "NotReadyGroupSetToNotReady"
   SubCommand_2 = "NotReadyGroupSetToReady"
   Profile      = ""
   Comments     = ""
   Description  = "Change ready state"
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:SimCallFound]
   Profile       = ""
   DeviceCommand = "SimulateCall"
   MenuPosition  = "0.2"
   Comments      = ""
   HotKey        = "Shift+F11"
   CmdData       = "SimCallFound"

   [CmdData:SimCallFound]
      Comments            = ""
      Param.CallVariable1 = "111222333"

[Command:ToolbarInMenu]
   SubCommand_1 = "RefreshToolbarInMenu"
   SubCommand_2 = "GetPreviousMsgInMenu"
   SubCommand_3 = "GetNextMsgInMenu"
   Profile      = ""
   MenuPosition = "60"
   Comments     = ""
   Title        = "Toolbar"
   CmdData      = ""

[Command:AssociateGroupInLocalMenu]
   SubCommand_1 = "AssociateAccount"
   SubCommand_2 = "AssociateContact"
   SubCommand_3 = "AssociateSR"
   SubCommand_4 = "AssociateDefect"
   SubCommand_5 = "AssociateCampaign"
   SubCommand_6 = "AssociateOppty"
   SubCommand_7 = "AssociateConsumer"
   Profile      = ""
   MenuPosition = "1"
   Comments     = ""
   Title        = "Associate"
   LocalMenu    = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:CompleteConsultativeTransferToPhone]
   Profile           = ""
   DeviceCommand     = "TransferComplete"
   Comments          = ""
   CmdData           = ""
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

[Command:ConferenceTransferToEmployee]
   Profile           = ""
   DeviceCommand     = "ConferenceInit"
   Comments          = ""
   Description       = "Conference transfer to employee"
   CmdData           = "ConferenceTransferToEmployee"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConferenceTransferToEmployee]
      Comments                = ""
      Param.CallNotifyText    = "Conference transfer from {@UserName}..."
      RequiredField.'Phone #' = "?*"
      BusComp                  = "Employee"
      Param.PhoneNumber        = "{Phone #:Lookup}"

[Command:RefreshToolbarInMenu]
   Profile       = ""
   MenuPosition  = "60.1"
   Comments      = ""
   Title         = "Refresh"
   Description   = "Refresh communications toolbar"
   HotKey        = "Alt+R"
   CmdData       = "RefreshToolbarInMenu"
   ServiceMethod = "Communications Client.ShellUIUpdate"

   [CmdData:RefreshToolbarInMenu]
      Comments      = ""
      Param.InitAll = "TRUE"

[Command:ReleasePhoneCallInMenu]
   SubCommand_1 = "ReleasePhoneCall"
   Profile      = ""
   MenuPosition = "7"
   Comments     = ""
   Title        = "Release Call"
   Description  = "Release call"
   HotKey       = "F12"
   CmdData      = ""

[Command:SignOnGroupInMenu]
   SubCommand_1 = "LoginToPBX"
   Profile      = ""
   MenuPosition = "20"
   Comments     = ""
   Title        = "Log In"
   Description  = "Log in"
   CmdData      = ""

[Command:ConsultativeTransferGroupInMenu]
   SubCommand_1 = "CompleteConsultativeTransferToPhone"
   SubCommand_2 = "ConsultativeTransferToPhone"
   SubCommand_3 = "ConsultativeTransferToSROwner"
   SubCommand_4 = "ConsultativeTransferToEmployee"
   SubCommand_5 = "ConsultativeTransferToExtension"
   SubCommand_6 = "ConsultativeTransferToCurrentPhone"
   SubCommand_7 = "ConsultativeTransferToPopupEmployee"
   Profile      = ""
   MenuPosition = "5"
   Comments     = ""
   Title        = "Consultative Transfer"
   HotKey       = "Shift+F7"
   CmdData      = ""

[Command:NotReadyForPhoneGroup]
   SubCommand_1 = "NotReadyForPhoneSetToReady"
   SubCommand_2 = "NotReadyForPhoneSetToNotReady"
   SubCommand_3 = "NotReadyForPhoneSetToReadyNoPopup"
   Profile      = ""
   Comments     = ""
   Description  = "Change ready state for phone"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:SignOffGroup]
   SubCommand_1 = "LogoutFromPBX"
   Profile      = ""
   Comments     = ""
   Description  = "Log out"
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:SimCampaignCall]
   Profile       = ""
   DeviceCommand = "SimulateCall"
   MenuPosition  = "0.3"
   Comments      = ""
   HotKey        = "Alt+F11"
   CmdData       = "SimCampaignCall"

   [CmdData:SimCampaignCall]
      Param.CampContactID = "FAKE_ID"
      Comments            = ""
      Param.CampID        = "1-CQZ"

[Command:AgentNotReady]
   Profile       = ""
   DeviceCommand = "ChangeNotReadyState"
   MenuPosition  = "50.1"
   Comments      = ""
   Title         = "Work Mode : Not Ready"
   Description   = "Set AgentWorkMode to Not-Ready"
   HotKey        = "Ctrl+F2"
   CmdData       = ""

[Command:AssociateDefect]
   Profile       = ""
   DeviceCommand = "@Associate"
   Comments      = ""
   CmdData       = "AssociateDefect"
   Hidden        = "TRUE"

   [CmdData:AssociateDefect]
      Comments                   = ""
      Param."Activity Defect Id" = "{Id}"
      BusComp                    = "Product Defect"

