Siebel Communications Server Administration Guide

Version 7.8, Rev. B
November 2005

Siebel Systems, Inc., 2207 Bridgepointe Parkway, San Mateo, CA 94404

Printed in the United States of America

Contents

Chapter 4: Configuring Session Communications

Chapter 5: Configuring Events and Commands

Chapter 6: Configuring User Interface Elements

Chapter 7: Configuring Advanced Communications Features

Chapter 8: Administering Siebel Communications Server

Chapter 9: Configuring Communications Templates

Chapter 10: Defining Outbound Communication Requests

Chapter 11: Communications Operations for End Users

Chapter 12: Using Siebel CTI Connect

Chapter 13: Using Email, Fax, and Other Systems

Appendix A: Developing a Communications Driver

Appendix D: Upgrading from Release 6.x

Index

1 What's New in This Release

What's New in Siebel Communications Server Administration Guide, Version 7.8, Rev. B

Table 1 lists changes described in this version of the documentation to support release 7.8 of the software.

Table 1. New Features in Siebel Communications Server Administration Guide, Version 7.8, Rev. B

Topic	Description
Siebel Wireless Messaging	Deleted content. Siebel Wireless Messaging is no longer supported. Content that described how to use Siebel Wireless Messaging has been removed. For more information, see Alert 1182 on Siebel SupportWeb.
"Creating Event Handlers" on page 68	Revised topic. Enhanced procedure that describes how to create an event handler.
"Configuring Automatic Log Out Using Server Script" on page 203	Revised topic. Corrected the server script.

What's New in Siebel Communications Server Administration Guide, Version 7.8, Rev. A

This revision includes editorial and minor content enhancements.

What's New in Siebel Communications Server Administration Guide, Version 7.8

Table 2 lists changes described in this version of the documentation to support release 7.8 of the software.

Table 2. New Features in Siebel Communications Server Administration Guide, Version 7.8

Topic	Description
Siebel Dedicated Web Client renamed as Siebel Developer Web Client	The Siebel Dedicated Web Client has been renamed as the Siebel Developer Web Client. This client type is now supported for administration, development, and troubleshooting usage scenarios only.
"Macros for Parameter Values" on page 182	Two new macros, @ClientHostName and @ClientIP, are available for use with the Communications Server.
"Testing Communications Drivers" on page 422	You can use the Communications Driver Test Engine to test third-party communications drivers or communications drivers that you develop.

2 About Siebel Communications Server

This chapter provides an overview of Siebel Communications Server. It includes the following topics:

What Is Siebel Communications Server?

Siebel Communications Server provides an infrastructure to support several kinds of communications activities for Siebel application users. Custom drivers for third-party products can be created using the Adaptive Communications API:

- **Session-based/interactive communications.** Supports multichannel interactive communications for call or contact center agents who use the communications toolbar to:
  - Make or receive voice calls through computer telephony integration (CTI) supported by third-party CTI middleware packages.
  - Receive inbound email messages as email work items (for Siebel Email Response, using Siebel Universal Queuing).
- **Inbound communications.** Supports integrating to third-party email servers and processing inbound email (when using Siebel Email Response).
- **Outbound communications.** Supports integrating to a variety of third-party communications systems such as email servers to send outbound communications:
  - Supports the Send Email and Send Fax commands for Siebel application users. (Send Page is also available but uses Page Manager rather than Communications Server.)
  - Supports agents sending email replies, for Siebel Email Response.
  - Supports users sending communications content to designated recipients using outbound communication requests. Communication requests can be created and submitted manually through a user interface described in this book, or created and submitted programmatically.

    Several Siebel modules invoke business service methods through workflows to send outbound communications.

NOTE: Your Siebel implementation may not have all the features described in this guide, depending on which software modules you have purchased.

Using Communications Server with Other Siebel Modules

This section provides high-level information about how Siebel Communications Server works with closely related product modules. Several Siebel modules use the Communications Server infrastructure to support certain functionality for the Siebel Business Applications.

Communications Server functionality is not limited to what is described in the following subsections. Many specific features of Siebel Business Applications that interrelate with Siebel Communications Server are discussed throughout this book.

For example, many customers may choose to implement Communications Server in order to support the Send Email or Send Fax commands. In addition, customers using Siebel Call Center may choose to implement Siebel CTI, Siebel Email Response, or Siebel Universal Queuing. The customer dashboard feature and Siebel SmartScript can be used with these modules.

Communications Server and Siebel CTI

Siebel CTI (Computer Telephony Integration) provides voice-channel support for call center agents using Siebel Business Applications. CTI capabilities are provided through integration with third-party CTI middleware packages, such as Intel NetMerge.

Siebel CTI Connect, available from Siebel Systems, Inc., includes Intel NetMerge middleware components. These components include Intel NetMerge Call Processing Software (formerly Dialogic CT Connect) and Intel NetMerge Call Information Manager (formerly Dialogic Call Information Manager.)

Siebel CTI Connect also includes the Siebel communications driver for Intel NetMerge and sample Siebel communications configuration data. Sample communications configurations provided by Siebel Systems contain events and commands that support CTI functionality.

The Communications Session Manager handles CTI communications. Agents are notified of incoming calls through the communications toolbar and can perform a range of call-handling activities using the toolbar and related menu commands.

Telesets are defined and associated with communications agents in the Administration - Communications screen.

Call routing can be handled by Siebel Universal Queuing, by the ACD (Automatic Call Distributor), by the CTI middleware, or by a third-party vendor's queuing engine that has been integrated with Siebel Communications Server. For more information about Siebel CTI Connect, see Chapter 12, "Using Siebel CTI Connect."

Communications Server and Siebel Email Response

Siebel Email Response allows agents to receive and reply to inbound email messages.

Inbound email routing can be handled by Siebel Universal Queuing or by Siebel Assignment Manager, with Siebel Workflow.

If routing is handled using Siebel Universal Queuing, then communications configurations can support email activities for agents, and agents are notified of incoming email messages through the communications toolbar. The sample communications configurations provided by Siebel Systems contain events and commands that support email-handling functionality.

Siebel Systems provides a communications driver that integrates with a variety of email servers that support SMTP and POP3 protocols. Communications administrators also configure response groups and workflows to determine how to process inbound email messages.

The Communications Inbound Receiver and Communications Inbound Processor components receive and process inbound email communications. The Communications Outbound Manager component handles outbound email for agent replies.

For more information about Siebel Email Response, see *Siebel Email Response Administration Guide*.

Communications Server and Siebel Universal Queuing

Siebel Universal Queuing is an optional infrastructure module that routes communications work items to agents. Agents are notified of the work items through the communications toolbar.

Siebel Universal Queuing can route work items of predefined channels such as email (for Siebel Email Response) or voice calls (for Siebel CTI). It can also route work items of new channel types that you implement.

For more information, see *Siebel Universal Queuing Administration Guide*.

Communications Server and Siebel Workflow

Siebel Workflow is a module (part of the base applications) that works closely with Siebel Communications Server. Communications Server business service methods can be invoked from workflow processes to serve a variety of purposes.

Many Siebel modules access outbound communications capabilities this way. Siebel eSales is one example.

Administrators for these products must work with communications administrators to create profiles for the Internet SMTP/POP3 Server communications driver.

Other Communications Server business service methods, such as those for the Communications Client service, can also be accessed from workflow processes.

For more information, see "Using Business Services with Communications Server" on page 216 and Appendix B, "Communications Server Business Services."

About Communications Server Architecture

Figure 1 illustrates the overall architecture of the communications infrastructure for the Siebel Business Applications.

NOTE: The Siebel Developer Web Client is now supported for administration, development, and troubleshooting usage scenarios only.

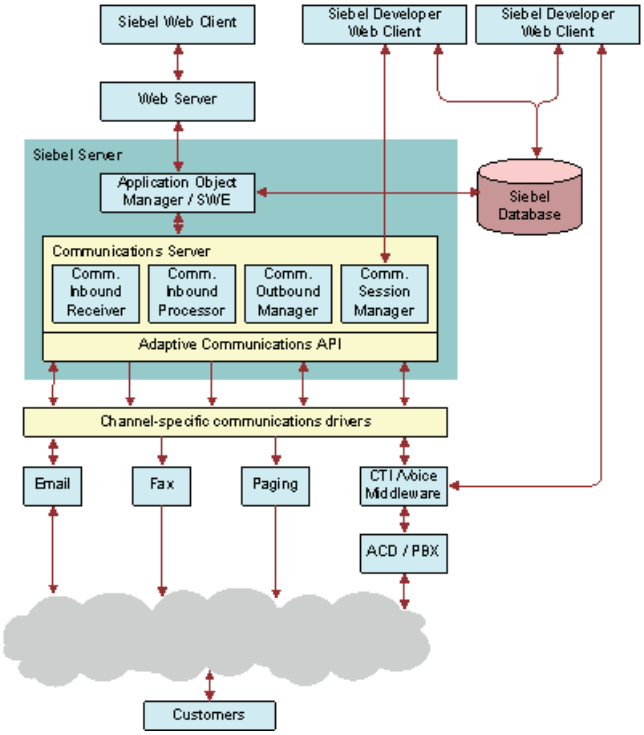

Figure 1. Architecture of Communications Infrastructure

About Communications Configuration Data

The Administration - Communications screen allows administrators to configure several kinds of communications data.

Sample communications configuration data is provided in the Siebel Database. You modify the preconfigured (out of the box) data, or create new data, to customize Siebel Communications Server and the Siebel Business Applications to support the communications channels you are using and to support your business processes.

You can also find the sample communications configuration data in the Siebel Sample Database.

Types of Communications Configuration Data

Figure 2 shows key kinds of configuration data that you work with in the Administration - Communications screen. This data enables you to specify communications functionality within the Siebel Business Applications.

One-to-one, one-to-many, and many-to-many relationships between elements are represented by the connector lines.

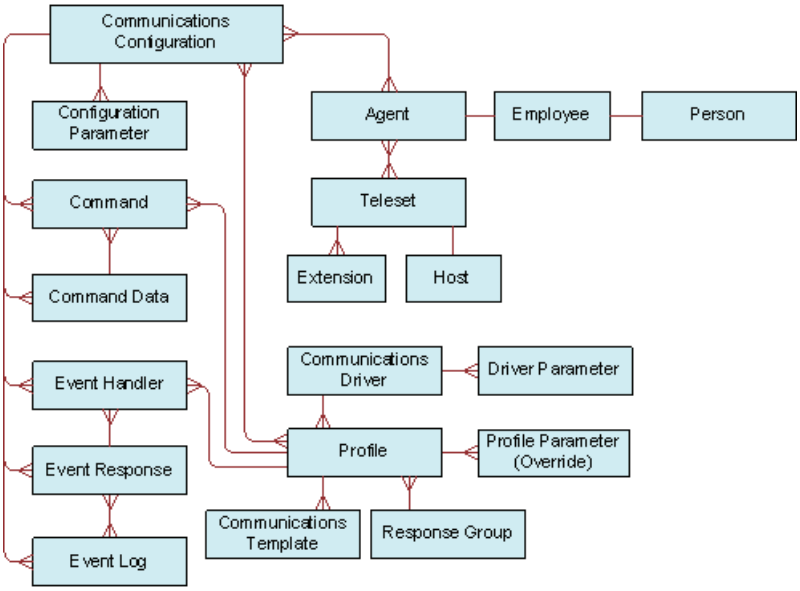

Figure 2. Overview of Communications Configuration Elements

Table 3 on page 22 identifies communications elements that are frequently created in the process of configuring Siebel Communications Server, indicating where the element is created, configured, or specified. Unless otherwise noted, the application views identified are in the Administration - Communications screen. Views in other screens are also mentioned, where applicable.

Table 3 is not comprehensive but attempts to show major elements in relation to other elements. Also see Figure 2 on page 21.

Table 3. Communications Configuration Elements

Configuration Element	Description	How Related	Where Created / Configured
Communications configuration	Set of related data for communications, used for agents working with channels such as voice or email. Includes parameters.	Parent of event handler, event response, event log, command, and command data. Associated with profile and agent.	All Configurations
UQ configuration	Set of related data applicable to Siebel Universal Queuing. For details, see *Siebel Universal Queuing Administration Guide*.	Associated with communications configuration by using configuration parameter UQConfigurationName.	All Configurations (Administration - Communications screen) Configurations (Administration - UQ screen)
Communications driver (Drivers are database records that reference files such as library files.)	Usually a library file that Communications Server uses to interface to a communications system, such as voice, or email. Includes parameters.	Parent of profile.	Communications Drivers and Profiles
Profile	Provides access to communications driver for specific contexts. Includes overrides for driver parameter.	Child of communications driver. Associated with configuration, event handler, and command. Also associated with response group, communications template, and outbound communication request.	Administrators (Administration - Communications screen): ■ Communications Drivers and Profiles End users (Communications screen): ■ My Profiles

Table 3. Communications Configuration Elements

Configuration Element	Description	How Related	Where Created / Configured
Agent	Contact center agent, manager, or other user in communications configuration. Can use multiple communications channels.	Derived from employee. Associated with configuration, teleset (for voice agents), and ACD queue (for voice agents).	All Configurations Agent General Profile All Telesets
ACD queue (agent queue)	A queue on the ACD that agents can log in to. Agents can also log in to multiple queues. Agents can log in to queues automatically, from the communications toolbar, or from the User Preferences screen.	Associated with agent (for voice agents).	Administrators: ■ List of Values (Administration - Data screen) ■ Agent General Profile (Administration - Communications screen) End users (User Preferences screen): ■ Communications
Teleset	Physical phone device such as used in a call center. Optionally, can specify host name for hoteling.	Parent of extension. Associated with agent (for voice agents).	All Telesets
Extension	Extension for teleset.	Child of teleset.	All Telesets
Event handler	Means of evaluating information received from communications driver. Includes parameters.	Child of configuration. Associated with profile and event response.	All Configurations All Event Handlers
Event response	Defines response invoked by event handler. Includes parameters.	Child of configuration. Associated with event handler and event log.	All Event Responses

Table 3. Communications Configuration Elements

Configuration Element	Description	How Related	Where Created / Configured
Event log	Specifies logging behavior for event response. Includes parameters.	Child of configuration. Associated with event response.	All Event Logs
Command	Means of performing a communications action. Commands are typically sent to communications driver. A command can be specified as a subcommand of another command. Includes parameters.	Child of configuration. Associated with profile and command data.	All Configurations All Commands
Command data	Provides data applicable to command. Includes parameters.	Child of configuration. Associated with command.	All Command Data
Response group	Defines how inbound communications such as email, or routed voice calls are processed. Invokes a workflow process that evaluates and processes inbound communications. Routing inbound email or voice calls requires Siebel Universal Queuing or another routing solution.	Associated with profile.	All Response Groups

Table 3. Communications Configuration Elements

Configuration Element	Description	How Related	Where Created / Configured
Communications template	Provides content or structure for outbound message using email, fax, or page channels. Allows Siebel field data substitution. Can contain text or contain multiple template items. Template items can specify text directly or reference files or literature items.	Associated with profile and outbound communication request. Also specified for Send Email, Send Fax, and Send Page commands, and for Siebel Email Response replies.	Administrators (Administration - Communications screen): ■ All Templates End users (Communications screen): ■ My Templates
Outbound communication request (This is an operational element rather than a configuration element.)	A request to send content from communications templates (optionally with field substitution) to recipients associated with business object instances —such as to contacts associated with an account. Requests are sent using email, fax, or page. Requests can be created and submitted directly or programmatically using workflow processes.	Associated with communications template.	Administrators (Administration - Communications screen): ■ All Outbound Requests ■ Outbound Request Overview End users (Communications screen): ■ My Outbound Requests ■ My Outbound Request Overview

NOTE: If Siebel Universal Queuing is deployed, employees and employee skills must be specified in configurations defined in the Administration - UQ screen in order for agents to receive inbound work items for channels routed by Siebel Universal Queuing. For more information, see *Siebel Universal Queuing Administration Guide*.

Communications Configuration Requirements for Siebel Modules

Much of the communications configuration data that this chapter is concerned with involves sets of data organized under a type of record called a *communications configuration*. Used as a noun, this term (sometimes shortened to *configuration*) refers to these records and associated elements such as configuration parameters, agents, telesets, profiles, commands, and events. Siebel Systems provides predefined communications configurations that support multichannel communications environments.

For information about additional kinds of configuration data related to communications, see "Types of Communications Configuration Data" on page 21.

Communications drivers and *profiles*, through which Siebel Communications Server can interface to external systems such as CTI middleware, email servers, or paging are part of the communications configuration data.

Communications drivers are usually based on library files, such as DLL (Dynamic Link Library) files on Microsoft Windows. Corresponding configuration data in the Siebel Database, including driver parameters, is defined to work with each driver. Communications profiles customize driver behavior for particular purposes by allowing driver parameters to be overridden. Communications configurations have associated profiles.

Every deployment of Siebel Communications Server uses communications drivers and profiles. However, as shown later in this section, some Siebel modules use communications drivers and profiles but do *not* use (or might not use) communications configurations. For more information about:

- Communications drivers and profiles, see "Configuring Communications Drivers and Profiles" on page 44
- Communications configurations, see "Creating or Modifying a Communications Configuration" on page 50

Modules That Use Communications Configurations

If you are using any of the Siebel following modules, you must create or use a communications configuration, such as a predefined multichannel configuration:

- Siebel CTI
- Siebel CTI Connect

For more information about these modules, see "Communications Server and Siebel CTI" on page 18.

Modules That May Use Communications Configurations

Siebel Email Response customers can also use communications configurations. Note that:

- If inbound email messages are routed using Siebel Universal Queuing or another integrated package for queuing and routing, you *must* use configurations.

  NOTE: If you choose to use Siebel Email Response with Siebel Universal Queuing, your communications configuration *must* include profiles for the communications driver named "User-interactive Email." Events and commands for handling email work items are supported by this driver.

- If inbound email messages are routed manually or using Siebel Assignment Manager, you *cannot* use communications configurations.

Using communications configurations enables your agents to use features such as the communications toolbar and communications commands in the application-level menus. It also enables you to use Siebel Universal Queuing or another integrated queuing package to route inbound work items and to support multichannel communications capabilities.

Features That Do Not Use Communications Configurations

The following outbound communications features generally require communications drivers and profiles, but *do not* employ communications configurations:

- Email or other messages sent automatically using workflow policies and workflow processes
- Email or other messages sent using outbound communication requests (multiple-recipient, high-volume)
- Send Email command
- Send Fax command
- Send Page command

Send Email, Send Fax, and Send Page Commands

By default, Send Email, Send Fax, and Send Page commands are available in the File application-level menu for Siebel Business Applications users.

Apart from Send Page, these commands use communications drivers and profiles, and are available when the communications infrastructure is in place. (Send Email does not use communications drivers and profiles when it is integrated with Microsoft Outlook or Lotus Notes.)

A communications configuration is not needed in order for employees to access these menu commands. However, these Send commands are also available through the communications toolbar for agents who are enabled to use interactive communications session functionality offered by Siebel CTI, or Siebel Email Response (with Siebel Universal Queuing).

For the Send commands to be available through the communications toolbar, employees must be specified as agents within a communications configuration, and communications must be enabled for the Siebel applications. For more information, see "Enabling Session Communications and Simulation" on page 240.

For more information about user interface elements, see the following:

- Chapter 6, "Configuring User Interface Elements"
- *Configuring Siebel Business Applications*

- *Using Siebel Tools*
- *Siebel Developer's Reference*

Process of Configuring Communications Server

To set up Siebel Communications Server, review the process described in this section. For some of the steps in the configuration process described here, detailed instructions are provided in subsequent chapters. In general, you perform the following tasks:

1. "Determining Your Communications Deployment Needs" on page 28.
2. "Setting Up Your External Communications System" on page 29.
3. "Setting Up Your Siebel Server Software" on page 30.
4. "Defining Communications Server Configuration Data" on page 31.
5. "Putting Siebel Communications Server into Production" on page 33.

For more information about Siebel Communications Server configuration data, see "About Communications Configuration Data" on page 20.

See also Chapter 3, "Configuring Communications Drivers and Profiles," and Chapter 4, "Configuring Session Communications."

Determining Your Communications Deployment Needs

This task is a step in "Process of Configuring Communications Server" on page 28. The multichannel support offered by Siebel Communications Server and its architecture requires that you plan your communications deployments carefully and thoroughly. Read all the relevant sections of this book before you start the configuration process.

NOTE: If you use an external communications system (such as CTI middleware or email server) that is not supported by an existing communications driver, you may need to create your own driver. For details, see Appendix A, "Developing a Communications Driver."

Consider the following issues as you plan your implementation of Siebel Communications Server in support of Siebel Business Applications:

- Determine the Siebel applications and other products you will be using in your implementation. Determine the communications channels and external communications systems you will be supporting for your Siebel applications. Channels may include voice, email, fax, and so on.

  Applications or modules may include Siebel Call Center, Siebel Service, Siebel Sales, Siebel Email Response, and so on. Related modules include Siebel CTI, Siebel CTI Connect, Siebel Universal Queuing, and so on. (Not all modules are available for all applications or parent modules.)

- Determine which Siebel application features, technologies, or advanced configurations you will implement, such as global or distributed deployments, application integration, work-item routing, hoteling, multitenancy, Send Email, Send Fax, or Send Page commands, workflow-based communications, the customer dashboard, and so on.
- Determine whether the third-party communications products you expect to use are supported by Siebel Systems. For information about supported products, see *System Requirements and Supported Platforms* on Siebel SupportWeb.
- For communications activities that make use of communications configurations, determine the number of agents for each supported channel. For Siebel CTI, Siebel Email Response, and other modules, you define agents within communications configurations. If you are using Siebel Universal Queuing, you specify employee skills and other settings to help determine how communications work items are routed.
- Determine data and application visibility issues for your users, because they relate to communications activities.
- Determine available resources in your communications systems. For Siebel CTI, determine the available telesets and extensions, hoteling workstations, ACD queues, and so on. For Siebel Email Response, determine the email accounts (mailboxes) that will be receiving inbound communications.
- Determine the work model for inbound or outbound communications in your call center or contact center: that is, the logic of each customer interaction.
- Determine your requirements for performance, scalability, and architecture, and determine your deployment plans.

  Determine the requirements in the preceding areas, because they relate to Siebel Communications Server, other Siebel modules or server components, and the Siebel client types you will use. Also consider architectural issues for external communications systems such as CTI middleware and email servers. Information you gather in this process will help determine how you perform later configuration steps.

  For more information about planning your system architecture, see *Deployment Planning Guide* and *Performance Tuning Guide*.
- Before you begin the process described in "Defining Communications Server Configuration Data" on page 31, review how communications configurations function, and assess your requirements in this area.

Setting Up Your External Communications System

This task is a step in "Process of Configuring Communications Server" on page 28. Perform the following steps before you can configure Siebel Communications Server:

1. For Siebel CTI, set up your PBX or ACD (Automatic Call Distribution) switch, and install and configure the CTI middleware. See your vendor documentation.
2. For email support, such as for Siebel Email Response or other features, install and configure the email servers and set up mail accounts as appropriate. See your vendor documentation. For issues specific to Siebel Email Response, see *Siebel Email Response Administration Guide*.

Once you have performed these preparatory tasks, you are ready to perform Siebel Server setup tasks.

Setting Up Your Siebel Server Software

This task is a step in "Process of Configuring Communications Server" on page 28. Deploying Siebel Communications Server requires that you enable and run the following Siebel Server components:

- Application Object Manager for your application, which supports user sessions in the Siebel Web Client. For example, you may be running instances of the Application Object Manager for Siebel Call Center.
- Server Request Broker and Server Request Processor, which coordinate communications activities between instances of the Application Object Manager and instances of the Siebel Communications Server components. Server Request Broker and Server Request Processor are automatically enabled for each Siebel Server.
- Siebel Communications Server components:
  - Communications Session Manager (CommSessionMgr)
  - Communications Configuration Manager (CommConfigMgr)
  - Communications Inbound Receiver (CommInboundRcvr)
  - Communications Inbound Processor (CommInboundProcessor)
  - Communications Outbound Manager (CommOutboundMgr)

Some minimal configuration of the preceding components may also be necessary. For more information, see Chapter 8, "Administering Siebel Communications Server." Also see *Siebel System Administration Guide*.

For more information about the architecture of Siebel Communications Server, see "About Communications Server Architecture" on page 20.

To set up your Siebel Server software

1. Install or upgrade your Siebel Server software, as described in:
   - *Siebel Installation Guide* for the operating system you are using
   - *Upgrade Guide* for the operating system you are using

   Make sure that you install and enable the Communications Management component group.
2. Enable session communications on the Application Object Manager (AOM).

   Do this on each AOM that is to support users for whom the communications toolbar and other interactive session communications functionality must be available. For details, see "Enabling Session Communications and Simulation" on page 240.
3. Configure each Siebel Communications Server server component (in the Communications Management group) that you require for your implementation.

   For more information, see Chapter 8, "Administering Siebel Communications Server."

Defining Communications Server Configuration Data

The task described in this section is a step in "Process of Configuring Communications Server" on page 28. This section describes in general terms how to define or modify Siebel Communications Server configuration data. Do this when all modules have been installed and the server components have been configured. You can perform many of the steps in a different sequence than that shown here.

For more information about communications configurations, see "About Communications Configuration Data" on page 20. For detailed procedures for working with communications configuration data, see applicable sections in Chapter 4, "Configuring Session Communications."

NOTE: When you have finished specifying the communications configuration data, you must exit and restart the Siebel application for the settings to take effect. After Siebel Communications Server has been deployed, any end users who are currently connected must also exit and restart the application for the new or modified configuration data to take effect.

To define Siebel Communications Server configuration data

1. Start the Siebel application, such as Siebel Call Center, logging in as the system administrator (for example, as the user SADMIN).
2. Navigate to the Administration - Communications screen.

   For information about the views in this screen, where you perform most of the tasks identified here, see Appendix C, "Views for Communications Administration."
3. Configure communications drivers and profiles. As applicable, associate profiles with communications configurations you have already created:
   a. For each communications driver you are using, specify default parameter values.
   b. For each communications driver you are using, create profiles. As appropriate, specify parameter values to override the driver parameters.
   c. If you are not using communications configurations, skip to Step 12 on page 33.

   For more information about configuring drivers and profiles, see Chapter 3, "Configuring Communications Drivers and Profiles."

   For more information about modules that use communications configurations, see "Communications Configuration Requirements for Siebel Modules" on page 26.
4. To support modules such as Siebel CTI and Siebel Email Response (optional), create a communications configuration. See "Creating or Modifying a Communications Configuration" on page 50.

   You can modify an existing communications configuration, or create a new configuration and transfer data into it from an existing configuration.
5. For the current configuration, specify configuration parameter values. See "Specifying Parameters for Communications Configurations" on page 52.
6. For the current configuration, associate profiles you created for your CTI driver and User-interactive Email driver. See "Creating or Modifying a Communications Configuration" on page 50.

7 For the current communications configuration, specify the agents for whom the communications configuration apply:

   a Add each agent to one or more configurations. If you add an agent to more than one configuration, specify which configuration is primary for the agent.

   b Configure the agents, such as (for voice agents) to specify which ACD queues they will use, define ACD agent login and password, and so on.

      NOTE: If you are not defining ACD queues, or if the driver parameter Service:IsQueueRequired is set to FALSE for the CTI driver or for applicable driver profiles associated with the configuration, then the above step is not necessary. For more information, see "Siebel CTI Connect Driver Parameters" on page 339.

   c If telesets have already been defined (as in Step 8), you can specify the telesets an agent can use.

      For more information on specifying agents, see "Specifying Agents" on page 59.

      You do not need to associate agents with telesets that will be used for hoteling, or associate agents with telesets for agents who will not use the voice channel.

8 If you are using Siebel CTI (voice channel), specify the telesets for your call center:

   a If a teleset will be used for hoteling, specify the host name for the hoteling computer.

   b For each teleset, specify the extensions for the teleset.

   c For each teleset, you can specify agents that can use the teleset.

      See "Specifying Telesets" on page 62.

      In order to associate an agent with a teleset, the agent must have already been added to the configuration (as in Step 7). You do not need to associate agents with telesets that will be used for hoteling.

9 Specify or verify event handlers and associated event responses and event logs, and associate them with the communications configuration:

   a Create the event logs that you specify in your event responses, and associate them with the configuration.

   b Create the event responses you need, optionally specifying one or more event logs for each event response, and associate the event responses with the configuration.

   c Create the event handlers you need, specifying an event response for each event handler, and associate the event handlers with the configuration.

      See "Defining Communications Events" on page 65.

10 Specify or verify commands and associated command data definitions, and associate them with the communications configuration:

   a Create the command data definitions that you specify in your commands, and associate them with the configuration.

   b Create the commands that you need, specifying a command data definition for each command, and associate the commands with the configuration.

      See "Defining Communications Commands" on page 70.

11 Optionally, use the Communications Simulator to test your configuration. See "Enabling Session Communications and Simulation" on page 240 and "Simulating a Communications Environment" on page 211.

12 Complete any other needed configuration, customization, or performance-tuning tasks for your Siebel implementation, and test the communications configurations you are using. For example, you may:

- Implement Siebel Universal Queuing to help route inbound work items of one or more communications channel. For more information, see *Siebel Universal Queuing Administration Guide*.
- Configure response groups for inbound communications. For more information, see *Siebel Email Response Administration Guide*.
- Create templates for outbound communications. For more information, see Chapter 9, "Configuring Communications Templates."
- Modify the *Online Help* for your users.

After you have verified the configuration, you can put Siebel Communications Server into production.

Putting Siebel Communications Server into Production

The task described in this section is a step in "Process of Configuring Communications Server" on page 28. This section describes how to transfer your communications settings from the test environment to the production environment.

To put Siebel Communications Server into production

1 Follow instructions for transferring your Siebel application from the test environment to the production environment. For details, see the *Siebel Installation Guide* for the operating system you are using

2 Configure all production instances of Siebel Server components for Communications Server, as noted in "Setting Up Your Siebel Server Software" on page 30.

3 Verify your communications configuration data in the production environment.

For example, you may need to modify any communications configuration parameter values appropriately for a production environment.

4 Provide your end users with instructions for the following activities, as appropriate for your implementation and for the particular types of users:

- Starting the Siebel client. For example, this can include providing the users with the URL for a Siebel Call Center Application Object Manager that is configured to provide access to communications functionality.
- Accessing the *Online Help*.
- Setting communications preferences in the User Preferences screen.

- Using user interface controls such as the communications toolbar, Communications submenu commands, or the Communications screen to perform communications-related tasks.
- Creating outbound communications templates for personal or public use.

  Most templates are created by advanced users, especially those for public use.
- Creating communications profiles for personal use with the Send Email or Send Fax command.

  Most profiles are created by administrators. By default, end user responsibilities do not include the My Profiles view.
- Creating and submitting outbound communication requests.

For information on these tasks, see:

- Chapter 11, "Communications Operations for End Users"
- Chapter 9, "Configuring Communications Templates"
- Chapter 10, "Defining Outbound Communication Requests"

Also see documentation for related Siebel products such as Siebel Universal Queuing, Siebel Email Response, and so on.

3 Configuring Communications Drivers and Profiles

This chapter provides an overview of communications drivers and profiles used with Siebel Communications Server, and describes how to configure drivers and profiles to interface with communications software such as CTI middleware or email servers. It includes the following topics:

About Communications Drivers and Profiles

This section provides background information about communications drivers and profiles and the roles they play in supporting Siebel Communications Server functionality.

For more information:

- If you are specifying communications configurations (such as for Siebel CTI or other session communications deployments), see Chapter 4, "Configuring Session Communications," and related chapters.
- If you are upgrading from a previous version, see also Appendix D, "Upgrading from Release 6.x." See also the *Upgrade Guide* for the operating system you are using.
- For all implementations, see also related Siebel documentation for your products, such as *Siebel Email Response Administration Guide*, or *Siebel Universal Queuing Administration Guide*.
- If you require integrating with communications systems that are not supported by drivers provided by Siebel Systems, see "Communications Drivers and Third-Party Systems" on page 36 and Appendix A, "Developing a Communications Driver."

Communications Drivers Provided by Siebel Systems

The communications drivers provided by Siebel Systems are as follows:

- **Siebel CTI Connect.** Supports interactive voice communications using CTI (Computer Telephony Integration) for deployments using Siebel CTI Connect (using CTI middleware from Intel NetMerge—formerly Dialogic).

  For more information, see "Communications Drivers and Third-Party Systems" on page 36, "Communications Server and Siebel CTI" on page 18, and Chapter 12, "Using Siebel CTI Connect."

- **Internet SMTP/POP3 Server.** Supports Internet mail servers that support the SMTP protocol for outbound email or fax or the POP3 protocol for inbound email (for Siebel Email Response). This driver supports both plain-text and HTML email.

  For more information, see "Communications Server and Siebel Email Response" on page 18, "Siebel-Validated Email Servers" on page 37, and "Interfacing with Email and Fax Servers" on page 355.

- **Push Keep Alive.** Provides a heartbeat function to help maintain push communications for session-communications drivers loaded by Communications Session Manager.

  For more information, see "Using Push Keep Alive Driver for Session Connections" on page 209 and "Other Communications Drivers" on page 395.

- **User-interactive Email.** Supports email interactivity within the Siebel application, enabling agents using Siebel Email Response, when deployed with Siebel Universal Queuing, to use the communications toolbar in working with email work items.

  For more information, see "Communications Server and Siebel Email Response" on page 18, "Communications Server and Siebel Universal Queuing" on page 19, and "Supporting Email Interactivity" on page 377.

- **Modem-based TAP Paging.** For Siebel Paging, supports paging over a modem using the Telocator Alphanumeric Paging (TAP) protocol, for outbound communication requests. (The Send Page command does not use this driver.)

  For more information, see "Other Communications Drivers" on page 395.

  NOTE: Siebel Paging is included with all base applications of the Siebel Business Applications.

- **FTP.** Supports File Transfer Protocol (FTP). This driver is used by Siebel Marketing to send contact lists to vendors who are contracted to execute campaigns. It can also be used to send outbound communication requests from any Siebel application.

  For more information, see "Other Communications Drivers" on page 395.

Communications Drivers and Third-Party Systems

By means of communications drivers, Siebel applications can work with a variety of third-party CTI middleware and email servers. The communications infrastructure from Siebel Systems can work with several types of communications systems, in these scenarios:

- Siebel Systems provides several communications drivers to support Siebel communications modules or third-party communications systems. Some Siebel modules, such as Siebel CTI Connect incorporate third-party technologies.
- Third-party vendors write communications drivers, using the Adaptive Communications API, to support additional communications systems. (Alternatively, vendors can write a driver to extend a Siebel-provided communications driver.)

For updated information about third-party products supported by Siebel Systems, see:

- *System Requirements and Supported Platforms* on Siebel SupportWeb
- *Release Notes* on Siebel SupportWeb

For information about third-party communications solutions validated by Siebel Alliances, see the Alliances section of the Web site for Siebel Systems.

NOTE: For detailed requirements and procedures for installing and configuring third-party products, refer to your third-party vendor documentation.

The capabilities of your communications system (such as CTI middleware and switch combination) partly determine which functionality you can implement. For example, some switches do not support automatic call forwarding. In addition, some features supported in the communications system may not be supported in the corresponding Siebel communications driver. Consequently, information in this guide about certain features may not apply to your implementation. Consult your vendor documentation for information about supported functionality and features.

About Error Messages Related to Communications

For agents using session-based communications such as Siebel CTI, the Siebel client may sometimes receive error messages relating to communications functionality.

Some messages may originate from CTI middleware or from the communications driver. Messages are presented without modification, whether they originate from external communications systems or from Siebel modules. Such messages are prefaced with Communication: and appear at the top of the application area in the browser, to the right of the application-level menus. For more information, see "Menu Commands for Displaying Error Messages" on page 322.

The same error messages may also be logged, if logging is set for the Communications Session Manager component. For more information, see "Parameters for Communications Session Manager" on page 250.

Siebel-Validated CTI Middleware/Switches

Siebel Systems provides a communications driver and provides communications configurations that are validated to work with Intel NetMerge CTI middleware.

Siebel CTI Connect, available from Siebel Systems, Inc., includes Intel NetMerge Call Processing Software and Intel NetMerge Call Information Manager. Siebel CTI Connect also includes the Siebel communications driver for Intel NetMerge and sample Siebel communications configuration data.

For information about installing and configuring Siebel CTI Connect and about configuring Siebel Communications Server to work with this module, including applicable driver parameters, events and commands, see Chapter 12, "Using Siebel CTI Connect."

For more information about currently supported third-party CTI middleware and telephony switches, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

Siebel-Validated Email Servers

Siebel Systems provides the Internet SMTP/POP3 Server communications driver, which has been validated to work with several third-party email servers.

The Internet SMTP/POP3 Server communications driver is compliant with the following RFC standards:

- 1894: delivery status notification
- 1939: POP3
- 2045 - 2049: MIME
- 2821: SMTP
- 2822: text message syntax
- 3207: SMTP session encryption using SSL

NOTE: Configure your email servers appropriately to work with Siebel applications. For example, for security reasons, your SMTP server may restrict from which originating machines outbound email messages can be relayed. In this case, you must configure the SMTP server to allow messages to be relayed from the Siebel Server machines running the Outbound Communications Manager server component.

For more information about currently supported third-party email servers and related products, see *System Requirements and Supported Platforms* on Siebel SupportWeb. Also see your third-party vendor documentation.