[Command:ConferenceTransferToExtension]
   Profile           = ""
   DeviceCommand     = "ConferenceInit"
   Comments          = ""
   Description       = "Conference transfer to extension"
   CmdData           = "ConferenceTransferToExtension"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConferenceTransferToExtension]
      Comments             = ""
      Param.CallNotifyText = "Conference transfer from {@UserName}..."
      BusComp              = "Telephone Status"
      Param.PhoneNumber    = "{Extension}"

[Command:ConsultativeTransferToEmployee]
   Profile           = ""
   DeviceCommand     = "TransferInit"
   Comments          = ""
   Description       = "Consultative transfer to employee"
   CmdData           = "ConsultativeTransferToEmployee"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"
```

```
      [CmdData:ConsultativeTransferToEmployee]
         Comments                = ""
         Param.CallNotifyText    = "Consultative transfer from {@UserName}..."
         RequiredField.'Phone #' = "?*"
         BusComp                 = "Employee"
         Param.PhoneNumber       = "{Phone #:Lookup}"

   [Command:MakeCallDummy]
      Profile       = ""
      DeviceCommand = "MakeCall"
      Comments      = ""
      Description   = "Make call"
      CmdData       = ""
      Hidden        = "TRUE"

   [Command:MakeCallToContactHome]
      Profile       = ""
      DeviceCommand = "MakeCall"
      Comments      = ""
      Description   = "Make call to contact home"
      CmdData       = "MakeCallToContactHome"
      Hidden        = "TRUE"

      [CmdData:MakeCallToContactHome]
         OnField                      = "Home Phone #"
         Comments                     = ""
         RequiredField.'Home Phone #' = "?*"
         BusComp                      = "Contact"
         Param.PhoneNumber            = "{Home Phone #:Lookup}"

   [Command:MakeCallToWebCollabCustomer]
      Profile        = ""
      DeviceCommand  = "MakeCall"
      AllViews       = "FALSE"
      Comments       = ""
      View           = "Web Collab All Activities View"
      Description    = "Make call to web customer"
      CmdData        = "MakeCallToWebCollabCustomer"
      Hidden         = "TRUE"
      ActiveWorkType = "Web Collaboration"

      [CmdData:MakeCallToWebCollabCustomer]
         RequiredField.'Call Me Phone No.' = "?*"
         Comments                          = ""
         Param.ParentWorkItemID            = ""{@SelectedWorkItem:WorkItemID}"
         BusObj                            = "Action"
         BusComp                           = "Action"
         Param.PhoneNumber                 = "{Call Me Phone No.::Lookup}"

   [Command:RetrieveCallPBX]
      Profile           = ""
      DeviceCommand     = "RetrieveCall"
      MenuPosition      = "70.1"
      Comments          = ""
      Title             = "Alternate (Phone)"
      CmdData           = "RetrieveCallPBX"
      Hidden            = "TRUE"
      CmdChannelOnFocus = "TRUE"

      [CmdData:RetrieveCallPBX]
         Param.TrackingID = "{@SelectedWorkItem:DriverWorkTrackID}"
         Comments         = ""
```

```
[Command:RetrieveWorkGroup]
   Profile           = ""
   DeviceCommand     = "ReconnectCall"
   Comments          = ""
   Description       = "Reconnect work"
   CmdData           = ""
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

[Command:AgentBusy]
   Profile       = ""
   DeviceCommand = "ChangeBusyState"
   MenuPosition  = "50.4"
   Comments      = ""
   Title         = "Work Mode : Busy"
   Description   = "Set AgentWorkMode to Busy"
   HotKey        = "Ctrl+F5"
   CmdData       = ""

[Command:AgentWorkMode]
   SubCommand_1 = "AgentReady"
   SubCommand_2 = "AgentNotReady"
   SubCommand_3 = "AgentOtherWork"
   SubCommand_4 = "AgentAfterCallWork"
   SubCommand_5 = "AgentBusy"
   Profile      = ""
   MenuPosition = "50"
   Comments     = ""
   Title        = "Work Mode"
   Description  = "Work mode"
   CmdData      = ""

[Command:AnswerCallGroup]
   Profile       = ""
   DeviceCommand = "AnswerCall"
   Comments      = ""
   CmdData       = ""
   Hidden        = "TRUE"

[Command:ConsultativeTransferToCurrentPhone]
   Profile           = ""
   DeviceCommand     = "TransferInit"
   Comments          = ""
   Description       = "Consultative transfer to "{@Phone}""
   CmdData           = "ConsultativeTransferToCurrentPhone"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConsultativeTransferToCurrentPhone]
      AttachContext        = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Consultative transfer from {@UserName} ..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:ConsultativeTransferToPhone]
   Profile           = ""
   DeviceCommand     = "TransferInit"
   Comments          = ""
   Description       = "Consultative transfer to "{@Phone}""
   CmdData           = "ConsultativeTransferToPhone"
   OnEditControl     = "TRUE"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"
   [CmdData:ConsultativeTransferToPhone]
      AttachContext        = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Consultative transfer from {@UserName} ..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:MakeCallGroupInLocalMenu]
   SubCommand_1  = "MakeCallToSRContact"
   SubCommand_2  = "MakeCallToSROwner"
   SubCommand_3  = "MakeCallToEmployeeHome"
   SubCommand_4  = "MakeCallToEmployee"
   SubCommand_5  = "MakeCallToContactHome"
   SubCommand_6  = "MakeCallToContact"
   SubCommand_7  = "MakeCallToAccount"
   SubCommand_8  = "MakeCallToCampaignContactHome"
   SubCommand_9  = "MakeCallToCampaignContact"
   SubCommand_10 = "MakeCallToExtension"
   SubCommand_11 = "MakeCallToActivityContact"
   SubCommand_12 = "MakeCallToCurrentPhone"
   Profile       = ""
   MenuPosition  = "5"
   Comments      = ""
   Title         = "Make Call"
   LocalMenu     = "TRUE"
   CmdData       = ""
   Hidden        = "TRUE"

[Command:MakeCallToPhone]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to "{@Phone}""
   CmdData       = "MakeCallToPhone"
   OnEditControl = "TRUE"
   Hidden        = "TRUE"

   [CmdData:MakeCallToPhone]
      Comments             = ""
      Param.CallVariable2  = "9782758589"
      Param.CallNotifyText = "Call from {@UserName}..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:NotReadyGroupSetToNotReady]
   SubCommand_1 = "NotReadyForPhoneSetToNotReady"
   SubCommand_2 = "NotReadyForPhoneSetToReadyNoPopup"
   Profile      = ""
   Comments     = ""
   Description  = "Set to not ready"
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:SignOffGroupInMenu]
   SubCommand_1 = "LogoutFromPBX"
   Profile      = ""
   MenuPosition = "22"
   Comments     = ""
   Title        = "Log out"
   Description  = "Log out"
   CmdData      = ""
```

```
[Command:UpdateDashboard]
   SubCommand_1 = "UpdateDashboardFromContact"
   SubCommand_2 = "UpdateDashboardFromSR"
   SubCommand_3 = "UpdateDashboardFromActivity"
   SubCommand_4 = "UpdateDashboardFromCampaign"
   Profile      = ""
   FilterSpec   = "[@SelectedWorkItem:MediaType] = 'Voice'"
   Comments     = ""
   HotKey       = "Alt+C"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:AssociateAccount]
   Profile       = ""
   DeviceCommand = "@Associate"
   Comments      = ""
   CmdData       = "AssociateAccount"
   Hidden        = "TRUE"

   [CmdData:AssociateAccount]
      Param."Account Id" = "{Id}"
      Comments           = ""
      BusComp            = "Account"

[Command:ConferenceTransferToPopupEmployee]
   Profile           = ""
   DeviceCommand     = "ConferenceInit"
   Comments          = ""
   Title             = "Conference Transfer"
   Description       = "Conference transfer to employee selected from popup list"
   CmdData           = "ConferenceTransferToPopupEmployee"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConferenceTransferToPopupEmployee]
      SelectBusComp        = "Employee"
      AttachContext        = "TRUE"
      SelectParam          = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Conference transfer from {@UserName}..."
      SelectBusObj         = "Employee"
      SelectApplet         = "ACD Transfer Call Applet"
      SelectTitle          = "Begin Conference Transfer to:"
      Param.PhoneNumber    = "[Phone #:Lookup]"

[Command:ConferenceTransferToSROwner]
   Profile           = ""
   DeviceCommand     = "ConferenceInit"
   Comments          = ""
   Description       = "Conference transfer to service request owner"
   CmdData           = "ConferenceTransferToSROwner"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConferenceTransferToSROwner]
      RequiredField.'Owner Phone' = "?*"
      AttachContext               = "TRUE"
      Comments                    = ""
      Param.CallNotifyText        = "Conference transfer from {@UserName} about SR
{Id}..."
      BusComp                     = "Service Request"
      Param.PhoneNumber           = "{Owner Phone:Lookup}"

[Command:GetNextMsgInMenu]
```

```
   Profile       = ""
   MenuPosition  = "60.3"
   Comments      = ""
   Title         = "Next Message"
   Description   = "Show next message"
   HotKey        = "Alt+N"
   CmdData       = ""
   ServiceMethod = "Communications Client.ShowNextMessage"

[Command:MakeCallGroupInMenu]
   SubCommand_1  = "MakeCallToPhone"
   SubCommand_2  = "MakeCallToSelectedPhone"
   SubCommand_3  = "MakeCallToSRContact"
   SubCommand_4  = "MakeCallToSROwner"
   SubCommand_5  = "MakeCallToEmployeeHome"
   SubCommand_6  = "MakeCallToEmployee"
   SubCommand_7  = "MakeCallToContactHome"
   SubCommand_8  = "MakeCallToContact"
   SubCommand_9  = "MakeCallToAccount"
   SubCommand_10 = "MakeCallToCampaignContactHome"
   SubCommand_11 = "MakeCallToCampaignContact"
   SubCommand_12 = "MakeCallToExtension"
   SubCommand_13 = "MakeCallToActivityContact"
   SubCommand_14 = "MakeCallToCurrentPhone"
   SubCommand_15 = "MakeCallToPopupEmployee"
   Profile       = ""
   MenuPosition  = "6"
   Comments      = ""
   Title         = "Make Call"
   HotKey        = "F7"
   CmdData       = ""

[Command:MuteTransferCallToEmployee]
   Profile           = ""
   DeviceCommand     = "TransferMute"
   Comments          = ""
   Description       = "Blind transfer call to employee"
   CmdData           = "MuteTransferCallToEmployee"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:MuteTransferCallToEmployee]
      Comments                = ""
      Param.CallNotifyText    = "Blind transfer from {@UserName}..."
      RequiredField.'Phone #' = "?*"
      BusComp                 = "Employee"
      Param.PhoneNumber       = "{Phone #:Lookup}"