For information about using the Internet SMTP/POP3 Server driver to integrate Siebel Communications Server with email servers, see "Interfacing with Email and Fax Servers" on page 355.

Nonsupported Communications Systems

If your existing communications system is not supported by the communications drivers provided with Siebel Communications Server, consider the following options for custom communications drivers:

- Use a communications driver developed, using the Siebel Adaptive Communications API, by a third party—such as a CTI middleware vendor. For information about third-party communications solutions validated by Siebel Alliances, see the Alliances section of the Web site for Siebel Systems.
- Develop your own communications driver, using the Siebel Adaptive Communications API, to work with your communications system.

NOTE: Communications drivers developed by customers or third parties are not supported by Siebel Systems, Inc., but by their respective developers or providers.

A Siebel Communications Server implementation using a custom communications driver developed using the Siebel Adaptive Communications API can take full advantage of standard features such as the communications toolbar, screen pops, communications commands, automatic creation of activities and logs, and user interface for administration and configuration.

A communications driver can be created that is to be used with an existing driver. Such a driver, known as an aggregate driver, extends the capabilities of the existing driver, enabling additional functionality. For more information, see Appendix A, "Developing a Communications Driver."

Communications Driver Settings

In the Communications Drivers and Profiles view, the Communications Drivers list displays settings for each driver. These settings are discussed in general terms later in this section.

In most cases, these settings do not need to be changed for the communications drivers provided by Siebel Systems. The settings are informative, however, and should be understood by the communications administrator. If you add a custom driver, you must specify values for these settings.

Driver settings include:

- **Channel Type.** For more information, see "Communications Drivers and Channels" on page 41.
- **Inbound.** For more information, see "Inbound and Outbound Drivers" on page 41.
- **Outbound.** For more information, see "Inbound and Outbound Drivers" on page 41.
- **Interactive.** For more information, see "Interactive Drivers" on page 41.
- **Channel String.** For more information, see "Communications Drivers and Channels" on page 41.
- **Library Name.** For more information, see "Communications Driver Files and Database Records" on page 39.
- **Icon File.** For more information, see "Icon Files for Interactive Communications Drivers" on page 42.

Communications Driver Files and Database Records

Communications drivers are usually based on library files, such as DLL (Dynamic Link Library) files on Microsoft Windows, or shared object files on UNIX. Alternatively, they may be executable files or other types of files. Each driver file must be written to support the Siebel Adaptive Communications API.

Communications driver files are located by default in the bin subdirectory of the Siebel Server. You can store them in another directory, if you specify an absolute path.

Each communications driver file has one or more corresponding database records in the Communications Drivers and Profiles view of the Administration - Communications screen.

NOTE: In this book, in discussions of configuration tasks in the Administration - Communications screen, the term *communications driver* or *driver* usually refers to a database record that references a driver file. In discussions of developing drivers, however, the term generally refers to the driver file itself, which is the direct product of the development process.

Driver File Naming on Microsoft Windows and UNIX Platforms

The names of actual driver files to be loaded are obtained from the Library Name field in the Communications Drivers list. In some cases, the value for this field may be manipulated to obtain the name of the driver file to load.

Driver File Naming on Microsoft Windows Platforms

On Microsoft Windows platforms, the name of the driver file to load is generally obtained by appending .DLL to the value of the Library Name field. However, if the value of the Library Name field includes a period (.), then no manipulation of this value is performed to obtain the name of the driver file.

If you add a new custom driver file that has the extension .DLL, you can provide a Library Name value in either of two ways:

- The value can correspond exactly to the name of the operating system file. For example, for a file named driver.dll, you can enter driver.dll as the Library Name value. This approach is recommended.
- The value can correspond to the name of the file but without the extension .DLL. For example, for a file named driver.dll, you can enter *driver* as the Library Name value.
- LIBdvalue of the LibraryNam.so

Driver File Naming on UNIX Platforms

On UNIX, the name of the driver file to load is generally obtained by adding LIB (without a character space) immediately before the value of the Library Name field and, appending .SO to the value of the Library Name field. However, if the value of the Library Name field includes a period (.), then no manipulation of this value is performed to obtain the name of the driver file.

If you add a new custom driver file that starts with LIB and has the extension .SO, you can provide a Library Name value in either of two ways:

- The value can correspond exactly to the name of the operating system file. For example, for a file named libdriver.so, you can enter libdriver.so as the Library Name value. This approach is recommended.
- The value can correspond to the name of the file but without the elements LIB and .SO. For example, for a file named libdriver.so, you can enter *driver* as the Library Name value.

Additional Considerations for Driver File Naming

Following are additional considerations for specifying driver file names.

- If you add a new custom driver file with any other naming pattern, then the file name *must* include a dot, and you *must* provide a Library Name value that corresponds exactly to the name of the operating system file.
- If you install any custom driver file in a nondefault location, then some element of the file or path name *must* include a dot, and you *must* provide a Library Name value that includes the full path where the file can be located by the Siebel Server that is to load the driver.
- If you create a custom driver that aggregates an existing driver provided by Siebel Systems, reference the operating system file for the Siebel driver by its exact name, regardless of what text is displayed for the Siebel driver in the Library Name field.
- If the file path is not specified, it is assumed that the driver is located in the bin subdirectory of the Siebel Server installation directory. The installation from which the driver file is loaded is the machine on which the channel manager runs.

Communications Drivers and Channels

Each communications driver file can be written to support a single communications *channel type*, such as voice, email, and so on. Alternatively, a driver file can be written to support multiple channel types. Most of the communications drivers provided by Siebel Systems each support a single channel type. For example, the driver Internet SMTP/POP3 Server supports the email channel.

Each communications driver record includes a string, the *channel string*, that is used by the driver to identify its own channel type. The channel string must match exactly the channel string as defined in the driver file.

For communications driver files that support multiple communications channel types, driver records corresponding to specific channels can reference different subsets of channel-specific functionality within the same driver file, by referencing different channel strings defined within the driver file.

For more information, see Appendix A, "Developing a Communications Driver."

Inbound and Outbound Drivers

Each driver record in the Communications Drivers and Profiles view is indicated to support, or not support, inbound or outbound communications:

- CTI drivers, such as that for Siebel CTI Connect (using Intel NetMerge), are for both inbound and outbound communications. They also support agent interactivity within the Siebel application.
- The user-interactive driver for email supports agent interactivity within the Siebel application. It does not directly support either inbound or outbound communications.
- The Push Keep Alive driver is an interactive driver that provides a service for agent communications sessions. It does not directly support either inbound or outbound communications.
- Communications drivers for email or fax, such as Internet SMTP/POP3 Server, support both inbound and outbound communications. Inbound functionality for such a driver is applicable only to Siebel Email Response.
- Drivers for paging and FTP support outbound communications only.

For more information about interactive drivers, see "Interactive Drivers" on page 41.

NOTE: The Inbound and Outbound fields are for informational purposes. Do not modify the values for these fields for the drivers provided by Siebel Systems.

Interactive Drivers

Depending on the purpose of the driver, the driver record in the Communications Drivers and Profiles view may be indicated as interactive. In particular, the Siebel CTI Connect driver, the user-interactive driver for email, and the Push Keep Alive driver are interactive drivers.

Interactive drivers enable communications functionality to be available to end users using the communications toolbar and related menu controls. Additional capabilities may also be required, such as a Siebel Universal Queuing integration to route communications work items to an agent's communications toolbar.

The Push Keep Alive driver, unlike other interactive drivers, does not have a user interface component.

Communications configurations can reference profiles for interactive drivers only. The events and commands defined in the configuration interact with the communications system through one of the interactive drivers.

NOTE: The Interactive field is for informational purposes. Do not modify the values for this field for the drivers provided by Siebel Systems. Custom drivers developed by third parties must be written to be interactive or not interactive, according to their purpose.

Icon Files for Interactive Communications Drivers

Generally, each interactive driver has a specified icon file, which designates the image file that represents the channel for interactive purposes in the communications toolbar. The icon file is used for the Channel Type Indicator and for blinking behavior, such as when an incoming work item has arrived.

For the following interactive communications drivers (driver record), provided by Siebel Systems, a default icon file is specified. For each interactive driver created by third parties, an icon file *must* be specified. Other communications drivers do not use icon files. (The Push Keep Alive driver, although it is an interactive driver, does not use an icon file.)

- The Siebel CTI Connect driver, for the voice channel, uses voice.gif.
- The User-interactive Email driver, for the email channel, uses mail.gif.

The directory `webmaster\images\language_code` in the Siebel Server installation directory stores the source image files, where *language_code* represents the language code for the installed software, such as ENU for U.S. English.

To use your own icon file, place your icon file into the directory identified above. You can either replace an existing file provided by Siebel Systems with your own, or use a new file and specify this filename in the Icon File field for the communications driver.

NOTE: New or updated files on the Siebel Server are automatically populated to the Web server whenever the Web server (where the Siebel Web Server Extension is installed) is restarted. Files can also be manually updated from the Siebel Server without restarting the Web server. For more information, see *Security Guide for Siebel Business Applications* and *Siebel Installation Guide* for the operating system you are using.

If you are using a custom communications driver (interactive driver), you can use your own icon file or reuse an icon file provided by Siebel Systems, such as one that represents the same communications channel.

To work within the existing toolbar framework, each image file used for a driver icon file must be 18 x 18 pixels, and must be of format GIF or JPG.

The driver icon files are not the same physical files as the image files that are specified as bitmaps for the toolbar commands in Siebel Tools, although some of the actual images may be identical or closely related. If you modify icons used in one context (interactive icon file or bitmap specified in Siebel Tools), modify them in both contexts to maintain user interface consistency.

For more information, see "About Communications Toolbar Configuration" on page 151.

About Profiles for Communications Drivers

A communications driver is initially configured by setting default parameter values for the driver within the Communications Drivers and Profiles view.

A communications driver can be made available to serve a particular purpose by creating one or more profiles that reference this driver. For each profile, parameter values can be specified to override those of the referenced driver or to provide values not otherwise specified.

Communications administrators, who are the primary audience for this book, generally configure drivers and create profiles. Depending on the features or products your company is deploying, multiple administrators may coordinate to perform these tasks.

For example, administrators who have expertise in CTI middleware or email servers may be tasked to configure the communications drivers and profiles that are specific to the technology or product areas with which they are familiar. Or some administrators may configure profiles only, under the direction of another administrator who configures drivers.

For profiles that may be specified by end users in some of the contexts described below, administrators can specify in the Communications Drivers and Profiles view in the Administration - Communications screen which Siebel responsibilities will have visibility to the profiles.

Profile visibility applies to the Send Email window (with the Siebel native email client) and Send Fax window, to email reply features using Siebel Email Response, and to outbound communication requests. Users can see only those profiles associated with their responsibilities, or the profiles they have created themselves.

If your company implements multiple organizations, administrators can also associate each profile with an organization. For example, this may be used by Siebel Email Response in looking up the contact associated with an inbound email message.

Contexts of Use for Communications Profiles

Communications profiles, whether created by administrators or (in limited contexts) by end users, may be specified for use by administrators and end users in many communications-related contexts within the Siebel application.

Administrator Contexts for Using Profiles

Administrators specify profiles in the following contexts:

- When defining communications configurations to support handling interactive work items such as voice calls for Siebel CTI or email messages for Siebel Email Response with Siebel Universal Queuing.
- When defining response groups for inbound communications, such as to support Siebel Email Response.

End Users Contexts for Using Profiles

General end users specify profiles in the following contexts:

- When specifying a default email/fax profile in the User Preferences screen (Outbound Communications options). The profile specifies the message sender.
- To specify the sender when sending outbound communications using the Send Email command (for the native Siebel email client) or Send Fax commands.

Specialized end users specify profiles in the following contexts:

- When defining communications templates for use with outbound communication requests.
- When defining contact lists for vendor campaign execution for Siebel Marketing.

If the capability is provided to them, end users can create profiles for their own use, by using the My Profiles view in the Communications screen. User-defined profiles support particular features, such as the Send Email or Send Fax commands, for which the users can directly specify the profile to be used.

Visibility Criteria for End Users Working with Profiles

In any of the preceding contexts, an end user can see only the profiles that meet one of the following conditions:

- The responsibility associated with the profile is one of the user's responsibilities.
- The organization associated with the profile includes the position that is the user's current position.
- The profile was created by the user.

Configuring Communications Drivers and Profiles

This section describes how to work with communications drivers and profiles to enable Siebel Communications Server to work with communications systems such as CTI middleware, email servers, and so on.

Every Siebel module that uses Communications Server makes use of communications drivers and profiles.

In the Communications Drivers and Profiles view, the Communications Drivers list displays settings for each driver. Generally, the only driver data that you create or modify directly for drivers provided by Siebel Systems are the default driver parameter values and the profiles you create for the drivers.

The rest of this section describes specifying driver parameter values, creating profiles, and adding new driver records for custom driver files created using the Adaptive Communications API.

For more information about the parameters supported by each driver provided by Siebel Systems, see the sections referenced in "Communications Drivers Provided by Siebel Systems" on page 35.

Specifying Driver Parameter Values

For each driver listed in the Communications Drivers and Profiles view, driver parameter data is displayed in the Driver Parameters list. For each driver parameter, a flag indicates whether the parameter requires a value.

Driver parameter values can be provided in one of two ways:

- By setting a default value for a driver parameter. For more information, see "Setting Driver Parameter Default Values."
- By specifying an override value for the parameter in each profile you create for the driver. For more information, see "Defining Communications Profiles" on page 46.

Driver parameter values are in effect for all profiles using the driver, unless you override parameter values for a given profile.

Values for many parameters are likely to be specific to your site (such as a server name) or to an individual profile (such as a mailbox name).

NOTE: For any required parameter for which no value has been provided as a driver parameter, a value must be provided for the parameter, for each profile, in the form of a profile parameter override. If any parameter is expected to be specified as an profile parameter override, then the driver parameter *must* first be defined in the Driver Parameters list.

Field Types for Driver Parameters

Parameters for communications drivers use one of the following field types:

- Characters (letters, numerals, _ (underscore), and special characters used in macro-expansion)

  Macro expansion can be used in fields of character type, as described in the section "Using Macro Expansion for Character Fields" on page 181. You can also use macros described in the section "Macros for Parameter Values" on page 182.
- Boolean (allowable values are TRUE and FALSE)
- Numeric (numerals 0–9)

Wildcard characters are not applicable to driver parameter values.

Setting Driver Parameter Default Values

You can specify a new default value for a driver parameter. You can modify any default value in a profile. Values for certain parameters will typically be provided through a profile parameter override.

If you expect that a particular parameter value will apply to all or most of the profiles that you create; that is, you do not expect to override the value—you should probably specify this value as the default driver parameter value. Doing so will save you time when you create profiles, because you may not have to define an override for this driver parameter.

For the Internet SMTP/POP3 Server driver (which supports the email channel) if end users create profiles for personal use with the Send Email or Send Fax commands, then you must carefully consider for which parameter values these users will be expected to provide override values.

For information about creating personal profiles, see "Creating Communications Profiles for Personal Use" on page 323. For information about using the Send commands, see "Sending Email, Fax, and Page Messages" on page 324.

For information about configuring the Send commands, see "Configuring the Send Commands in the File Menu" on page 162.

To specify a driver parameter default value

1. Navigate to Administration - Communications > Communications Drivers and Profiles.
2. In the Communications Drivers list, select the driver that you want to modify the default parameter values.
3. Click the Driver Parameters view tab.
4. In the Driver Parameters list, clear the existing default value for a parameter, and enter a new value.

Defining Communications Profiles

After you configure the communications drivers you will use, as described in "Specifying Driver Parameter Values" on page 45, you create communications profiles.

Each profile specifies which driver is to be used, and how the driver will be used, for communications that use the profile. For email drivers, the profile also specifies the sender of outbound communications that use the profile.

When a Siebel Communications Server administrator or end user creates a communications template to be included in a communication request, the profile that will be used to deliver the template must be associated with the template.

If you copy or delete a communications profile, any associated parameters are also copied or deleted (cascade copy or cascade delete). A profile you copy may not display its child parameter records unless you step off, then reselect, the new profile record you created.

NOTE: For more information about creating and using profiles with Siebel Email Response, see *Siebel Email Response Administration Guide*.

To create a communications profile

1. Navigate to Administration - Communications > Communications Drivers and Profiles.
2. In the Communications Drivers list, select the driver for which you will be creating a profile.
3. Click the Profiles view tab.
4. In the Profiles list, add a new record to create a profile for the current driver.
5. Provide a name for the new profile.
6. Specify the organization that will use the profile.
7. Specify one or more responsibilities that will use the profile.

   The profile will be visible to a user in the profile's context of use—for example, in Siebel Email Response and in the Send Email and Send Fax commands—if the profile meets one of the visibility criteria described in "Contexts of Use for Communications Profiles" on page 43.

Now you specify parameter override values, as described in the following procedure.

Specifying Parameter Override Values for Profiles

This section describes specifying any new values to override the default values of driver parameters. You must do this for required parameters that are not defined for the driver, or for parameters for which the driver parameter default value is not appropriate for each profile.

For example, if you are using the Internet SMTP/POP3 Server driver within an environment supporting multiple email servers, you must create different profiles for this driver. Each profile provides a different value for the SMTP Server parameter, identifying the name of the server.

For more information about driver parameters, see "Specifying Driver Parameter Values" on page 45.

To specify parameter override values for a profile

1. Navigate to Administration - Communications > Communications Drivers and Profiles.
2. In the Communications Drivers list, select the driver for which you created the profile you are configuring.
3. Click the Profiles view tab.
4. In the Profiles list, select the profile you are configuring.
5. In the Profile Parameter Overrides list, add a new record for each driver parameter whose default value you are overriding.
6. For each parameter, specify the parameter name.
7. Enter an override value for the parameter, then commit the record.

   You must specify a parameter override for any parameter for which no value has been provided as a driver parameter.

8 For a profile that is used by an active response group, you can optionally choose Submit Profile Changes from the applet menu of the Profiles list.

   If you change profile parameters (or the underlying driver parameters) on a profile (typically a profile for the Internet SMTP/POP3 Server driver), use this command to update this profile in the active response group. The command unloads and reloads the profile and profile parameters in any active response group that includes the profile—such as in response groups for email accounts that are currently being monitored for Siebel Email Response.

NOTE: Submit Profile Changes has no effect on profiles that are not used in active response groups. Also, this command does not cause any other profiles to be reloaded for affected response groups. To reload an entire response group and all of its profiles, use the Submit Response Group Changes command in the All Response Groups view. If you have deleted a profile, use Submit Response Group Changes to unload the profile from any active response group. For more information about response groups, see *Siebel Email Response Administration Guide*.

Adding a Custom Communications Driver

If you have developed a custom communications driver, using the Adaptive Communications API, or acquired such a driver from another vendor, you need to add one or more records for the driver in the Communications Drivers and Profiles view.