[Command:MuteTransferCallToExtension]
   Profile           = ""
   DeviceCommand     = "TransferMute"
   Comments          = ""
   Description       = "Blind transfer call to extension"
   CmdData           = "MuteTransferCallToExtension"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:MuteTransferCallToExtension]
      Comments             = ""
      Param.CallNotifyText = "Blind transfer from {@UserName}..."
      BusComp              = "Telephone Status"
      Param.PhoneNumber    = "{Extension}"

[Command:RefreshDashboardAtCallChange]
```

```
   Profile       = ""
   FilterSpec    = "[@SelectedWorkItem:ChannelType] = 'Voice' AND
[@SelectedWorkItem:ContactId] IS NOT NULL"
   Comments      = ""
   CmdData       = "RefreshDashboardAtCallChange"
   ServiceMethod = "Persistent Customer Dashboard.Update Dashboard from CTI"
   Hidden        = "TRUE"

   [CmdData:RefreshDashboardAtCallChange]
      ServiceParam.Value = "{@SelectedWorkItem:ContactId}"
      Comments           = ""
      ServiceParam.Field = "Id"

[Command:UnHoldCall]
   Profile           = ""
   DeviceCommand     = "UnHoldCall"
   Comments          = ""
   CmdData           = "UnHoldCall"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:UnHoldCall]
      Param.TrackingID = "{@SelectedWorkItem:DriverWorkTrackID}"
      Comments         = ""

[Command:ViewWorkObjectInMenu]
   Profile       = ""
   DeviceCommand = "@ViewWorkObject"
   MenuPosition  = "8"
   Comments      = ""
   Title         = "View Work Item"
   HotKey        = "Shift+F8"
   CmdData       = "ViewWorkObject"

   [CmdData:ViewWorkObject]
      Param.View = "Activity Attachment View"
      Comments   = ""

[Command:AcceptWorkGroup]
   SubCommand_1       = "AnswerCallGroup"
   MultiActiveCmdIcon = "misc_work.gif"
   IndicateActiveCmd  = "TRUE"
   Profile            = ""
   Comments           = ""
   Description        = "Accept work item"
   CmdData            = ""
   Hidden             = "TRUE"

[Command:ConferenceTransferGroup]
   SubCommand_1 = "CompleteConferenceTransferToPhone"
   SubCommand_2 = "ConferenceTransferToPhone"
   SubCommand_3 = "ConferenceTransferToSROwner"
   SubCommand_4 = "ConferenceTransferToEmployee"
   SubCommand_5 = "ConferenceTransferToExtension"
   SubCommand_6 = "ConferenceTransferToCurrentPhone"
   SubCommand_7 = "ConferenceTransferToPopupEmployee"
   Profile      = ""
   Comments     = ""
   Description  = "Conference transfer work item"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:ForwardWorkGroup]
   Profile           = ""
   DeviceCommand     = "ToggleForward"
   Comments          = ""
   CmdData           = ""
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

[Command:LogoutFromPBX]
   Profile       = ""
   DeviceCommand = "LogOut"
   MenuPosition  = "22.1"
   Comments      = ""
   Title         = "Log Out (Phone)"
   CmdData       = "LogoutFromPBX"

   [CmdData:LogoutFromPBX]
      Param.ReasonCode = "1"
      Comments         = ""

[Command:MakeCallToCurrentPhone]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to "{@Phone}""
   CmdData       = "MakeCallToCurrentPhone"
   Hidden        = "TRUE"

   [CmdData:MakeCallToCurrentPhone]
      AttachContext        = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Call from {@UserName}..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:MakeCallToSROwner]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to service request owner"
   CmdData       = "MakeCallToSROwner"
   Hidden        = "TRUE"

   [CmdData:MakeCallToSROwner]
      RequiredField.'Owner Phone' = "?*"
      AttachContext               = "TRUE"
      Comments                    = ""
      Param.CallNotifyText        = "Call from {@UserName} about SR {Id}..."
      BusComp                     = "Service Request"
      Param.PhoneNumber           = "{Owner Phone:Lookup}"

[Command:NotReadyGroupSetToReady]
   SubCommand_1 = "NotReadyForPhoneSetToReady"
   Profile      = ""
   Comments     = ""
   Description  = "Set to ready"
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:SendSMSGroup]
   Profile       = ""
   DeviceCommand = "@CreatePopupFrame"
   Comments      = ""
   Description   = "Send wireless message"
   HotKey        = "Ctrl+F9"
   CmdData       = "SendSMSGroup"
   Hidden        = "TRUE"

   [CmdData:SendSMSGroup]
      Comments          = ""
      PostInvokeCommand = "SendSMS"

[Command:SuspendDeselectedCall]
   Profile       = ""
   FilterSpec    = "[@DeselectedWorkItem:WorkState] = 'Active'"
   DeviceCommand = "SuspendDeselectedCall"
   Comments      = ""
   CmdData       = "SuspendDeselectedCall"
   Hidden        = "TRUE"

   [CmdData:SuspendDeselectedCall]
      Param.TrackingID = "{@DeselectedWorkItem:DriverWorkTrackID}"
      Comments         = ""

[Command:UpdateDashboardFromCampaign]
   Profile       = ""
   Comments      = ""
   CmdData       = "UpdateDashboardFromCampaign"
   ServiceMethod = "Persistent Customer Dashboard.Update Dashboard from CTI"
   Hidden        = "TRUE"

   [CmdData:UpdateDashboardFromCampaign]
      ServiceParam.Value         = "{Contact Id}"
      RequiredField.'Contact Id' = "?*"
      WorkTrackingObj.ContactId  = "{Contact Id}"
      Comments                   = ""
      ServiceParam.Field         = "Id"
      BusComp                    = "Campaign List Contact"

[Command:AnswerCallInMenu]
   SubCommand_1 = "AnswerCallGroup"
   Profile      = ""
   MenuPosition = "1"
   Comments     = ""
   Title        = "Answer Call"
   Description  = "Answer call"
   HotKey       = "F6"
   CmdData      = ""

[Command:AssociateSR]
   Profile       = ""
   DeviceCommand = "@Associate"
   Comments      = ""
   CmdData       = "AssociateSR"
   Hidden        = "TRUE"

   [CmdData:AssociateSR]
      Param."Contact Id"     = "{Contact Id}"
      Param."Account Id"     = "{Account Id}"
      Comments               = ""
      Param."Activity SR Id" = "{Id}"
      BusComp                = "Service Request"

[Command:ConsultativeTransferGroupInLocalMenu]
   SubCommand_1 = "ConsultativeTransferToSROwner"
   SubCommand_2 = "ConsultativeTransferToEmployee"
   SubCommand_3 = "ConsultativeTransferToExtension"
   SubCommand_4 = "ConsultativeTransferToCurrentPhone"
   Profile      = ""
   MenuPosition = "4"
   Comments     = ""
   Title        = "Consultative Transfer"
   LocalMenu    = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:MakeCallToActivityContact]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to activity contact"
   CmdData       = "MakeCallToActivityContact"
   Hidden        = "TRUE"

   [CmdData:MakeCallToActivityContact]
      Param.RowId                   = "{Id}"
      AttachContext                 = "TRUE"
      Param.ContactPhone            = "{Contact Phone}"
      Comments                      = ""
      Param.CallNotifyText          = "Call from {@UserName}..."
      RequiredField.'Contact Phone' = "?*"
      BusComp                       = "Action"
      Param.PhoneNumber             = "{Contact Phone:Lookup}"

[Command:MuteTransferCallToSROwner]
   Profile           = ""
   DeviceCommand     = "TransferMute"
   Comments          = ""
   Description       = "Blind transfer call to service request owner"
   CmdData           = "MuteTransferCallToSROwner"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:MuteTransferCallToSROwner]
      RequiredField.'Owner Phone' = "?*"
      AttachContext               = "TRUE"
      Comments                    = ""
      Param.CallNotifyText        = "Blind transfer from {@UserName} about SR {Id}..."
      BusComp                     = "Service Request"
      Param.PhoneNumber           = "{Owner Phone:Lookup}"

[Command:MuteTransferGroupInMenu]
   SubCommand_1 = "MuteTransferCallToPopupEmployee"
   SubCommand_2 = "MuteTransferCallToCurrentPhone"
   Profile      = ""
   MenuPosition = "3"
   Comments     = ""
   Title        = "Blind Transfer"
   HotKey       = "Alt+F7"
   CmdData      = ""

[Command:SimGroupInMenu]
   SubCommand_1 = "SimCallFound"
   SubCommand_3 = "SimCampaignCall"
   Profile      = ""
   MenuPosition = "0"
   Comments     = ""
   Title        = "Simulate call"
   Description  = "Simulate call"
   CmdData      = ""

[Command:AgentOtherWork]
   Profile       = ""
   DeviceCommand = "AgentWorkMode2"
   MenuPosition  = "50.2"
   Comments      = ""
   Title         = "Work Mode : Other Work"
   Description   = "Set AgentWorkMode to Other-Work"
   HotKey        = "Ctrl+F3"
   CmdData       = ""

[Command:MakeCallToContact]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to contact"
   CmdData       = "MakeCallToContact"
   Hidden        = "TRUE"

   [CmdData:MakeCallToContact]
      RequiredField.'Work Phone #' = "?*"
      Comments                     = ""
      BusComp                      = "Contact"
      Param.PhoneNumber            = "{Work Phone #:Lookup}"

[Command:SendPageGroup]
   Profile       = ""
   DeviceCommand = "@CreatePopupFrame"
   Comments      = ""
   Description   = "Send page"
   HotKey        = "Shift+F9"
   CmdData       = "SendPageGroup"
   Hidden        = "TRUE"

   [CmdData:SendPageGroup]
      Comments          = ""
      PostInvokeCommand = "SendPage"

[Command:SignOnGroup]
   SubCommand_1 = "LoginToPBX"
   Profile      = ""
   Comments     = ""
   Description  = "Log in"
   ExecuteAll   = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:CompleteConferenceTransferToPhone]
   Profile           = ""
   DeviceCommand     = "ConferenceComplete"
   Comments          = ""
   CmdData           = ""
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

[Command:MakeCallToEmployeeHome]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to employee home"
   CmdData       = "MakeCallToEmployeeHome"
   Hidden        = "TRUE"

   [CmdData:MakeCallToEmployeeHome]
      OnField                      = "Home Phone #"
      Comments                     = ""
      RequiredField.'Home Phone #' = "?*"
      BusComp                        = "Employee"
      Param.PhoneNumber              = "{Home Phone #:Lookup}"