For more information about drivers and about the settings mentioned below, see "Communications Driver Settings" on page 39.

For more information about developing a custom driver, see Appendix A, "Developing a Communications Driver." See also "Configuring Communications List of Values Types" on page 163.

To add a custom communications driver

1 Navigate to Administration - Communications > Communications Drivers and Profiles.

2 In the Communications Drivers list, add a new record for the driver.

   The Communications Drivers form appears.

3 Provide a name for the driver.

4 Specify the channel type for the driver.

5 Specify the channel string the driver uses to identify its own channel type.

6 Specify if the driver will be used for inbound communications.

7 Specify if the driver will be used for outbound communications.

8 Specify if the driver is interactive.

9 Enter the name of the file, such as a library file, this driver record references.

   Include the complete file name of the file, with the file extension.

10 For an interactive driver, enter the name of the driver's icon file.

11 Optionally, provide a description for the new driver.

4 Configuring Session Communications

This chapter provides information about configuring session communications, such as for Siebel CTI. It includes the following topics:

About Configuring Session Communications

This chapter explains how to set up communications configurations and related data. Communications configurations support Siebel CTI and other forms of session communications. This chapter also provides information about importing and exporting communications configuration data.

Previous and subsequent chapters address related and advanced topics pertinent to the topics in this chapter:

- For an overview of steps involved in configuring your communications environment, see "Process of Configuring Communications Server" on page 28.
- To configure communications drivers and profiles, see Chapter 3, "Configuring Communications Drivers and Profiles."
- For more information about configuring events or commands in your communications configurations, see Chapter 5, "Configuring Events and Commands."
- To perform a variety of advanced or special-purpose configuration tasks, see Chapter 7, "Configuring Advanced Communications Features."
- To configure communications elements in the user interface for Siebel Business Applications, see Chapter 6, "Configuring User Interface Elements."
- To enable interactive communications sessions, and to configure and run Siebel Server components for Siebel Communications Server, see Chapter 8, "Administering Siebel Communications Server."

For more information:

- If you are upgrading from a previous version, see also Appendix D, "Upgrading from Release 6.x." This appendix primarily addresses upgrade issues for Siebel CTI and Siebel Communications Server. See also the *Upgrade Guide* for the operating system you are using.
- For all implementations, see also related Siebel documentation for your products, such as *Siebel Email Response Administration Guide* or *Siebel Universal Queuing Administration Guide*.

Creating or Modifying a Communications Configuration

In the Administration - Communications screen, use the All Configurations view to create or modify a communications configuration.

The sample configuration shown in Table 4 is provided with Siebel Communications Server to support handling work items of communications channels such as voice and email. These channels are supported by third-party products such as CTI middleware/switches or email software.

Your installation may include other configurations beside this one, such as those that may be imported from .DEF files. For more information about .DEF files, see "Exporting and Importing Configuration Data" on page 74.

Table 4. Sample Communications Configuration

Configuration Name	Voice Channel Requirements	Email Channel Requirements
Multichannel configuration A	■ Requires Siebel CTI Connect (based on Intel NetMerge) ■ Configuration requires profile for Siebel CTI Connect communications driver ■ Optionally, routing work items uses Siebel Universal Queuing	■ Requires integration to email server using Internet SMTP/POP3 Server driver ■ Configuration requires profile for User-interactive Email communications driver ■ Routing work items requires Siebel Universal Queuing or another routing solution

After you create a communications configuration, you must create and associate parameters, profiles, agents, events, and commands, in order for the configuration to be functional:

- For information about specifying parameters, see "Specifying Parameters for Communications Configurations" on page 52.
- For information about specifying profiles, see "Configuring Communications Drivers and Profiles" on page 44.
- For information about specifying agents, see "Specifying Agents" on page 59.
- For information about specifying events, see "Defining Communications Events" on page 65.
- For information about specifying commands, see "Defining Communications Commands" on page 70.

Rather than using the Administration - Communications screen to create or modify communications configuration data directly, you can export some of the data to files and modify or extend it using a text editor. Then you can import the data into the test or production Siebel Database when you are ready to test, deploy, or update the communications configuration. For more information, see "Exporting and Importing Configuration Data" on page 74.

To create a communications configuration

1. Navigate to Administration - Communications > All Configurations.

   The All Configurations view appears.
2. In the Configurations list, add a new record.
3. In the Name field, enter the name of the configuration.
4. Add any comments.
5. As appropriate, create or associate elements such as parameters, profiles, agents, commands, and event handlers, as described later in this chapter.

To edit an existing communications configuration

1. Navigate to Administration - Communications > All Configurations.

   The All Configurations view appears.
2. In the Configurations list, select the record for the configuration to edit.
3. Make your changes.
4. As appropriate, create or associate elements such as parameters, profiles, agents, commands, and event handlers, as described later in this chapter.

Copying or Deleting a Communications Configuration

You can copy or delete an existing communications configuration record.

Copying a Communications Configuration

You can copy an existing communications configuration record by using the available menu commands.

NOTE: When you copy a record for a communications configuration, associated configuration parameters, profiles, agents, commands (commands and command data), and events (event handlers, event responses, and event logs) are *not* copied.

Deleting a Communications Configuration

You can delete an existing communications configuration by using the available menu commands.

When you delete a record for a communications configuration, associated configuration parameters, commands (commands and command data), and events (event handlers, event responses, and event logs) are also deleted, recursively. For agents and profiles, however, the associations with these elements are deleted, but the underlying employee records and profile records are *not* deleted. When you perform this operation, the application may take several minutes to respond.

Viewing All Communications Configuration Data

Use the Configuration Explorer view to view information about all the elements of your communications configuration.

The Configuration Explorer view displays communications configuration data in a hierarchical format. This view comprises an explorer (tree control) on the left and lists on the right. The explorer on the left is read-only.

To view communications configuration data

1. Navigate to Administration - Communications > Configuration Explorer.

   The Configuration Explorer view is displayed. Any item in the explorer that is preceded by a plus sign (+) may contain other configuration items. For example, configurations contain commands, which in turn contain command parameters and associated command data definitions.

2. Click the plus sign for an item in the explorer to expand the item and reveal its specific subitems or categories of subitems.

3. Click an item's name in the explorer applet to display subitem information in the relevant list on the right.

Specifying Parameters for Communications Configurations

Use the All Configurations view to specify parameter values for a communications configuration. Configuration parameters are in effect for all agents who are included in the configuration for the contact center.

Certain elements associated with a communications configuration also have parameters that affect the overall function of the configuration. Such elements include communications drivers and profiles, commands, and events.

The communications configurations provided by Siebel Systems include configuration parameters and default values. "Parameters for Communications Configurations" on page 53 lists these configuration parameters, and provides applicable default values which are in effect for each parameter if it is not defined. Some parameters must be included, while others are optional.

For more information about creating or modifying communications configurations, see "Creating or Modifying a Communications Configuration" on page 50.

To specify configuration parameters

1. Navigate to Administration - Communications > All Configurations.
2. In the Configurations list, click to select the configuration record for which you are specifying parameters.
3. In the Parameters list:
   a. If the configuration does not include all supported parameters you require, add a record for each such parameter, specifying the parameter name and parameter value.
   b. Verify or edit values for all the parameters in your configuration.
   c. Click the check mark for the Active field to enable or disable a configuration parameter. A disabled parameter has no effect on the communications configuration.

Parameters for Communications Configurations

This section describes the configuration parameters for communications configurations and displays applicable default values. The following are the parameters for communications configurations:

- **AutoLogin.** Specifies whether autologin is the global default setting for all agents in the communications configuration, or whether agents can set autologin:
  - If AutoLogin is set to TRUE, then autologin is in effect for all agents.
  - If AutoLogin is set to FALSE, then autologin is disabled for all agents. If the parameter is not defined, the default setting is FALSE.
  - If AutoLogin is set to UserPreference, then autologin is determined by the autologin setting (Auto Login to Call Center at Startup) in the agent's User Preferences screen (Communications options).

  If AutoLogin is either TRUE or FALSE, then agents cannot set the preference Auto Login to Call Center at Startup.

  For more information, see "Setting Communications User Preferences" on page 305 and "Configuring Communications Log In and Log Out" on page 201.

- **AutoLoginCmd.** Specifies which communications command for logging in is executed for each user's session, if AutoLogin is set to TRUE.

  For more information, see "Setting Communications User Preferences" on page 305 and "Configuring Communications Log In and Log Out" on page 201.

- **BackupCommSessionMgr.** Specifies the name of the backup Communications Session Manager component.

  The backup Communications Session Manager component can be accessed without agent interruption, in case the primary Communications Session Manager fails and does not restart.

  You *must* define this parameter if the backup Communications Session Manager component has a different name than "CommSessionMgr." Otherwise, it is optional. The value is the component alias (such as CommSessionMgr), not the component name (such as Communications Session Manager).

  If you are using a backup server for Communications Session Manager, then you define the parameters BackupCommSessionMgr, BackupRequestServer, BackupEnterpriseServer, and BackupGatewayAddress, as needed. These parameters fully identify the backup Communications Session Manager applicable to this communications configuration.

  The BackupRequestServer parameter is always required in order to support a backup Communications Session Manager.

  NOTE: It is recommended that you run the backup Communications Session Manager and the Application Object Manager on Siebel Servers for which the Enterprise Servers (one or more) are served by the same Siebel Gateway Name Server. In this case, the BackupEnterpriseServer and BackupGatewayAddress parameters are optional.

- **BackupEnterpriseServer.** Specifies the name of the Siebel Enterprise for the backup Communications Session Manager.

  You *must* define this parameter if the Siebel Servers supporting the backup Communications Session Manager and the Application Object Manager run within different Siebel Enterprise Servers. Otherwise, it is optional.

  For more information, see the description of the BackupCommSessionMgr parameter.

- **BackupGatewayAddress.** Specifies the name of the Siebel Gateway Name Server applicable to the backup Communications Session Manager. The value is the Name Server machine name or IP address.

  Include the port number if the Siebel Gateway Name Server uses a port other than the default (2320). For example, mygateway:*new_port_num*.

  You *must* define this parameter if the Siebel Servers supporting the backup Communications Session Manager and the Application Object Manager run within Siebel Enterprise Servers that are served by different Siebel Gateway Name Servers. Otherwise, it is optional.

  For more information, see the description of the BackupCommSessionMgr parameter.

- **BackupRequestServer.** Specifies the name of the Siebel Server on which the backup Communications Session Manager is running.

  You *must* define this parameter when running a backup Communications Session Manager. The BackupRequestServer parameter is always required in order to support a backup Communications Session Manager.

  For more information, see the description of the BackupCommSessionMgr parameter.

- **ChannelCleanupTimer.** Specifies a timeout value, in seconds, that can help the Application Object Manager to clean up orphaned communications sessions, such as in the event of browser failure.

  If a communications message (for example, a new inbound call) is not successfully pushed from the Communications Session Manager to the Application Object Manager, then the value of ChannelCleanupTimer is compared to the number of seconds since the last successful push message was delivered to the agent's browser. If the number of seconds since the last successful push message was delivered is greater than the value of ChannelCleanupTimer, then the agent's communications session is considered to be orphaned. The session is terminated and communications session resources on the Application Object Manager are freed up for other uses.

  For example, if a message cannot be pushed, ChannelCleanupTimer is set to 60 (seconds), and if the last successful push message occurred 180 seconds ago, then this communications session will be terminated.

  NOTE: It is recommended to define the ChannelCleanupTimer parameter if you are using the Push Keep Alive driver with your communications configuration. For more information, see "Using Push Keep Alive Driver for Session Connections" on page 209.

- **CheckPopupBeforeExecute.** Specifies whether a screen pop is generated only after an agent has closed a pending pop-up window.

  A pop-up window is pending after an agent has initiated an action that displays a pop-up window and before the agent has completed and closed the window. A screen pop is likely to be disruptive to an agent's workflow if it occurs during this time:

  - If CheckPopupBeforeExecute is set to TRUE, then a screen pop will not be generated for an agent if a pop-up window is pending. This parameter is TRUE by default.
  - If FALSE, a screen pop may be generated regardless of any pending pop-up.

- **CommSessionMgr.** Specifies the name of the Communications Session Manager component applicable to this communications configuration.

  Define this parameter when the Communications Session Manager is running on a different machine than the Siebel Server on which the Application Object Manager is running.

  You *must* define this parameter if your Communications Session Manager component has a different name than "CommSessionMgr." Otherwise, it is optional. The value is the component alias (such as CommSessionMgr), not the component name (such as Communications Session Manager).

  You define the CommSessionMgr, RequestServer, EnterpriseServer, and GatewayAddress parameters, as needed. These parameters fully identify the Communications Session Manager applicable to this communications configuration. The RequestServer parameter is always required in order to run Communications Session Manager on a designated machine.

  NOTE: It is recommended that you run Communications Session Manager and the Application Object Manager on Siebel Servers for which the Enterprise Servers (one or more) are served by the same Siebel Gateway Name Server. In this case, the EnterpriseServer and GatewayAddress parameters are optional.

  NOTE: If Siebel Server Load Balancing is enabled, you generally run Communications Session Manager on *all* of the load-balanced machines (where Application Object Manager is running). If Communications Session Manager must be disabled on *any* of these machines, then you must run it on another machine and specify its location using the parameters described here.

  If you are also running a backup Communications Session Manager, see also the descriptions for the BackupCommConfigMgr parameter and related parameters.

  See also "Administering Communications Session Manager" on page 254.

- **ConnectString.** Specifies the connect string to indicate the name of a remote instance of a server such as a CTI middleware server.

  For more information, see "Configuring Remote Transfers and Conferences" on page 204.

- **DialingFilter.Rule*N*.** Used by the Siebel application to manipulate telephone numbers for voice calls made, transferred, or made in a conference call.

  Specifies a set of phone-number translation rules that are invoked when the Lookup or PhoneTypeLookup keyword is specified in macro-expanded text in a communications command for the voice channel (using Siebel CTI).

  The first set of numbers is searched for. If there is a match, the searched numbers are translated to the numbers after the -> symbols. For example:

  DialingFilter.Rule001="650477->"

  This filter rule takes a ten-digit domestic phone number (for example) and translates it into four digits for dialing an internal extension.

  For more information, see "Working with Dialing Filters" on page 195 and "Using Macro Expansion for Character Fields" on page 181.

- **EnterpriseServer.** Specifies the name of the Siebel Enterprise Server for the Siebel Server on which the applicable Communications Session Manager is running.

  You *must* define this parameter if the Siebel Servers supporting Communications Session Manager and Application Object Manager components run within different Siebel Enterprise Servers. Otherwise, it is optional.

  For more information, see the description of the CommSessionMgr parameter.

- **GatewayAddress.** Specifies the name of the Siebel Gateway Name Server for the Siebel Server on which the applicable Communications Session Manager is running. The value is the Name Server machine name or IP address.

  Include the port number if the Siebel Gateway Name Server uses a port other than the default (2320). For example, mygateway:*new_port_num*.

  You *must* define this parameter if the Siebel Servers supporting Communications Session Manager and Application Object Manager components run within Siebel Enterprise Servers that are served by different Siebel Gateway Name Servers. Otherwise, it is optional.

  For more information, see the description of the CommSessionMgr parameter.

- **MaxCommToolbars.** Specifies the number of instances of the Siebel application, for each agent, for which communications session capability can be enabled at one time—that is, for which the communications toolbar can be active.

  If this parameter is not defined, the applicable value is 1; only one communications toolbar can be active at one time for each agent. It is generally advised to leave this parameter set to 1, so that communications events are received by one Siebel application instance only.

  If an agent is running multiple Siebel application instances (for which communications would otherwise be enabled), the number of active communications toolbars cannot exceed the value of this parameter. If the maximum has been reached, and another Siebel application instance is started, the communications toolbar is not displayed in the new instance.

  By using the Reset Active Session Count command, an agent can reset communications session capabilities. An agent may need to use this command, for example, following a browser hang. In such a scenario, the Communications Session Manager may internally retain an agent session that is unavailable to the agent, and does not allow any new agent session that exceeds the value of MaxCommToolbars.

  After using Reset Active Session Count, the next time the agent starts a Siebel application instance, the communications toolbar will be enabled.

  To access this command, an agent chooses Tools > Communications > Reset Active Session Count from the application-level menu, or uses the keyboard shortcut Ctrl+F8.

  The Reset Active Session Count command is configured in Siebel Tools and should not be modified. The shortcut may be modified, if you want.

  The command is available when the user is defined as an agent within a valid configuration and the Enable Communication parameter is TRUE for the agent's application.

- **MultiTenancy.** Specifies whether or not organization-visibility rules are applied:

- If MultiTenancy is set to FALSE (the default), then organization-visibility rules will not be applied. Use this setting if your Siebel implementation does not use the multitenancy (multiple organization) feature.
- If the parameter is set to TRUE, then organization-visibility rules apply. Use this setting if your Siebel implementation uses multitenancy.

  For more information, see "Supporting Multitenancy" on page 198.

- **PreferenceLoginCmd.** Specifies which communications command is executed when the agent clicks the Login button in the Agent Queues list, located in the Communications options of the User Preferences screen.

  The default value is PreferenceLoginCmd.

  For more information, see "Setting Communications User Preferences" on page 305 and "Configuring Communications Log In and Log Out" on page 201.

- **PreferenceLogoutCmd.** Specifies which communications command is executed when the agent clicks the Logout button in the Agent Queues list, located in the Communications options of the User Preferences screen.

  The default value is PreferenceLogoutCmd.

  For more information, see "Setting Communications User Preferences" on page 305 and "Configuring Communications Log In and Log Out" on page 201.

- **RequestServer.** Specifies the name of the Siebel Server on which the applicable Communications Session Manager is running.

  You *must* define this parameter when the Communications Session Manager is running on a different machine than the Siebel Server on which the Application Object Manager is running.

  For more information, see the description of the CommSessionMgr parameter.

- **RestoreScreenOnWorkResumed.** Specifies whether or not the screen state is restored when a suspended work item of any channel is resumed:
  - If this parameter is set to TRUE, then the screen state is restored when a work item of any channel is resumed. This parameter is TRUE by default.
  - If this parameter is set to FALSE, then the screen state is not restored when a work item of any channel is resumed.

- **UpdateChannelStatusTable.** Channel status data is logged for display in the All Channel Items view only if the UpdateChannelStatusTable configuration parameter is set to TRUE.

  By default, this parameter is set to TRUE.

  In a large call center, setting this parameter to TRUE may have performance implications: the Siebel Database is updated as communications activity occurs, and the database is queried each time a manager or administrator displays or refreshes the All Channel Items view. For more information, see "Viewing Communications Status Data" on page 233.

- **UQConfigurationName.** Specifies the name of the Siebel Universal Queuing configuration to use for agents specified for this communications configuration.