[Command:AgentReady]
   Profile       = ""
   DeviceCommand = "ChangeReadyState"
   MenuPosition  = "50.0"
   Comments      = ""
   Title         = "Work Mode : Ready"
   Description   = "Set AgentWorkMode to Ready"
   HotKey        = "Ctrl+F1"
   CmdData       = ""

[Command:AssociateGroupInMenu]
   SubCommand_1 = "AssociateAccount"
   SubCommand_2 = "AssociateContact"
   SubCommand_3 = "AssociateSR"
   SubCommand_4 = "AssociateDefect"
   SubCommand_5 = "AssociateCampaign"
   SubCommand_6 = "AssociateOppty"
   SubCommand_7 = "AssociateConsumer"
   Profile      = ""
   MenuPosition = "2"
   Comments     = ""
   Title        = "Associate"
   HotKey       = "F8"
   CmdData      = ""

[Command:AssociateOppty]
   Profile       = ""
   DeviceCommand = "@Associate"
   Comments      = ""
   CmdData       = "AssociateOppty"
   Hidden        = "TRUE"

   [CmdData:AssociateOppty]
      Param."Opportunity Id" = "{Id}"
      Param."Account Id"     = "{Account Id}"
      Comments               = ""
      BusComp                = "Opportunity"

[Command:BlindTransferGroup]
   SubCommand_1 = "MuteTransferCallToPhone"
   SubCommand_2 = "MuteTransferCallToSROwner"
   SubCommand_3 = "MuteTransferCallToEmployee"
   SubCommand_4 = "MuteTransferCallToExtension"
   SubCommand_5 = "MuteTransferCallToCurrentPhone"
   Profile      = ""
   Comments     = ""
   Description  = "Mute transfer work item"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:ConferenceTransferGroupInMenu]
   SubCommand_1 = "CompleteConferenceTransferToPhone"
   SubCommand_2 = "ConferenceTransferToPhone"
   SubCommand_3 = "ConferenceTransferToSROwner"
   SubCommand_4 = "ConferenceTransferToEmployee"
   SubCommand_5 = "ConferenceTransferToExtension"
   SubCommand_6 = "ConferenceTransferToCurrentPhone"
   SubCommand_7 = "ConferenceTransferToPopupEmployee"
   Profile      = ""
   MenuPosition = "4"
   Comments     = ""
   Title        = "Conference Transfer"
   HotKey       = "Ctrl+F7"
   CmdData      = ""

[Command:ConsultativeTransferToExtension]
   Profile           = ""
   DeviceCommand     = "TransferInit"
   Comments          = ""
   Description       = "Consultative transfer to extension"
   CmdData           = "ConsultativeTransferToExtension"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConsultativeTransferToExtension]
      Comments             = ""
      Param.CallNotifyText = "Consultative transfer from {@UserName}..."
      BusComp              = "Telephone Status"
      Param.PhoneNumber    = "{Extension}"

[Command:ConsultativeTransferToPopupEmployee]
   Profile           = ""
   DeviceCommand     = "TransferInit"
   Comments          = ""
   Title             = "Consultative Transfer"
   Description       = "Consultative transfer to employee selected from popup list"
   CmdData           = "ConsultativeTransferToPopupEmployee"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ConsultativeTransferToPopupEmployee]
      SelectBusComp        = "Employee"
      AttachContext        = "TRUE"
      SelectParam          = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Consultative transfer from {@UserName}..."
      SelectBusObj         = "Employee"
      SelectApplet         = "ACD Transfer Call Applet"
      SelectTitle          = "Begin Consultative Transfer to:"
      Param.PhoneNumber    = "[Phone #:Lookup]"

[Command:MakeCallToPopupEmployee]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Title         = "Make Call"
   Description   = "Make call to employee selected from popup list"
   CmdData       = "MakeCallToPopupEmployee"
   Hidden        = "TRUE"

   [CmdData:MakeCallToPopupEmployee]
      SelectBusComp        = "Employee"
      AttachContext        = "TRUE"
      SelectParam          = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Call from {@UserName}..."
      SelectBusObj         = "Employee"
      SelectApplet         = "ACD Transfer Call Applet"
      SelectTitle          = "Make Call to:"
      Param.PhoneNumber    = "[Phone #:Lookup]"

[Command:MuteTransferCallToCurrentPhone]
   Profile           = ""
   DeviceCommand     = "TransferMute"
   Comments          = ""
   Title             = "Blind Transfer"
   Description       = "Blind transfer to "{@Phone}""
   CmdData           = "MuteTransferCallToCurrentPhone"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:MuteTransferCallToCurrentPhone]
      AttachContext        = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Blind transfer from {@UserName}..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:MuteTransferGroupInLocalMenu]
   SubCommand_1 = "MuteTransferCallToSROwner"
   SubCommand_2 = "MuteTransferCallToEmployee"
   SubCommand_3 = "MuteTransferCallToExtension"
   SubCommand_4 = "MuteTransferCallToCurrentPhone"
   SubCommand_5 = "MuteTransferCallToEmployee"
   Profile      = ""
   MenuPosition = "2"
   Comments     = ""
   Title        = "Blind Transfer"
   LocalMenu    = "TRUE"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:ResumeSelectedCall]
   Profile       = ""
   FilterSpec    = "[@SelectedWorkItem:WorkState] = 'Suspended'"
   DeviceCommand = "ResumeSelectedCall"
   Comments      = ""
   CmdData       = "ResumeSelectedCall"
   Hidden        = "TRUE"

   [CmdData:ResumeSelectedCall]
      Param.TrackingID = "{@SelectedWorkItem:DriverWorkTrackID}"
      Comments         = ""

[Command:UpdateDashboardFromContact]
   Profile       = ""
   Comments      = ""
   CmdData       = "UpdateDashboardFromContact"
   ServiceMethod = "Persistent Customer Dashboard.Update Dashboard from CTI"
   Hidden        = "TRUE"

   [CmdData:UpdateDashboardFromContact]
      ServiceParam.Value        = "{Id}"
      WorkTrackingObj.ContactId = "{Id}"
      Comments                  = ""
      ServiceParam.Field        = "Id"
      BusComp                   = "Contact"

[Command:AssociateCampaign]
   Profile       = ""
   DeviceCommand = "@Associate"
   Comments      = ""
   CmdData       = "AssociateCampaign"
   Hidden        = "TRUE"

   [CmdData:AssociateCampaign]
      Comments            = ""
      Param."Campaign Id" = "{Id}"
      BusComp             = "Campaign"
```

```
[Command:GetPreviousMsgInMenu]
   Profile       = ""
   MenuPosition  = "60.2"
   Comments      = ""
   Title         = "Previous Message"
   Description   = "Show previous message"
   HotKey        = "Alt+P"
   CmdData       = ""
   ServiceMethod = "Communications Client.ShowPreviousMessage"

[Command:MakeCallToCampaignContactHome]
   Profile       = ""
   DeviceCommand = "MakeCall"
   Comments      = ""
   Description   = "Make call to campaign contact home"
   CmdData       = "MakeCallToCampaignContactHome"
   Hidden        = "TRUE"

   [CmdData:MakeCallToCampaignContactHome]
      OnField                      = "Home Phone #"
      Comments                     = ""
      Param.CampaignName           = "{Campaign Name}"
      Param.ContactId              = "{Contact Id}"
      Param.CampaignId             = "{Campaign Id}"
      Param.CampaignPhone          = "{Home Phone #}"
      RequiredField.'Home Phone #' = "?*"
      BusComp                      = "Campaign List Contact"
      Param.PhoneNumber            = "{Home Phone #:Lookup}"

[Command:MuteTransferCallToPhone]
   Profile           = ""
   DeviceCommand     = "TransferMute"
   Comments          = ""
   Description       = "Blind transfer to "{@Phone}""
   CmdData           = "MuteTransferCallToPhone"
   OnEditControl     = "TRUE"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:MuteTransferCallToPhone]
      AttachContext        = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Blind transfer from {@UserName}..."
      RequiredField.@Phone = "?*"
      Param.PhoneNumber    = "{@Phone:PhoneTypeLookup}"

[Command:MuteTransferCallToPopupEmployee]
   Profile           = ""
   DeviceCommand     = "TransferMute"
   Comments          = ""
   Title             = "Blind Transfer"
   Description       = "Blind transfer call to employee selected from popup list"
   CmdData           = "MuteTransferCallToPopupEmployee"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:MuteTransferCallToPopupEmployee]
      SelectBusComp        = "Employee"
      AttachContext        = "TRUE"
      SelectParam          = "TRUE"
      Comments             = ""
      Param.CallNotifyText = "Blind transfer from {@UserName}..."
      SelectBusObj         = "Employee"
      SelectApplet         = "ACD Transfer Call Applet"
      SelectTitle          = "Begin Blind Transfer to:"
      Param.PhoneNumber    = "[Phone #:Lookup]"

[Command:ReleasePhoneCall]
   Profile           = ""
   DeviceCommand     = "ReleaseCall"
   Comments          = ""
   CmdData           = "ReleasePhoneCall"
   Hidden            = "TRUE"
   CmdChannelOnFocus = "TRUE"

   [CmdData:ReleasePhoneCall]
      Param.TrackingID = "{@SelectedWorkItem:DriverWorkTrackID}"
      Comments         = ""

[Command:ResumeWorkGroup]
   SubCommand_1 = "UnHoldCall"
   Profile      = ""
   Comments     = ""
   Description  = "Resume work item"
   CmdData      = ""
   Hidden       = "TRUE"

[Command:UpdateDashboardFromSR]
   Profile       = ""
   Comments      = ""
   CmdData       = "UpdateDashboardFromSR"
   ServiceMethod = "Persistent Customer Dashboard.Update Dashboard from CTI"
   Hidden        = "TRUE"

   [CmdData:UpdateDashboardFromSR]
      ServiceParam.Value         = "{Contact Id}"
      RequiredField.'Contact Id' = "?*"
      WorkTrackingObj.ContactId  = "{Contact Id}"
      Comments                   = ""
      ServiceParam.Field         = "Id"
      BusComp                    = "Service Request"

[EventHandler:OutboundActivityContactCall]
   Profile      = ""
   Filter.RowId = "?*"
   Comments     = ""
   Order        = "7"
   Response     = ""
   DeviceEvent  = ""

[EventHandler:OutboundCampaignContactCall]
   Profile              = ""
   Comments             = ""
   Filter.CampaignPhone = "?*"
   Order                = "9"
   Response             = ""
   DeviceEvent          = ""