  The Siebel Universal Queuing configuration is defined in the Administration - UQ screen.

  For more information about configurations for Siebel Universal Queuing, see *Siebel Universal Queuing Administration Guide*.

Specifying Agents

Use the All Configurations view and the Agent General Profile view to specify and configure agents for your communications configurations.

You specify agents by adding employees to one or more communications configurations in the All Configurations view, as described in "Creating or Modifying a Communications Configuration" on page 50.

As an alternative to adding individual employees to a configuration, you can add employees by adding responsibilities to the configuration. If you do this, all employees who have the specified responsibilities are added as agents.

After you have added agents, you configure them using the Agent General Profile view. For example, you can associate telesets or specify agent login and password for ACD queues.

For more information about the role of agents within a communications configuration, see "About Communications Configuration Data" on page 20.

Relationship of Agents and Telesets for Siebel CTI

If you are using Siebel CTI, note that you must associate agents with communications configurations, specify telesets for your call center, and then associate the agents with any telesets they will use. However, you can choose whether you want to associate agents with specific hoteling telesets that they use.

NOTE: When an agent starts the Siebel application, any explicit association of the agent with a teleset is overridden if the user's computer (host name) has been associated with the teleset for hoteling purposes.

For more information about specifying telesets, see "Specifying Telesets" on page 62. For more information about hoteling, see "Configuring Telesets for Hoteling" on page 197.

Agents and ACD Queue Settings

If agents receive voice calls through ACD queues, one or more ACD queues can be associated with the agent using the Agent General Profile view.

NOTE: If the parameter Service:IsQueueRequired is FALSE for any profiles for applicable CTI drivers that are associated with this configuration, then you need not associate agents with ACD queues. For more information, see "Siebel CTI Connect Driver Parameters" on page 339.

One or more ACD queues can be designated as primary for the agent. The Log In and Log Out buttons on the communications toolbar, and the autologin function, log an agent into or out of all primary ACD queues.

In the User Preferences screen, an agent can selectively log in to or out of any of the associated queues, including those not designated as primary. For more information, see "Setting Communications User Preferences" on page 305.

For more information, see "Configuring Communications Log In and Log Out" on page 201.

For information about setting ACD queue values, see also "Configuring Communications List of Values Types" on page 163.

Specifying Agents for Configurations

This section provides instructions for adding agents to a communications configuration. Once you have added an agent to a configuration, using the All Configurations view, you can further configure the agent, including associating the agent to other configurations, using the Agent General Profile view.

To add agents to a communications configuration

1. Navigate to Administration - Communications > All Configurations.
2. In the Configurations list, select the record for the configuration to which you add the agent.

   You can add an agent to more than one configuration, though only one configuration is in effect for the agent at a time.
3. Click the Agents view tab.
4. In the Agents list, add a new record. The Add Agents dialog box appears:
   a. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   b. For each employee that you want to add as an agent, click the check box to select this employee record.
   c. Click OK to add all selected employees to the Agents list.
5. Alternatively, in the Agents list, click Add by Responsibilities or choose it from the menu to add to the configuration all agents with the specified responsibilities:
   a. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   b. For each responsibility that you want to add, click the check box to select this record.
   c. Click OK to add all selected responsibilities to the Agents list.
6. To add an agent to another configuration, return to Step 2 and repeat the steps that follow.

Configuring Agents

This section provides instructions for configuring an agent after the agent has been added to at least one communications configuration.

Using the Agent General Profile view, you can add the agent to additional configurations, associate the agent with telesets, and specify the ACD queues from which the agent may receive voice calls. You can also designate which configuration is primary for the agent, and which ACD queues are primary for the agent.

NOTE: The Agent General Profile view lists only those agents who have been associated with one or more configurations. For more information, see "Creating or Modifying a Communications Configuration" on page 50.

Only one configuration can be designated as primary for the agent, and only one configuration is in effect at one time. When an agent logs in for the first time, the primary configuration is in effect for the agent. In the Communications options of the User Preferences screen, the agent can choose a different configuration, which will be in effect the next time the agent logs in. For more information, see "Setting Communications User Preferences" on page 305.

Telesets can be associated with an agent in this procedure only if the telesets have been defined. For more information, see "Specifying Telesets" on page 62.

You can define the Agent Login and Password that are retrieved from the database each time an agent logs into the ACD queue. You must provide these values for agents who will be configured to log in to the ACD queue, when required by the ACD. These values do not need to be specified for other users. The agent login and password need to be specified only once for an agent.

NOTE: For an Aspect switch, Agent Login also represents the number that other call-center agents dial to reach the agent.

For more information about configuring login commands, see "Configuring Communications Log In and Log Out" on page 201. For more information about agent login procedures, see "Using the Communications Toolbar" on page 314.

To configure agents

1. Navigate to Administration - Communications > Agent General Profile.

   The Agent General Profile list displays all agents that have been added to one or more communications configurations.

2. For agents who will need to log in to ACD queues, enter the appropriate values into the Agent Login and Password fields, if no values have been specified previously.

3. Specify one or more existing configurations to associate this agent with:

   a. Click the Configurations view tab.

   b. In the Configurations list, add a new record. The Add Configurations dialog box appears.

   c. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.

   d. Select one or more configurations with which you want to associate the agent.

e Click OK to associate all selected configurations with the agent.

4 If the agent is associated with multiple configurations, specify the primary configuration for the agent:

a In the Configurations list, select the record of the configuration that you want to be the agent's primary configuration.

b Click the check box for the Primary field to make this the primary configuration.

5 For voice agents, you can specify one or more existing telesets to associate this agent with:

a Click the Telesets view tab.

b In the Telesets list, add a new record. The Add Telesets dialog box appears.

c Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.

d Select one or more telesets with which you want to associate the agent.

e Click OK to associate all selected telesets with the agent.

6 To associate the agent with one or more ACD queues:

a Click the ACD Queues view tab.

b In the ACD Queues list, add a new record.

c Specify an ACD queue the agent will use.

d If this ACD queue is one of the primary ACD queues for the agent, click the check box for the Primary field.

Specifying Telesets

Use the All Telesets view to specify telesets for your communications configurations, including associating agents and extensions.

Because teleset data is specific to your enterprise, you must specify this data for your call center. Some demo telesets are defined in the Sample Database, but no predefined teleset data is otherwise provided.

Teleset data is indirectly associated with communications configurations, by way of the agents who use the teleset. In this way, Siebel Communications Server can support CTI functions for your call center agents.

You define extensions for the telesets, then associate the extensions with agents (unless you are using hoteling) that have already been associated with a configuration.

For information about how end users can set preferences for telesets and extensions, see "Setting Communications User Preferences" on page 305.

Teleset Naming and Hoteling Considerations

You can provide any sort of unique name or number for a teleset. Naming telesets after cubicle or station identifiers or machine names may be a good approach to naming telesets. This approach avoids problems that may be associated with naming them after users or extensions, given that multiple users and extensions may be associated with one teleset.

If you implement hoteling, you may choose to name a hoteling teleset after the host name of the associated hoteling computer.

When you have specified a host name for a teleset in order to support hoteling, any agent logging in to the Siebel client on the hoteling computer will also use the associated hoteling teleset. Associating a host name with a teleset overrides any association of the teleset with an agent. For more information about hoteling, see "Configuring Telesets for Hoteling" on page 197.

Specifying Telesets, Agents, and Extensions

This section provides instructions for specifying telesets and specifying agents and extensions for telesets.

To specify telesets

1. Navigate to Administration - Communications > All Telesets.
2. In the Telesets list, add a new record.
3. In the Teleset field, enter the name of the teleset.
4. If the teleset and computer are to be used for hoteling, specify in the Host field the name of the computer where the teleset is located.

   The host name you enter will be stored using all uppercase letters.
5. Enter any desired comments for the teleset.
6. For a nonhoteling teleset, click the Agents view tab and add agents for the telesets, as described below.
7. Click the Extensions view tab and add extensions for the teleset, as described below.

Specifying Agents for Telesets

This section provides instructions for specifying agents for telesets. You associate an agent with a teleset in order to authorize the agent to use this teleset.

All agents you associate with telesets must already have been associated with one or more communications configuration. For more information, see "Specifying Agents" on page 59.

Agents and telesets have a many-to-many relationship.

Agents who use hoteling exclusively, or who do not use the voice channel (CTI), do not need to be associated with telesets.

To specify agents for a teleset

1. In the All Telesets view, select the record for the teleset to which you add agents who will be authorized users of the teleset.
2. Click the Agents view tab.
3. In the Agents list, add a new record. The Add Agents dialog box appears, displaying employees who have been added as agents to one or more configurations:
   a. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   b. For each agent that you want to associate with the current teleset, click the check box to select this record.
   c. Click OK to add all selected agents to the Agents list.
4. If no Agent Login and Password values exist for a given agent, you can add this data, as necessary:
   a. Click the hyperlink in the Agent field to drill down to the Agent General Profile view.
   b. In the Agent General Profile list, enter values for the Agent Login and Password fields.

   For more information, see "Specifying Agents" on page 59.

Specifying Extensions for Telesets

This section describes how to specify extensions for telesets. After you specify telesets, agents, and extensions, the agent who uses the teleset must specify a default standard extension from the teleset that he or she is associated with. The agent does not need to specify a default standard extension if the teleset is a hoteling teleset. For details, see "Setting Communications User Preferences" on page 305.

The way you specify extension data varies according to the switch you are using. Table 5 documents how extension type must be specified (S or A) for each of the switches supported in the communications configurations provided by Siebel Systems. For more information about supported switches, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

Table 5. Extension Type by Switch

Switch Name	Extension	Extension Type
Avaya (Lucent) Definity G3	Specify one extension per teleset.	S – Standard DN
Nortel Meridian	Specify two extensions per teleset: one of each type.	S – Standard DN (also called Primary DN) A – ACD DN (also called Position DN)
Siemens Hicom 300E (Generic CSTA Phase II)	Specify one extension per teleset.	S – Standard DN

To specify extensions for a teleset

1. Navigate to Administration - Communications > All Telesets.
2. In the All Telesets view, select the record for the teleset to which you add extensions.
3. Click the Extensions view tab.
4. In the Extensions list, add a new record to create a new extension record.
5. In the Extension field, enter the extension.
6. In the Extension Type field, specify the type, as required for the switch:
   - "S" stands for standard DN: for all supported switch types, specify one extension of this type for each teleset.
   - "A" stands for the ACD DN: this extension type is only used for Nortel Meridian switches.

Viewing Extension Data

Use the All Extensions view to view information about the extensions for your telesets or to locate records for particular extensions.

You do not have to enter data in this view, because it displays data you previously specified in the All Telesets view.

To view extension data

- Navigate to Administration - Communications > All Extensions.

Defining Communications Events

You can specify events for your communications configuration. Siebel Communications Server supports three event types:

- Event handlers
- Event responses
- Event logs

In general, you create event logs first, then create event responses and associate event logs with them, then create event handlers and associate event responses with them.

Event definitions are provided by Siebel Systems. Verify whether the events in your communications configuration meet your needs before you modify them or create new events.

For more information about the role of events within a communications configuration, see "About Communications Configuration Data" on page 20.

For detailed information about working with communications events, including supported parameters, see Chapter 5, "Configuring Events and Commands."

Creating Event Logs

This section describes how to create event logs, using the All Event Logs view.

The Event Responses view tab in the All Event Logs view lets you view the event responses with which the current event log is associated.

NOTE: If you are implementing a custom communications driver, see also "Configuring Communications List of Values Types" on page 163.

To create an event log

1. Navigate to Administration - Communications > All Event Logs.
2. In the Event Logs list, add a new record.
3. In the Name field, enter the name of the event log.
4. Specify the configuration to associate this event log with:
   a. In the Configuration field, click the select button. The Pick Configuration dialog box appears.
   b. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   c. Select an existing configuration with which you want to associate the event log, then click OK.
5. Enter any desired comments for the event log.

Specifying Event Log Parameters

This section describes how to add parameters for the current event log.

To add parameters for the current event log

1. Navigate to Administration - Communications > All Event Logs.
2. In the Event Logs list, select the event log for which you want to add parameters.
3. Click the Event Log Parameters view tab.
4. In the Event Log Parameters list, add a new record.
5. In the Name field, specify the name of the parameter.

   You can choose the parameter name from a list or type the name. For a parameter of type group, choose the parameter name, then type the name of the subparameter after the period. Available event log parameters are documented in "Event Logs" on page 123.
6. In the Value field, enter the value for the parameter.

Creating Event Responses

This section describes how to create event responses, using the All Event Responses view.

You can associate zero or more event logs with an event response. Some event responses do not need an event log.

The Event Handlers view tab in the All Event Responses view lets you view the event handlers with which the current event response is associated.

The Event Responses view tab in the All Event Logs view lets you view the event responses with which the current event log is associated.

NOTE: If you are implementing a custom communications driver, see also "Configuring Communications List of Values Types" on page 163.

To create an event response

1. Navigate to Administration - Communications > All Event Responses.
2. In the Event Responses list, add a new record.
3. In the Name field, enter the name of the event response.
4. Specify the configuration to associate this event response with:
   a. In the Configuration field, click the select button. The Pick Configuration dialog box appears.
   b. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   c. Select an existing configuration with which you want to associate the event response, then click OK.
5. Enter any desired comments for the event response.

Specifying Event Response Parameters

This section describes how to add parameters for the current event response.

To add parameters for the current event response

1. Navigate to Administration - Communications > All Event Responses.
2. In the Event Responses list, select the event response for which you want to add parameters.
3. Click the Event Response Parameters view tab.
4. In the Event Response Parameters list, add a new record.
5. In the Name field, specify the name of the parameter.

   You can choose the parameter name from a list or type the name. For a parameter of type Group, choose the parameter name, then type the name of the subparameter after the period. Available event response parameters are documented in "Event Responses" on page 110.

6 In the Value field, enter the value for the parameter.

Associating Event Logs with an Event Response

This section describes how to associate an event log with the current event response.

To associate an event log with the current event response

1 Navigate to Administration - Communications > All Event Responses.

2 In the Event Responses view, select the event response with which you want to associate an event log.

3 Click the Associated Event Logs view tab.

4 In the Associated Event Logs list, add a new record.

5 Specify the event log to associate with this event response:

   a In the Name field, click the select button. The Pick Event Log dialog box appears, displaying existing event logs associated with the same configuration as the event response.

   b Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.

   c Select an event log with which you want to associate the event response, then click OK.

6 In the Log Type field, specify the type for this log.

   Available types are AddLog, ContextLog, FindLog, Log, MultiLog, and SingleLog. For more information, see the parameter descriptions for these elements in "Event Responses" on page 110.

Creating Event Handlers

This section describes how to create event handlers, using the All Event Handlers view.

Associating an event response to an event handler is optional.

The Event Handlers view tab in the All Event Responses view lets you view the event handlers with which the current event response is associated.

The Associated Event Logs view tab in the All Event Handlers view lets you associate an event log with the event response that is associated with the current event handler. The procedure is the same as that described in the section "Creating Event Responses" on page 67.

In Step 6 in the following procedure, associating a profile with an event handler is optional. It can help you avoid collisions if the same device event is available in multiple drivers.

NOTE: If you are implementing a custom communications driver, see also "Configuring Communications List of Values Types" on page 163.

To create an event handler

1. Navigate to Administration - Communications > All Event Handlers.
2. In the Event Handlers list, add a new record.
3. In the Name field, enter the name of the event handler.
4. Specify the configuration to associate this event handler with:
   a. In the Configuration field, click the select button. The Pick Configuration dialog box appears.
   b. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   c. Select an existing configuration with which you want to associate the event handler, then click OK.
5. Specify the event response to associate this event handler with:
   a. In the Event Response field, click the select button. The Pick Event Response dialog box appears, displaying existing event responses associated with the same configuration as the event handler.
   b. Select an event response with which you want to associate the event handler, then click OK.
6. Optionally, specify the profile to associate this event handler with:
   a. In the Profile field, click the select button. The Pick Profile dialog box appears, displaying existing profiles associated with the same configuration as the event handler.
   b. Select a profile with which you want to associate the event handler, then click OK.

   The name of the communications driver for the profile is displayed along with that of the profile.

   NOTE: If you change or remove a profile associated with the CTI configuration, you must update the values in the Profile field for the CTI configuration's event handlers.
7. In the Device Event field, enter the name of the device event this event handler will match.

   The device events you can specify are those supported by the communications driver for the associated profile, or special events.
8. In the Order field, specify an integer representing the order in which this event handler will be checked, relative to other event handlers specifying the same device event.
9. Enter any desired comments for the event handler.

Specifying Event Handler Parameters

This section describes how to add parameters for the current event handler.

To add parameters for the current event handler

1. Navigate to Administration - Communications > All Event Handlers.
2. In the Event Handlers list, select the event handler for which you want to add parameters.

3 Click the Event Handler Parameters view tab.

4 In the Event Handler Parameters list, add a new record.

5 In the Name field, specify the name of the parameter.

You can choose the parameter name from a list or type the name. For a parameter of type Group, choose the parameter name, then type the name of the subparameter after the period. Available event handler parameters are documented in "Event Handlers" on page 104.

6 In the Value field, enter the value for the parameter.

Specifying Event Response Parameters

This section describes how to add event response parameters for the event response associated with the current event handler.

To add parameters for the current event response

1 Navigate to Administration - Communications > All Event Handlers.

2 In the Event Handlers list, select the event handler for whose associated event response you want to add parameters.

3 Click the Event Response Parameters view tab.

4 In the Event Response Parameters list, add a new record.

5 In the Name field, specify the name of the parameter.

You can choose the parameter name from a list or type the name. For a parameter of type Group, choose the parameter name, then type the name of the subparameter after the period. Available event response parameters are documented in "Event Responses" on page 110.

6 In the Value field, enter the value for the parameter.

Defining Communications Commands

You can specify commands for your communications configuration. Siebel Communications Server supports two command types:

- Commands
- Command data definitions

In general, you create command data definitions first, then create commands and associate command data definitions with them.

Commands and command data definitions are provided by Siebel Systems. Verify whether the commands in your communications configuration meet your needs before you modify them or create new commands.

For more information about the role of commands within a communications configuration, see "About Communications Configuration Data" on page 20.

For information about configuring commands for the communications toolbar and for communications menu items, see Chapter 6, "Configuring User Interface Elements."

For detailed information about working with communications commands, including supported parameters, see Chapter 5, "Configuring Events and Commands."

Creating Command Data Definitions

This section describes how to create command data definitions, using the All Command Data view. The Commands view tab in the All Command Data view lets you view the commands with which the current command data definition is associated.

NOTE: If you are implementing a custom communications driver, see also "Configuring Communications List of Values Types" on page 163.