[EventHandler:InboundCampaignCall]
   Filter.DNIS = "?*"
   Profile     = ""
   Comments    = ""
   Order       = "6"
   Response    = "InboundCampaignCall"
   DeviceEvent = "EventAnswer"

   [EventResponse:InboundCampaignCall]
```

```
      QueryBusObj           = "Campaign"
      QuerySpec             = "[DNIS]='{DNIS}'"
      Comments              = ""
      SingleView            = "Campaign Detail"
      QuerySpec2            = "[Work Phone #]='{ANI}'"
      QueryBusComp2         = "Campaign List Contact"
      QueryBusComp          = "Campaign"
      InvokeMethodIfNoData2 = "NewRecord"

[EventHandler:EventUpdateCurCallData]
   Profile     = ""
   Comments    = ""
   Order       = "14"
   Response    = ""
   DeviceEvent = "EventUpdateCurCallData"

[EventHandler:InboundCallReceived]
   Profile               = ""
   Comments              = ""
   Order                 = "2"
   Response              = "OnInboundCallReceived"
   Filter.CallVariable2 = "*"
   DeviceEvent           = "EventAnswer"

   [EventResponse:OnInboundCallReceived]
      ServiceParam.Value = "{CallVariable2}"
      QueryBusObj        = "Contact"
      QuerySpec          = "'Work Phone #'='{CallVariable2}'"
      Comments           = ""
      Log                = "LogIncomingCallContactNotFound"
      SingleLog          = "LogIncomingCallContactFound"
      MultiView          = "Contact List View"
      SingleView         = "Service Contact Detail View"
      ServiceParam.Field = "Work Phone #"
      UseCtxData         = "TRUE"
      ServiceMethod      = "Persistent Customer Dashboard.Update Dashboard from CTI"
      QueryBusComp       = "Contact"
      FindField.CSN      = "Ask Caller"
      FindDialog         = "Service Request"

      [EventLog:LogIncomingCallContactNotFound]
         AfterWork.'ACD Call Duration' = "{@WorkDuration}"
         Comments                      = ""
         Display                       = "TRUE"
         LogField.Description          = "EventLog:LogIncomingCallContactNotFound"
         LogField.Comment              = "Account: {CallVariable2};TIme: {@Now}"
         BusObj                        = "Action"
         BusComp                       = "Action"
         LogField.Type                 = "Call - Inbound"

      [EventLog:LogIncomingCallContactFound]
         AfterWork.'ACD Call Duration'  = "{@WorkDuration}"
         Comments                       = ""
         AfterWork.'Started'            = "{@WorkStartTime}"
         Display                        = "TRUE"
         LogField.Description           = "EventLog:LogIncomingCallContactFound"
         AfterWork.'Planned'            = "{@WorkStartTime}"
         LogField.Comment               = "Account: {CallVariable2};
WorkDuration:{@WorkDuration}; Time:{@Now}"
         BusObj                         = "Action"
         AfterWork.'Done'               = "{@Now}"
         BusComp                        = "Action"
         AfterWork.'Planned Completion' = "{@Now}"
         LogField.Type                  = "Call - Inbound"
```

```
[EventHandler:OutboundCampaignCall]
   Profile       = ""
   Comments      = ""
   Order         = "5"
   Filter.CampID = "?*"
   Response      = "OutboundCampaignCall"
   DeviceEvent   = ""

   [EventResponse:OutboundCampaignCall]
      QueryBusObj  = "Campaign"
      QuerySpec    = "Id='{CampID}'"
      Comments     = ""
      SingleView   = "Campaign Detail"
      QueryBusComp = "Campaign"

[EventHandler:InsideCallReceived]
   Filter.ANI  = "?*"
   Profile     = ""
   Comments    = ""
   Order       = "4"
   Response    = "OnInsideCallReceived"
   DeviceEvent = "EventAnswer"

   [EventResponse:OnInsideCallReceived]
      Comments   = ""
      UseCtxData = "TRUE"

[EventHandler:EventTransferred]
   Profile     = ""
   Comments    = ""
   Order       = "13"
   Response    = ""
   DeviceEvent = "EventTransferred"

[EventHandler:InboundConsumerCall]
   Profile                = ""
   Comments               = ""
   Order                  = "3"
   Response               = "InboundConsumerCall"
   Filter.CollectedDigits = "?*"
   DeviceEvent            = "EventAnswer"

   [EventResponse:InboundConsumerCall]
      QueryBusObj   = "Consumer"
      QuerySpec     = "[CSN]='{CollectedDigits}'"
      Comments      = ""
      SingleView    = "Consumer Detail View"
      QueryBusComp  = "Consumer"
      FindField.CSN = "{CollectedDigits}"
      FindDialog    = "Consumer"

[EventHandler:AssignToFieldChanged]
   Profile     = ""
   Comments    = ""
   Order       = "51"
   Response    = ""
   DeviceEvent = "@RuntimeEvent"

[EventHandler:EventAgentStatistics]
   Profile     = ""
   Comments    = ""
   Order       = "15"
   Response    = ""
   DeviceEvent = "EventAgentStatistics"

[EventResponse:EventTransferred]
   Comments = ""
   Command  = "AssignToFieldChanged"

[EventResponse:EventUpdateCurCallData]
   Comments = ""
   Command  = "AssignToFieldChanged"

[EventResponse:AssignToFieldChanged]
   Comments = ""
   Command  = "AssignToFieldChanged"
```

INDEX