To create a command data definition

1. Navigate to Administration - Communications > All Command Data.
2. In the Command Data list, add a new record.
3. In the Name field, enter the name of the command data definition.
4. Specify the configuration to associate this command data definition with:
   a. In the Configuration field, click the select button. The Pick Configuration dialog box appears.
   b. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   c. Select an existing configuration with which you want to associate the command data definition, then click OK.
5. Enter any desired comments for the command data definition.

To add parameters for the current command data definition

1. Click the Command Data Parameters view tab.
2. In the Command Data Parameters list, add a new record.
3. In the Name field, specify the name of the parameter.

   You can choose the parameter name from a list or type the name. For a parameter of type Group, choose the parameter name, then type the name of the subparameter after the period. Available command data parameters are documented in "Command Data" on page 142.
4. In the Value field, enter the value for the parameter.

Creating Commands

This section describes how to create commands, using the All Commands view.

Associating a command data definition to a command is optional. For example, commands that contain subcommands do not need a command data definition.

The Commands view tab in the All Command Data view lets you view the commands with which the current command data definition is associated.

In Step 6 in the following procedure, associating a profile with a command is optional. It can help you avoid collisions if the same device command is available in multiple drivers.

NOTE: If you are implementing a custom communications driver, see also "Configuring Communications List of Values Types" on page 163.

To create commands

1. Navigate to Administration - Communications > All Commands.
2. In the Commands list, add a new record.
3. In the Name field, enter the name of the command.
4. Specify the configuration to associate this command with:
   a. In the Configuration field, click the select button to display the Pick Configuration dialog box.
   b. Select an existing configuration with which you want to associate the command, then click OK.
5. Specify the command data to associate with this command:
   a. In the Command Data field, click the select button. The Pick Command Data dialog box appears, displaying existing command data definitions associated with the same configuration as the command.
   b. Scroll to display any additional records that are not shown, or use Query or Find to locate records matching certain criteria.
   c. Select a command data definition you want to associate with the command, then click OK.
6. Optionally, specify the profile to associate this command with:
   a. In the Profile field, click the select button. The Pick Profile dialog box appears, displaying existing profiles associated with the same configuration as the command.
   b. Select a profile with which you want to associate the command, then click OK.

      The name of the communications driver for the profile is displayed along with that of the profile.
7. Enter any desired comments for the command.

Specifying Subcommands for a Group Command

This section describes how to specify subcommands for a group command. For more information about group commands that specify subcommands, see "Communications Group Commands in Toolbar" on page 159.

When you specify subcommands for a group command, do not specify invalid recursive relationships between subcommands and group commands.

For example, do not specify a subcommand that is itself a group command for which the current group command is specified as a subcommand. A relationship like this invalidates the communications configuration when it is loaded for an agent's session, and disables the communications toolbar.

To specify subcommands for a group command

1. Navigate to Administration - Communications > All Commands.
2. In the Commands list, create or select a command to which you add subcommands.
3. Click the Subcommands view tab.
4. In the Subcommands list, add a new record.
5. In the Subcommands field, click the select button to display the Pick Command dialog box.
6. Select an existing command that you want to serve as a subcommand for the current command, then click OK.
7. Specify the order for the subcommand.
8. Repeat Step 4 through Step 7, as necessary, for each subcommand.

Specifying Command Parameters

This section describes how to add parameters for the current command.

To add parameters for the current command

1. Navigate to Administration - Communications > All Commands.
2. In the Commands list, select a command to which you add parameters.
3. Click the Command Parameters view tab.
4. In the Command Parameters list, add a new record.

5 In the Name field, specify the name of the parameter.

You can choose the parameter name from a list or type the name. For a parameter of type Group, choose the parameter name, then type the name of the subparameter after the period. Available command parameters are documented in "Commands" on page 128.

For example, the DeviceCommand parameter lets you specify a device command to execute, such as for a communications driver to pass on to an external communications package such as CTI middleware.

6 In the Value field, enter the value for the parameter.

Specifying Command Data Parameters

This section describes how to add command data parameters for the command data definition associated with the current command.

To add parameters for the current command data definition

1 Navigate to Administration - Communications > All Commands.

2 In the Commands list, select a command for which you will add parameters to the associated command data definition.

3 Click the Command Data Parameters view tab.

4 In the Command Data Parameters list, add a new record.

5 In the Name field, specify the name of the parameter.

You can choose the parameter name from a list or type the name. For a parameter of type Group, choose the parameter name, then type the name of the subparameter after the period. Available command data parameters are documented in "Command Data" on page 142.

6 In the Value field, enter the value for the parameter.

Exporting and Importing Configuration Data

Siebel Communications Server provides import and export mechanisms that enable administrators to move communications configuration data between the database and ASCII text files. You can use these mechanisms to:

- Upgrade your implementations from previous versions of Siebel CTI (release 6.*x* and earlier)
- Move configuration data between databases or between communications configurations within the same database
- Move configuration data and custom communications drivers obtained from another source such as a third-party vendor into Siebel Communications Server

- Edit or enter data in text files rather than directly within the Administration - Communications screen

For more information about adding custom communications drivers, see Chapter 3, "Configuring Communications Drivers and Profiles."

For more information about upgrading Siebel CTI from previous versions, see Appendix D, "Upgrading from Release 6.x."

The basic options for exporting and importing are as follows:

- Export communications configuration data from the database to a .DEF file
- Import communications configuration data into the database from a .DEF file (or from an .ini file exported from a previous release)

  Table 6 describes how driver and profile parameters are handled by the import process from a .DEF file to the database.

Table 6. Parameter Management by the Import Process

If the parameter ...	For example ...
Does not exist in the database, the import process adds it.	If the .DEF file contains an entry: `Driver:CIMServer = "Blitzlab29"` The import process updates the database as follows: `Driver:CIMServer = "Blitzlab29"`

Exists in the database, the import process updates it.	If the database contains an entry:
	`CIMServer = "CHANGE_ME"`
	and the .DEF file contains an entry:
	`Driver:CIMServer = "Blitzlab29"`
	The import process updates the database as follows:
	`Driver:CIMServer = "Blitzlab29"`

Is deleted from the .DEF file, it remains in the database after the import process is completed.	If both the .DEF file and the database contain the entry: `Driver:CIMServer = "Blitzlab29"` but then you delete the entry from the .DEF file and run the import process, the database retains the entry that you deleted from the .DEF file.

For more information about the .DEF file format, see "Communications .DEF Files" on page 78.

NOTE: Using the Export Configuration and Import Configuration features, you can transfer all types of communications configuration data from one configuration to another *except* data for agents (and ACD queues) and telesets (and extensions). Driver and profile data could not be exported or imported prior to version 7.0.

Exporting Communications Configuration Data

Exporting communications configuration data from the database into a file can make it easier to review your configuration settings. You may want to do this before you put your system into production, or when you consult with Siebel Technical Support. Alternatively, you may prefer making communications configuration data modifications in text files, and then importing this data.

For instructions on exporting CTI configuration data from a previous version, such as release 6.x, see the Siebel CTI documentation (*Siebel CTI Guide*) for that version.

To export communications configuration data

1. Navigate to Administration - Communications > All Configurations.
2. In the Configurations list, select the configuration record for which you want to export data.
3. Click Export Configuration, located at the top of the view.

   The Export Configuration dialog box appears. It contains four export options:

   - **Configuration Parameters.** Exports configuration parameters. This data is displayed in the All Configurations view, in the Configuration Parameters list.
   - **Commands.** Exports commands and command data. This data is displayed in the All Commands and All Command Data views.
   - **Events.** Exports event handlers, event responses, and event logs. This data is displayed in the All Event Handlers, All Event Responses, and All Event Logs views.
   - **Drivers and Profiles.** Exports communications drivers and profiles. This data is displayed in the Communications Drivers and Profiles view.
4. Click the applicable check boxes named in Step 3 to specify what kinds of data to export, then click Next.

   The following message is displayed: "Please press OK button to begin exporting. The process may take several minutes. Please wait while exporting..."
5. Click OK.

   A File Download window prompts you to open or save this file.
6. Specify whether to open or save the file, then click OK:

   - If you choose to open, the .DEF file is opened in your default text editor.

- If you choose to save, you are prompted to choose a location and file name to save to. Click Save to export to this destination file.

NOTE: When you export a configuration, the prompted filename has the extension .DEF. However, files when exported may be appended by .txt or another extension, depending on your system.

7 After the export has completed, close the Export Configuration dialog box.

8 Rename the exported file to change the extension to .DEF.

Importing Communications Configuration Data

You may need to import communications configuration data from a file into the database for several reasons. You can import data previously exported from the same software version, or from a previous version (Siebel CTI). You can also use .DEF files to import data that you previously exported from another configuration or another database—for example, exporting data from a test database and importing it into your production Siebel Database.

An import file must contain valid data and use correct .DEF file notation. The import file must be consistent with files exported from the current version of Siebel Communications Server, or with previous versions of Siebel CTI.

As described in "Upgrading Siebel CTI from Release 6.x (and Earlier)" on page 468, each configuration file you export from the current version has an entry like this:

```
[Siebel]
    CommServerVersion = "7.0"
```

CAUTION: For any communications configuration file you import that does *not* contain an entry as shown, upgrade steps will be performed on the file. Do not manually add such an entry to a configuration file unless the file is known to be valid for the current version and should not be upgraded.

For files created in previous versions, the upgrade scenario applies. When importing files from previous versions, import the .ini file first, then the .DEF file.

For more information about the .DEF file format, see "Communications .DEF Files" on page 78. For more information about upgrades, see Appendix D, "Upgrading from Release 6.x."

CAUTION: When you import communications configuration data, all existing data of the same type is first deleted from the database for the current configuration. New data is then inserted. You cannot cancel an import operation after you begin it. If you are not sure of the effects of importing configuration data, export the old configuration data to a .DEF file before you import the configuration data.

To import communications configuration data

1 Navigate to Administration - Communications > All Configurations.

2 In the Configurations list, create a new configuration record, or select the configuration record for which you want to import data.

   For information about creating configurations, see "Creating or Modifying a Communications Configuration" on page 50.

3 Click Import Configuration, located at the top of the view.

4 Click Next.

   The Import Configuration dialog box appears. It contains four import options:

   - **Configuration Parameters.** Imports configuration parameters. This data is displayed in the All Configurations view, in the Configuration Parameters list.
   - **Commands.** Imports commands and command data. This data is displayed in the All Commands and All Command Data views.
   - **Events.** Imports event handlers, event responses, and event logs. This data is displayed in the All Event Handlers, All Event Responses, and All Event Logs views.
   - **Drivers and Profiles.** Imports communications drivers and profiles. This data is displayed in the Communications Drivers and Profiles view.

5 Click the applicable check boxes named in Step 4 to specify what kinds of data to import.

6 Enter the name of the source file, or click Browse and select the file.

7 Click OK to import the specified configuration data from the source file.

   Wait for the import process to finish, as shown by the imported configuration elements displayed in the Siebel application user interface. After the import has completed, the Import Configuration dialog box closes.

Communications .DEF Files

This section describes the sample communications definitions (.DEF) files that are provided by Siebel Systems. The following .DEF files are provided:

- **multichannelA.def.** Contains communications definitions that correspond to the communications configuration named Multichannel configuration A. Supports Siebel Universal Queuing and the voice channel (for Siebel CTI Connect) and email channel (for Siebel Email Response).
- **callrouteA.def.** Contains communications definitions that support using Siebel Universal Queuing to route communications work items of the following channels: voice, email.

The sample .DEF files provided by Siebel Systems are in the following locations:

- `bin\`*`language_code`*, in the Siebel Server installation directory
- `bin\`*`language_code`*, in the Siebel Mobile Web Client installation directory

where *language_code* represents the language code for the installed software, such as ENU for U.S. English.

Siebel Communications Server does not directly use the .DEF files; they are samples only. However, in addition to the sample definitions, these files contain comments to help you understand how the communications configuration elements work.

The .DEF files contain the following types of information:

- Communications drivers and profiles
- Configuration parameters
- Communications commands
- Communications events

File Format for .DEF Files

The Siebel Communications Server .DEF file are text files that employ the following file format:

- The names in brackets ([ ]) identify elements such as drivers, profiles, commands, and events. Each bracketed name is followed by lines containing names and values for parameters.
- Field values are enclosed in double quotation marks (which are not used for this data within the Administration - Communications screen), and are indicated using an equals sign (=), as in the following example line from a communications command:

  ```
  DeviceCommand = "MakeCall"
  ```

- Parameters are used in .DEF files to represent some elements that, in the current version, do *not* specify using parameters when you work in the Administration - Communications screen. For example, the line below, for an event handler definition, associates the event handler with an event response:

  ```
  Response = "InboundConsumerCall"
  ```

- Any lines preceded by a semicolon (;) are not in effect and may contain explanatory comments.
- A file exported from release 7.x contains lines like the following:

  ```
  [Siebel]
      CommServerVersion = "7.0"
  ```

  This allows the export file to be handled differently (as an upgrade case) than files exported from previous versions. Do not remove these lines from your export files. (Files exported from versions subsequent to version 7.0 may include an updated version number.)

If you need to manually prepare a file for import, you can export a file of the same type containing preconfigured data, then check that your file to be imported uses the correct format.

Example Section from .DEF File

Each .DEF file includes lines like the ones in the following example, from a configuration that supports Siebel CTI Connect.

```
; --- Handle inbound customer call ---
```

```
[EventHandler:InboundConsumerCall]
    DeviceEvent            = "TpAnswered"
    Response               = "InboundConsumerCall"
    Filter.CollectedDigits = "?*"
    Order                   = "3"

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

[EventLog:LogIncomingCallConsumerFound]
    Display                      = "TRUE"
    BusObj                       = "Consumer"
    BusComp                      = "Action"
    LogField.Type                = "Call - Inbound"
    LogField.'Contact Id'        = "{Consumer.Id}"
    LogField.Description         = "Inbound consumer call"
    LogField.'Call Id'           = "{refId}"
    AfterWork.'ACD Call Duration' = "{@WorkDuration}"

[EventLog:LogIncomingCallConsumerNotFound]
    BusObj                       = "Consumer"
    BusComp                      = "Action"
    LogField.Type                = "Call - Inbound"
    LogField.Description          = "Unknown Consumer CSN({dtmfDigits})"
    LogField.'Call Id'           = "{refId}"
    AfterWork.'ACD Call Duration' = "{@WorkDuration}"
```

5 Configuring Events and Commands

This chapter provides information about configuring events and commands for session communications, such as for Siebel CTI, to serve your business needs. It includes the following topics:

About Events and Commands

You can customize the way Siebel Communications Server operates for functions supported by the Communications Session Manager server component:

NOTE: Communications events and commands apply only to communications involving interactive drivers that control communications toolbar functionality, such as for voice and email channels.

- You can customize the way Communications Server handles *events* received from the communications system, such as your CTI middleware. Examples of events are when an inbound work item such as a voice call or email message is routed to an agent, when the agent accepts an inbound work item, when an agent completes a conference call, and so on.
- You can customize *commands* that are to be sent from Communications Server to the external communications system, or that are to invoke some other function. Examples of commands are when an agent accepts an inbound work item, initiates an outbound work item, transfers a work item, and so on.

You can customize event or command definitions—for example, to attach different types of data to incoming, outgoing, or transferred communications work items.

For an overview of communications configuration data, see "About Communications Configuration Data" on page 20.

For information about how to use views in the Administration - Communications screen to modify events and commands, see "Defining Communications Commands" on page 70 and "Defining Communications Events" on page 65.

Communications Definition Data in the Database

The Siebel Database contains configuration data that helps determine how your communications system is integrated with your Siebel application. Among other functions, the communications configuration data:

- Defines how a Siebel application handles events received from the communications system
- Defines how commands are generated and sent from the Siebel application to the communications system
- Defines the appearance and availability of commands in the Communications submenu in the Siebel application—by specifying hot keys, menu command titles, menu item sequence, and so on

NOTE: The events recognized by the communications driver, and the commands supported by the driver, are a subset of those supported by the communications system, such as your CTI middleware. For the supported driver parameters, events, and commands, see Chapter 12, "Using Siebel CTI Connect."

To customize the communications configuration, you must understand your call center's workflow model, as well as the key business objects for your Siebel application.

You customize communications features by editing or entering data in a series of views in the Administration - Communications screen, as described in Appendix C, "Views for Communications Administration."

NOTE: For some examples in this chapter, the label Communications Simulator indicates an example that you can try using communications simulation. Use the Communications Simulator to test aspects of your configuration before deployment. For more information, see "Enabling Session Communications and Simulation" on page 240 and "Simulating a Communications Environment" on page 211.

Communications Data Sets

A communications *data set* defines the commands, events, and associated data passing between the elements of your communications system—such as between your CTI middleware and the communications-enabled Siebel application that an agent is running. Events and commands have attributes and corresponding data.

Data set attributes can be supplied by:

- External modules such as call control tables, campaign manager modules, predictive dialers, voice response units, call routers, and so on.
- Siebel features or modules that play a role in supporting inbound communications—for example, Siebel CTI Connect, Siebel Universal Queuing, Siebel Email Response, Siebel Workflow, Siebel business service methods, and so on.

Each attribute has a name and contains associated data, such as the telephone number for the caller or for the caller's organization.

To see examples of communications data sets, you can view the Siebel Communications Server log file for the communications drivers you use, and for your external communications systems—for example, ctc.log for Siebel CTI Connect (using Intel NetMerge middleware).

Each type of communications event generated has a unique name and an associated data set. Each event can be handled individually by the communications system. Similarly, commands the Siebel client sends to the communications system can accept parameter strings and data sets.

For example, a command to transfer a work item can, along with parameter data specifying data such as a phone extension, accept any data set. When transferring the work item, the data set might include a service request ID or context information for the agent's current Siebel view.

Communications Server associates the data set with the work item ("attaches data") and sends the data set to the Siebel client of the agent receiving the transferred work item. A screen pop may be generated for the receiving agent, causing the current Siebel view to be displayed with a relevant record—such as a customer's current service request.

Event and Command Definitions

This section lists the types of events and commands supported by Siebel Communications Server, describes the field types for event and command parameters, describes how wildcard characters can be used in parameter values, and notes the form in which parameters are defined in events or commands.

Communications configuration parameters, communications drivers and profiles control the feature set provided by the communications system. Event and command definitions control how and when these features are used, how data attached to communications work items affects application behavior, and so on. Each event or command definition must have a unique name within its type.

Each communications command definition or event definition has a unique name within its type, and includes a set of parameters for which values are specified. Each type of event or command, and its supported parameters, is described in detail later in this chapter.

Event and Command Types

The following types of events and commands are defined in a communications configuration:

- **Event.** Communications Server supports three types of event definitions:
  - **Event handler.** Defines how the Siebel client application responds to events in the communications system. Specifies a response to execute when an event handler matches an event. For details, see "Event Handlers" on page 104.
  - **Event response.** Determines a specific response to an event that matched the event handler that invokes the response. For details, see "Event Responses" on page 110.
  - **Event log.** Defines Siebel log-generation rules for communications events. For instance, events may be logged as activities (Action business component) or logged as another type of record. For details, see "Event Logs" on page 123.
- **Command.** Defines communications commands that can be invoked in the Siebel application and specifies how they are to appear in the application. For details, see "Commands" on page 128.

- **Command data.** Defines data associated with a communications command that is invoked. For details, see "Command Data" on page 142.

Event and Command Parameters

This section provides general information about parameters you can define for your event and command definitions.

Field Types for Event and Command Parameters

Parameters for each type of event or command use one of the following field types:

- Characters (letters, numerals, _ (underscore), and special characters used in macro-expansion)

  Macro expansion can be used in fields of character type, as described in the section "Using Macro Expansion for Character Fields" on page 181. You can use the macros described in the section "Macros for Parameter Values" on page 182.

- Boolean (allowable values are TRUE and FALSE)
- Numeric (numerals 0–9)

Forms of Event and Command Parameters

Parameters of character type can be one of the following forms:

- Single parameter

  The parameter must be unique within the definition.

- Multivalue

  The parameter name does not need to be unique within the definition. Multiple instances of a multivalue parameter can be defined; each instance has a different value.

- Group of subparameters

  The parameter name must be unique within the definition, but multiple instances of the parameter can be defined, each specifying a unique subparameter. For instance, if you have parameters called Filter.A and Filter.B, A and B are subparameters of the group parameter Filter.

Table 7 illustrates the usage of parameters of the three forms of character fields. Parameters must be specified in a form similar to what is shown in Table 7.

Table 7. Event and Command Parameter Forms

Parameter Name	Parameter Value
SingleParam	Value
MultiValueParam	Value1
MultiValueParam	Value2

Table 7. Event and Command Parameter Forms

Parameter Name	Parameter Value
GroupParam.SubParam1	Value
GroupParam.SubParam2	Value

Wildcard Characters in Event and Command Parameters

The wildcard characters * (matching zero or more characters) and ? (matching exactly one character) can be used within communications event, or command parameter values of character type.

Use the wildcard characters to perform pattern matching on communications event data field values or on Siebel Database field values. Such matching can help determine which event or command is to be invoked.

You can use (? *) to match field values when the corresponding field value must not be empty. Separately, ? matches exactly one character, while * matches zero or more characters. The combination of these wildcard characters matches one or more characters.

NOTE: The characters * and ? do not function as wildcards for event or command parameters that pass parameters to a Siebel business service, a Siebel SmartScript, or a Siebel VB or Siebel eScript script.

Special Events for Device Events

This section describes special events, which are available for any communications configuration and are independent of the communications driver. Special events can be specified as device events in event handlers.

For more information about device events, see "Defining Communications Events" on page 65, and see the description for the DeviceEvent parameter in "Event Handler Parameters" on page 105.

For more information about defining event handlers, see "Event Handlers" on page 104.

The special events are not received from the communications system, such as CTI middleware, as are many other communications events for which a device event is specified.

NOTE: Special events are primarily used for defining event handlers in your communications configuration. However, you can also invoke them from scripts written using Siebel VB or Siebel eScript.

When using special events, you use WorkItemID, instead of SiebelWorkItemID, to refer to work item attributes. For example, you might define an event handler as follows:

```
[EventHandler:OnWorkItemStarted]
   DeviceEvent = "@PreWorkItemStartedEvent"
   FilterSpec = "[$GetWorkItemAttr(WorkItemID, ANI)] IS NOT NULL"
```

Many of the special events correspond to client handle methods, and include work item attributes. For more information, see "Work Item Attributes" on page 190. For more information about the client handle methods, see "Methods of ISC_CLIENT_HANDLE" on page 414.

Special Event Attributes

Many of the special events shown in Table 8 on page 87 have some or all of the following attributes:

- **ChannelType.** The language-independent value representing the channel type of this work item.
- **ProfileName.** The communications driver profile for which this work item is applicable.
- **WorkItemID.** The ID number of this work item.
- **WorkItemMode.** The mode of this work item, where 1 represents an inbound work item, and 2 represents an outbound work item.

List of Special Events

The special events, which have names that start with the symbol @, are described in Table 8.

Table 8. Special Events

Event Name	Description
@HandleNonRealtimeWorkItem	An event representing a nonreal-time work item received by an agent from Siebel Universal Queuing. No communications channel is associated with such a work item, which may represent something like a service request that has been assigned to the agent. Administrators may want to configure the communications configuration to respond to such work items in certain ways. Following are an event handler that specifies this type of event as the device event, and an event response that is in turn invoked: [EventHandler:HandleNonRealTimeWorkItem] DeviceEvent="@HandleNonRealTimeWorkItem" Response="HandleNonRealTimeWorkItem" Order="50" [EventResponse:HandleNonRealTimeWorkItem] QueryBusObj="Service Request" QueryBusComp="Service Request" QuerySpec="[Id]='{SRID}'" SingleField.'Owned By Id'="{@UserId}" The event response in this example sets the value of the "Owned By Id" field for the service request record to the ID of the assigned agent. For more information about nonreal-time work items, see *Siebel Universal Queuing Administration Guide*.
@PreIndicateNewWorkItemEvent @PostIndicateNewWorkItemEvent	Respectively, these special events correspond to just before, and just after, the client handle method IndicateNewWorkItem is invoked. These special events have the following attributes: ChannelType ProfileName WorkItemID WorkItemMode

Table 8. Special Events

Event Name	Description
@PreWorkItemReleasedEvent @PostWorkItemReleasedEvent	Respectively, these special events correspond to just before, and just after, the client handle method WorkItemReleased is invoked. These special events have the following attributes: ■ ChannelType ■ ProfileName ■ WorkItemID
@PreWorkItemResumedEvent @PostWorkItemResumedEvent	Respectively, these special events correspond to just before, and just after, the client handle method WorkItemResumed is invoked. These special events have the following attributes: ■ ChannelType ■ ProfileName ■ WorkItemID
@PreWorkItemStartedEvent @PostWorkItemStartedEvent	Respectively, these special events correspond to just before, and just after, the client handle method WorkItemStarted is invoked. These special events have the following attributes: ■ ChannelType ■ ProfileName ■ WorkItemID

Table 8. Special Events

Event Name	Description
@PreWorkItemSuspendedEvent @PostWorkItemSuspendedEvent	Respectively, these special events correspond to when the client handle method WorkItemSuspended is invoked. These special events have the following attributes: ■ ChannelType ■ ProfileName ■ WorkItemID
@RuntimeEvent	Corresponds to a Siebel runtime event, which was defined in the Administration - Runtime Events screen. This special event has the following attributes: ■ Context – The business service context. ■ ActionSet – The name of the action set. ■ Action – The name of the action. ■ EventId – The ID of the runtime event. ■ Event Name – The name of the runtime event (for example, "SetFieldvalue"). ■ Sub Event – The name of the subevent. ■ Event Type – The event type specified in the "Object Type" field. ■ Object Name – The object name. ■ BusObjName – The business object name. ■ BusCompName – The business component name. ■ ActiveRowID – The current active row ID of the business component. ■ ViewName – The current view name. For more information about configuring runtime events, see *Siebel Personalization Administration Guide*.

Special Commands for Device Commands

This section describes special commands, which are available for any communications configuration and are independent of the communications driver. Special commands can be specified as device commands in commands.

For more information about device commands, see "Defining Communications Commands" on page 70, and see the description for the DeviceCommand parameter in "Command Parameters" on page 130.

For more information about defining commands, see "Commands" on page 128.

The communications driver does not receive these commands and so does not pass them to a system such as CTI middleware, as with many other device commands you specify. Some special commands may, however, require another module such as Siebel Universal Queuing.

NOTE: Special commands are primarily used for defining commands in your communications configuration. However, you can also invoke them from scripts written using Siebel VB or Siebel eScript.

Each special command allows attachment of *any* user-defined key/value pair to the work item. Within the communications configuration, these key/value pairs are represented by event or command parameters and the associated parameter values.

Commands in which special commands are specified as device commands support the event and command parameters documented in Chapter 5, "Configuring Events and Commands." In your command definitions, you can specify custom subparameters for command parameters of type Group.

The special commands starting with "@UQ" are applicable only to Siebel implementations using Siebel Universal Queuing. These special commands invoke methods of business services for Siebel Universal Queuing. Command examples can be found in the sample configurations provided by Siebel Systems. For information about Siebel Universal Queuing business services, see *Siebel Universal Queuing Administration Guide*.

List of Special Commands

The special commands, which have names that start with the at sign (@), are described in Table 9 on page 91. An asterisk (*) before a parameter name means that the parameter is optional for this command. Command parameters are described in Table 10 on page 102.

Table 9. Special Commands

Command Name	Parameters	Description
@Associate	*bus_comp_field*	Updates a work item's tracking object. This object is specified using the event logging feature and the AfterWork event log parameter. **NOTE:** In order for a command to be enabled that is based on the @Associate special command, all applicable event log definitions must include the AfterWork parameter. AfterWork.'ACD Call Duration'="{@WorkDuration}" For more information about the AfterWork parameter, see "Event Logs" on page 123. To associate the tracking object to an account record, for example, the associated command data definition includes parameters like these, to specify the business component and field: BusComp ="Account" Param."Account Id"="{Id}" Parameter names enclosed in single or double quotes, as shown for Param."Account Id", are interpreted as field names. Values for such a parameter—enclosed in quotes on the right—are interpreted as field values, for which macro-expansion is supported. (The quotes enclosing the parameter values in this example are *not* part of the parameter values.) The @Associate command always operates on the selected work item. This is the work item that has the current focus—which an agent has selected from the Work Items list in the communications toolbar. For more information about event logging, see "Event Logs" on page 123.

Table 9. Special Commands

Command Name	Parameters	Description
@CreatePopupFrame	*AppletMode *Dimension *PageURL	Causes the browser to display a window in which to display content. The browser window content can be filled in as appropriate for a specified URL, based on what you specify for the PageURL parameter. Alternatively, it can display an applet. For example, a command definition using @CreatePopupFrame as the device command could have an associated command data definition that provides a URL to be displayed in the window—for example: Param.PageURL="http://www.siebel.com" The URL can also be specified without the prefix "http://", if the URL begins with "www.". A third-party company integrating with Siebel applications can use this method to open a new browser window and connect to its own Web server for integration purposes. In order to display an applet, the command data for a command using @CreatePopupFrame should include the following parameters: Param.AppletMode="*mode_name*" SelectParam="TRUE" SelectApplet="*applet_name*" where *mode_name* is either Edit or Base (use Edit for form applets and Base for list applets), and *applet_name* is the applet name in Siebel Tools. Base is the default value for AppletMode. The browser window dimensions can be specified using the Dimension parameter. For example, the following parameter creates a window with dimensions 500 x 300 pixels: Param.Dimension="500x300" If the Dimension parameter is not defined, the dimensions are determined by the applet specified using SelectApplet, or by the browser.

Table 9. Special Commands

Command Name	Parameters	Description
@InvokeSWECommand	SWECommand	Invokes a command on the Siebel Web Engine (SWE). The SWE command is specified using the SWECommand parameter. For example, for the command SendFaxGroup, which uses @InvokeSWECommand as the device command, this command data parameter is defined: Param.SWECommand="Send Fax (SWE)" All of the Send commands (Send Email, Send Fax, and Send Page) are invoked using @InvokeSWECommand.
@OpenView	View	Navigates the Siebel application to the specified view. In the command data definition, specify the view name as defined in Siebel Tools—for example: Param.View="Service Contact Detail View"
@UpdateRecord	*bus_comp_field*	Updates a database record for the current business component. For example, if the current view is based on the Service Request business component, you can change a status-related field for this business component. In the command data definition, specify parameters based on what business component field to update and what values the field supports. For example: BusComp="Service Request" Param.'Status'="Pending"
@UQAbortWorkItem	WorkItemID	Notifies Siebel Universal Queuing that the work item that was assigned to the agent has been aborted. For example, for a routed voice call, if a calling customer hangs up before the agent answers, this command can be used to notify Siebel Universal Queuing of the outcome. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{@SelectedWorkItem: UQWorkItemID}"

Table 9. Special Commands

Command Name	Parameters	Description
@UQAcceptWorkItem	WorkItemID	Notifies Siebel Universal Queuing that the agent has accepted the work item that was assigned to the agent. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{@SelectedWorkItem: UQWorkItemID}"
@UQAddWorkItem	MediaType WorkItemID **workitem_attr1* **workitem_attr2...*	Requests Siebel Universal Queuing to create a new work item, typically a nonreal-time work item such as a service request. In the command data definition, you can specify the MediaType parameter as follows, for example: Param.MediaType="Service Request" The language-independent strings that you use to represent the channel in the value for MediaType are from the List of Values type COMM_MEDIA_TYPE (where Parent LIC is set to "UQ"). Default channels include "Activity" and "Service Request". In the command data definition, specify the WorkItemID parameter. For example: Param.WorkItemID="{Id}" Additional work item attributes can be specified by the administrator to be passed to Siebel Universal Queuing. For example, define parameters like this: Param.MyWorkItemAttr1="{MyField1}" Param.MyWorkItemAttr2="{MyField2}"

Table 9. Special Commands

Command Name	Parameters	Description
@UQAgentAvailable	*MediaType	Notifies Siebel Universal Queuing that the agent is available for any or all supported channels: ■ When specified without the MediaType parameter, notifies Siebel Universal Queuing that the agent is available for all channels. ■ When specified with MediaType, notifies Siebel Universal Queuing that the agent is available for the channel specified using the MediaType parameter. The language-independent strings that you use to represent the channel in the value for MediaType are from the List of Values type COMM_MEDIA_TYPE (where Parent LIC is set to "COMMON"). Default channels include Email and Voice. In the command data definition, you can specify the MediaType parameter as follows, for example: Param.MediaType="Email"
@UQAgentChange ReadyState	*Reason	Toggles the Ready/Not Ready state for a specified channel. Optionally, a business component field can be specified from which a reason code is obtained at runtime. This special command must be specified with an attribute value representing a communications channel, as follows: @UQAgentChangeReadyState(*channel*) The language-independent strings that you use to represent the channel are from the List of Values type COMM_MEDIA_TYPE (where Parent LIC is set to "COMMON"). Default channels include Email and Voice. The channels you can specify are those for the drivers whose profiles are associated with the communications configuration for the agent. In the command data definition, you can specify the Reason parameter as follows: Param.Reason="[*reason_code*]" where *reason_code* represents a business component field.

Table 9. Special Commands

Command Name	Parameters	Description
@UQAgentInitAuxWork	*MediaType *Reason	Notifies Siebel Universal Queuing that the agent is unavailable for any or all supported channels, in order to perform auxiliary work. Optionally, a business component field can be specified from which a reason code is obtained at runtime: ■ When specified without the MediaType parameter, notifies Siebel Universal Queuing that the agent is unavailable for all channels. ■ When specified with MediaType, notifies Siebel Universal Queuing that the agent is unavailable for the channel specified using the MediaType parameter. The language-independent strings that you use to represent the channel in the value for MediaType are from the List of Values type COMM_MEDIA_TYPE (where Parent LIC is set to "COMMON"). Default channels include Email and Voice. In the command data definition, you can specify the parameters as follows, for example: Param.MediaType="Email" Param.Reason="[*reason_code*]" where *reason_code* represents a business component field.
@UQAgentLogon		Logs the agent into Siebel Universal Queuing.
@UQAgentLogout		Logs the agent out of Siebel Universal Queuing.
@UQAgentSignOnOff		Toggles the login/logout state of the agent for Siebel Universal Queuing. You can use this command to configure a toolbar button that will toggle the agent's login/logout state. Internally, this command invokes either @UQAgentLogin or @UQAgentLogout, according to whether the agent is currently logged off or logged on, respectively.

Table 9. Special Commands

Command Name	Parameters	Description
@UQBlindTransfer WorkItemToAgent	AgentLogin WorkItemID	Performs a blind transfer of the current work item to another agent. In the command definition, the FilterSpec parameter could be used as follows, to make sure a command is only activated for a particular channel, such as email: FilterSpec="[@SelectedWorkItem:ChannelType]='Email'" In the command data definition, specify the AgentLogin and WorkItemID parameters as follows: Param.AgentLogin="{Login Name}" Param.WorkItemID="{@SelectedWorkItem:UQWorkItemID}"
@UQBlockAgent Channel	AgentLogin MediaType	Blocks a particular inbound communications channel for an agent. This special command supports the sequential assignment feature for Siebel Universal Queuing.
@UQCancelTransfer	WorkItemID	Cancels the transfer of a work item through Siebel Universal Queuing. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{@SelectedWorkItem:UQWorkItemID}"

Table 9. Special Commands

Command Name	Parameters	Description
@UQChangeAgent MediaMode	AgentMediaMode MediaType	Notifies Siebel Universal Queuing about the agent's availability and work mode. The language-independent strings that you use to represent the channel in the value for MediaType are from the List of Values type COMM_MEDIA_TYPE (where Parent LIC is set to "COMMON"). Default channels include Email and Voice. In the command data definition, you can specify the parameters as follows, for example: Param.MediaType="Email" Param.AgentMediaMode="*N*" where *N* is either 1 or 2. These values are defined for the ChangeAgentMediaMode business service method for Siebel Universal Queuing.
@UQCompleteTransfer	WorkItemID	Completes the transfer of a work item through Siebel Universal Queuing. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{@SelectedWorkItem: UQWorkItemID}"
@UQComplete WorkItem	WorkItemID	Notifies Siebel Universal Queuing that the agent has completed the assigned work item. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{@SelectedWorkItem: UQWorkItemID}"
@UQHoldWorkItem	WorkItemID	Notifies Siebel Universal Queuing that the agent has suspended the assigned work item. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{@SelectedWorkItem: UQWorkItemID}"

Table 9. Special Commands

Command Name	Parameters	Description
@UQInitTransfer	AgentLogin WorkItemID	Initiates a transfer of a work item through Siebel Universal Queuing. In the command data definition, specify the parameters as follows: Param.AgentLogin="{Login Name}" Param.WorkItemID="{@SelectedWorkItem:UQWorkItemID}"
@UQRejectWorkItem	WorkItemID	Notifies Siebel Universal Queuing that the agent has rejected the assigned work item. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{$GetInboundWorkItemAttr(*channel*, UQWorkItemID)}" where *channel* represents the channel type for the work item.

Table 9. Special Commands

Command Name	Parameters	Description
@UQTransfer	AgentLogin TransferCmd WorkItemID	Transfers a work item in two steps, through Siebel Universal Queuing. This command invokes @UQInitTransfer, then invokes a specified transfer command, then invokes either @UQCompleteTransfer or @UQCancelTransfer. For example, if the command in which @UQTransfer is specified matches based on channel type (such as email), then a specified transfer command for that channel can be invoked: ■ If the specified transfer command is successful, then @UQCompleteTransfer is invoked and the work item is transferred. ■ If the specified transfer command is not successful, then @UQCancelTransfer is invoked, and the work item is returned to the original agent. In the command data definition, specify parameters as follows: Param.AgentLogin="{Login Name}" Param.TransferCmd="*command_name*" Param.WorkItemID="{@SelectedWorkItem:UQWorkItemID}"
@UQTransferWorkItemToRoute	RouteID WorkItemID	Transfers a work item to a route in Siebel Universal Queuing. In the command data definition, specify parameters as follows: Param.RouteID="[*route_name*]" Param.WorkItemID="{@SelectedWorkItem:UQWorkItemID}" For information about values you can specify for RouteID, see *Siebel Universal Queuing Administration Guide*.
@UQUnBlockAgentChannel	AgentLogin MediaType	Unblocks (makes available) a particular inbound communications channel for an agent. This special command supports the sequential assignment feature for Siebel Universal Queuing.

Table 9. Special Commands

Command Name	Parameters	Description
@UQUnHoldWorkItem	WorkItemID	Notifies Siebel Universal Queuing that the agent has resumed the assigned work item. In the command data definition, specify the WorkItemID parameter as follows: Param.WorkItemID="{@SelectedWorkItem: UQWorkItemID}"
@ViewWorkObject	View	Views a work-tracking object for a work item. Navigates to the view, such as a view in the Activities screen, where you can view the tracking object's record. @ViewWorkObject can be invoked from a business service, but not directly from a script. **NOTE:** In order for a command to be enabled that is based on the @ViewWorkObject special command, all applicable event log definitions must include the AfterWork parameter. AfterWork.'ACD Call Duration'="{@WorkDuration}" For more information about the AfterWork parameter, see "Event Logs" on page 123. A command data parameter with the name Param.View must exist and must contain the name of the view in which to view the tracking object. For example, if the tracking object is Action (for activities), then this record can be found in the view specified as follows: Param.View="Activity Attachment View" The @ViewWorkObject command always operates on the selected work item. This is the work item that has the current focus—which an agent has selected from the Work Items list in the communications toolbar. **NOTE:** Work-item duration is updated for the work-tracking object (such as an activity record) when a work item is released. This update may conflict with changes made by the agent. Consequently, an agent viewing a work-tracking record must save any changes before releasing the work item. To make changes after releasing the work item, the record must first be refreshed.

Special Command Parameters

Table 10 lists special command parameters and their usage. Table 9 on page 91 identifies the commands that use each of these parameters.

Within the communications configuration, special command parameters and values can be specified within command data definitions as subparameters of the Param parameter. For more information, see "Command Data" on page 142.

Table 10. Parameters for Special Commands

Command Parameter	Description
AgentLogin	For @UQBlindTransferWorkItemToAgent, @UQBlockAgentChannel, @UQInitTransfer, @UQTransfer, or @UQUnBlockAgentChannel, specifies the login name of an agent to whom a work item is routed or transferred through Siebel Universal Queuing.
AgentMediaMode	For @UQChangeAgentMediaMode, specifies the media mode for an agent.
AppletMode	For @CreatePopupFrame, specifies the mode for an applet to be displayed in the new browser window.
Dimension	For @CreatePopupFrame, specifies the dimension of the new browser window.
MediaType	For @UQAddWorkItem, @UQAgentAvailable, @UQAgentInitAuxWork, @UQBlockAgentChannel, @UQChangeAgentMediaMode, or @UQUnBlockAgentChannel, specifies the channel (media) type, such as the channel for which an agent changes availability.
PageURL	For @CreatePopupFrame, specifies a URL to be displayed in the new browser window.
Reason	For @UQAgentChangeReadyState or @UQAgentInitAuxWork, specifies a business component field from which a reason code is obtained. The agent chooses the reason code when changing availability for any or all channels.
RouteID	For @UQTransferWorkItemToRoute, specifies the name of a route defined for Siebel Universal Queuing.
SWECommand	For @InvokeSWECommand, specifies a Siebel Web Engine (SWE) command to be invoked.
TransferCmd	For @UQTransfer, specifies the name of a transfer command in the communications configuration that is to be invoked.
View	For @ViewWorkObject, specifies the name of a view.
WorkItemID	For several special commands (relating to Siebel Universal Queuing), specifies the ID for a work item upon which the command operates. Set the value as appropriate for the command.

Special Command Examples

This section describes examples of special commands.

@Associate Special Command Example

Table 11 and Table 12 provide an example command using the @Associate special command.

This example defines a command and associated command data that creates an association between the current activity object and the current account object. The example command is enabled only when the agent is working with accounts and when a work-item tracking object exists. For more information, see the description for @Associate in Table 9 on page 91.

Table 11. Command: AssociateAccount

Parameter Name	Parameter Value
DeviceCommand	@Associate
Hidden	TRUE

Table 12. Command Data: AssociateAccount

Parameter Name	Parameter Value
BusComp	Account
Param."Account Id"	{Id}

@ViewWorkObject Special Command Example

Table 13 and Table 14 provide an example command using the @ViewWorkObject special command.

This example defines a command and associated command data that allows a user to view the work-item tracking record. The command specifies the menu item to be used for viewing the record, and assigns the command to the hot key SHIFT+F8. The command is enabled only when a work-item tracking object exists. For more information, see the description for @ViewWorkObject in Table 9 on page 91.

Table 13. Command: ViewWorkObject

Parameter Name	Parameter Value
DeviceCommand	@ViewWorkObject
HotKey	SHIFT+F8
MenuPosition	8
Title	View Work Item Object

Table 14. Command Data: ViewWorkObject

Parameter Name	Parameter Value
Param.View	Activity Attachment View

Event Handlers

Communications events from the communications system are directly passed to Siebel Communications Server, in particular the Communications Session Manager, for further processing.

NOTE: Each event log specified in a single event response must be unique. (Event logs may be specified in the All Event Handlers view, but they are actually associated with the event response that is associated with the event handler.) For more information, see "Event Responses" on page 110.

Event Handler Process Overview

Communications events involve the following phases of activity:

1. An event or action, such as a phone call being disconnected, occurs in the communications system, such as a telephony switch. The switch forwards events to a communications middleware server, such as a CTI middleware server.
2. The middleware server, such as CTI middleware, forwards the event to the Communications Client business service.

3 The Communications Client business service processes the event and executes any actions defined in the configuration data in the database, or forwards events to business service methods or to Siebel VB or Siebel eScript code.

By working with event data in the Administration - Communications screen, you can define the actions to be performed when a particular communications event is received. Such actions are invoked immediately upon receipt of such an event.

Event handlers specify what kind of event from the communications system will be processed, and specify which event response is called as a result.

Event Handler Parameters

Table 15 describes the parameters available in event handlers in the communications configuration data. A dash (—) in the Macro column indicates that macro expansion is not applicable for the parameter. The character Y indicates that macro expansion applies to the parameter.

Table 15. Event Handler Parameters

Parameter	Type	Macro	Description
DeviceEvent	Char	—	Communications device-generated event name. For possible values, see the events table for the relevant driver in Chapter 12, "Using Siebel CTI Connect." **NOTE:** Do not specify this parameter as an event handler parameter. Rather, specify the device event for the event handler directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
Filter	Group	—	Event data-field filter. This works like a standard IsLike() function. All filter results are logically joined by AND to determine if the event matches this event handler. You can filter data received from the communications driver, data attached to the work item by the channel manager, or data attached or modified by an agent who previously handled the work item—such as a transferring agent. The Filter and FilterSpec parameters can be used together to evaluate event data-field data received with the device event associated with this event handler. Evaluations using the Filter parameter (AND conditions) occur before evaluations using the FilterSpec parameter (compound predicate). For information on fields available for filtering, see the event data fields table for the relevant driver in Chapter 12, "Using Siebel CTI Connect."

Table 15. Event Handler Parameters

Parameter	Type	Macro	Description
FilterSpec	Char	Y	Event data-field filter that supports simple or complex queries. Filter results are evaluated using a compound predicate that can include standard query operators to determine if the event handler matches the event. FilterSpec queries use standard comparison operators, including: = LIKE AND OR EXISTS > < >= <= For example: FilterSpec="[*attr1*] IS NOT NULL OR [*attr2*] LIKE '*value**'" where *attr1* and *attr2* are event data fields and *value* represents part of a comparison value for *attr2*. In this example, the event handler is evaluated for a match if *attr1* is not empty or if *attr2* is like the value of "value*". Here is another example: FilterSpec="[@UserName]='SADMIN'" In this example, @UserName is macro-expanded before the query is performed. The Filter and FilterSpec parameters can be used together to evaluate event data-field data received with the device event associated with this event handler. Evaluations using the Filter parameter (AND conditions) occur before evaluations using the FilterSpec parameter (compound predicate). See also the descriptions for the Filter event handler parameter and the QuerySpec event response parameter.

Table 15. Event Handler Parameters

Parameter	Type	Macro	Description
Order	Numeric	—	The order in which event handlers for which the same device event is specified will be tested against event matching. Each received communications event is checked against all event handlers to determine which response to execute. Event handlers with a lower Order value will be checked before those with higher values. The default is 0. The check stops when there is a match. **NOTE:** Do not specify this parameter as an event handler parameter. Rather, specify the order for the event handler directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
Profile	Char	—	Name of the communications driver profile that generates the device event associated with this event handler. If a profile is associated with the event handler, then the event handler is evaluated only for events received from the communications driver for that profile. See also "Creating Event Handlers" on page 68. **NOTE:** Do not specify this parameter as an event handler parameter. Rather, specify the profile for the event handler directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
Response	Char	—	Name of the event response that will be executed when a matching event is detected. **NOTE:** Do not specify this parameter as an event handler parameter. Rather, specify the event response for the event handler directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.

Table 15. Event Handler Parameters

Parameter	Type	Macro	Description
ServiceMethod	Char	—	The name of a Siebel business service and method to be called in order to evaluate the event handler for a match. You can use the ServiceMethod and ServiceParam parameters to supplement or replace filter mechanisms using the Filter or FilterSpec parameters. You can also use this method to execute custom code before executing a specified event response. The business method should set a parameter of Result to a value 1 or 0 (or TRUE or FALSE): ■ 1 (TRUE) indicates that this event handler matches and is executed, and the associated event response is executed. ■ 0 (FALSE) invalidates this event handler; the next event handler will then be evaluated. The service and method are specified in the form *service*.*method*. Optionally, you can use the ServiceParam parameter to provide parameter names and values to pass to the method to be called. For more information, see "Using Business Services with Communications Server" on page 216.
ServiceParam	Group	Y	A group of subparameters that are parameters to the Siebel business service method (if any) invoked using the ServiceMethod parameter. You create each parameter you need in the form ServiceParam.*param_name*, then specify a parameter value. The parameter name and value are both passed to the service method. Order parameters in the sequence in which they are expected by the service method: ServiceParam.Param1="value1" ServiceParam.Param2="name" Param1 and Param2 are subparameters of the ServiceParam parameter. For more information, see "Using Business Services with Communications Server" on page 216.

Handling an Inbound Call Received by an Agent

Communications events provide the following general options for handling device events for calls received by an agent. Such an inbound call may employ events like *call incoming* or *event ringing*, or *call connected* or *event established*, as defined for a communications system employing CTI.

The following list describes the options for handling device events:

- **Handle when call connected after agent answers (default).** In this scenario, the agent answers the call, triggering the event handler.

  The phone rings. The agent clicks Accept Work Item. The *call connected* or *event established* type of event is received. This event triggers handling such as to perform a query to generate a screen pop.

  In this case, an agent's work is not interrupted by an unexpected screen pop.

  The scenario above uses the TpAnswered device event for the Siebel CTI Connect driver.

- **Handle when agent answers (for Siebel CTI Connect).** In this scenario, the agent answers the call, triggering the event handler (before the TpAnswered event is received).

  The phone rings. The agent clicks Accept Work Item. The EventAnswerCall device event (for Siebel CTI Connect driver) is triggered when the AnswerCall command is invoked. This event triggers handling such as to perform a query to generate a screen pop. Then the call is connected, as described for the scenario called: Handle when call connected after agent answers.

  In this case, an agent's work is not interrupted by an unexpected screen pop.

  This scenario uses the EventAnswerCall device event for the Siebel CTI Connect driver.

  For more information, see "Using Device Event to Enhance Screen Pop Performance" on page 207.

- **Handle immediately, when phone rings.** In this scenario, the phone rings, triggering the event handler (before the agent answers the call).

  The phone rings. The *call incoming* or *event ringing* type of event is received. The event triggers handling such as to perform a query to generate a screen pop. The agent clicks Accept Work Item, then the connection is established.

  In this case, an agent's work could be interrupted by a screen pop when the phone rings, without any notifications or confirmations.

  This scenario uses the InboundCall device event for the Siebel CTI Connect driver.

NOTE: If the user has modified the current record, this record is automatically committed anytime an agent clicks a button on the communications toolbar or anytime a screen pop is triggered. If the user has not provided values for all required fields, a pop-up window prompts the user to provide the values to complete the record.

The Siebel sample communications configurations provide examples of some of these scenarios. You can implement other screen-pop behavior using Siebel VB or Siebel eScript.

Example of Using Event Handler to Handle Inbound Call

Here is an example event handler, named InboundCallReceived, that is based on the default inbound call screen-pop scenario described in the scenario, handle when call is connected after agent answers, above. For this event handler:

- Set the Order field to 0 to make sure it takes priority over any other event handlers that may otherwise match.
- Set the Device Event field to one of the following:
  - TpAnswered (for Siebel CTI Connect driver)
  - *event_name* (for another Adaptive Communications driver, where *event_name* corresponds to an event like TpAnswered)
- Include the following event handler parameter:

  Filter.ANI="*"

The example event handler is associated with an event response OnInboundCallReceived.

Because the Order field is set to 0, this event handler will be checked first for any event received where the device event is as specified. The event handler filters the event data field ANI. If this field exists, the event matches and invokes the event response OnInboundCallReceived.

To handle the *call incoming* or *event ringing* type of event without waiting for the agent to click Accept Work Item, you can create a similar event handler specifying the InboundCall device event for the Siebel CTI Connect driver. You might name such an event handler ImmediateRingingHandler, and invoke a similarly named response.

Event Responses

Event responses in the communications configuration data specify how the Siebel client will associate response actions with an event. Each event handler specifies a response that corresponds to an event response. The same event response can be initiated by different event handlers.

NOTE: Each event log specified in a single event response must be unique. When you associate event logs with an event response in the Siebel client, the user interface prevents you from associating the same event log more than once. For example, if you have already associated the event log OutboundActivityContactCall, using type Log, you will not be able to associate the same event log using another Log Type, such as SingleLog. If you create .DEF files to import, do not specify, within an event response definition, multiple event log parameters that specify the same value. If you do so, an error will be generated when you attempt to import this configuration data.

Event Response Process Overview

In executing the event response, Siebel Communications Server performs the following actions:

1. If a Siebel business service method name is specified, such as to update the customer dashboard, that method is invoked.

   If the method does not return the required result, such as a value of TRUE (1) for the output argument Continue, the response will stop executing. If an event log of type Log is specified, that event log is processed. Go to Step 13 on page 112.

   The business service method can also alter the event data fields that were associated with the work item, such as to add new fields.

   For more information about the parameters mentioned here, see Table 16 on page 112.

2. If a Siebel VB or Siebel eScript script name is specified, that script is invoked.

   If the script does not return the required result, such as Continue, the response will stop executing. If an event log of type Log is specified, that event log is processed. Go to Step 13 on page 112.

   For more information about invoking a Siebel VB or Siebel eScript script from an event response, see "Integrating with Siebel Scripting Languages" on page 222, and see the descriptions for the Script and ScriptParam parameters in Table 16 on page 112.

3. If a communications command is specified, the command is executed.

4. If the UseCtxData parameter is defined and set to TRUE, Communications Server checks to determine if the event has screen context data attached.

   If screen context data is available, the corresponding screen and view are displayed. This behavior is applicable only for screen pops and screen transfers. (In addition, the Thread Applet and Thread Field properties must be set for the view in order to support screen pops and screen transfers.)

   If an event log of type ContextLog is specified, that event log is processed. If ContextLog is not specified, but an event log of type Log is specified, that event log is processed. Go to Step 13 on page 112.

5. If the QueryBusObj, QueryBusComp, and QuerySpec parameters are defined, then the query specified by QuerySpec is macro-expanded and executed. If these parameters are not defined, go to Step 9; otherwise, go to Step 6.

   Step 7 and Step 8 define the views upon which this query will be displayed.

6. If there is only one row in the query result from QuerySpec, and if the QueryBusComp2 and QuerySpec2 parameters are defined, then the query specified by QuerySpec2 is macro-expanded and executed. If these parameters are not defined, go to Step 9; otherwise, go to Step 7.

7. If there is only one row in the query result from QuerySpec, then that row is displayed in the view specified by the SingleView parameter. Any field specified by SingleField is populated with data defined in the parameter value. Go to Step 9.

8. If there are several rows in the query result from QuerySpec, then those rows are displayed in the view specified by the MultiView parameter. Go to Step 9.

9. If the OpenView parameter is specified, then the view is displayed that is specified by OpenView. Go to Step 10.

10 If the AddBusComp, AddBusObj, AddRecordView, AddRecordApplet, and AddField parameters are defined, then the view is displayed that is specified by AddRecordView.

A new record is created in the applet specified by AddRecordApplet. Any field specified by AddField is populated with data defined in the parameter value. The record is not committed. Go to Step 13 on page 112.

11 If a Siebel SmartScript script name or other data is specified to be passed to Siebel SmartScript, that SmartScript is invoked.

For more information about invoking a SmartScript from an event response, see "Integrating with Siebel SmartScript" on page 228 and see the description for the SmartScript parameter in Table 16 on page 112.

12 If the FindDialog and FindField parameters are defined, the Search Center appears, populated with search criteria. The Search Center does not appear if the OpenView parameter is also specified as in Step 9.

13 If an event log is associated with this event response, then a log record is created using the object defined in the event log—for example, an activity record.

Multiple event logs can be associated, one of which matches, depending on how the event response executed. For example:

- If UseCtxData is invoked, as in Step 4 on page 111, a ContextLog log can be created.
- If SingleView is invoked, as in Step 7 on page 111, a SingleLog log can be created.
- If MultiView is invoked, as in Step 8 on page 111, a MultiLog log can be created.
- If AddBusComp, AddBusObj, AddRecordView, AddRecordApplet, and AddField are invoked, as in Step 10 on page 112, an AddLog log can be created.
- If FindDialog is invoked, as in Step 12, a FindLog log can be created.
- Otherwise, a log can be created as specified by Log.

For more information about event logs, see "Event Logs" on page 123.

Event Response Parameters

Table 16 describes the parameters available in event responses. A dash (—) in the Macro column indicates that macro expansion is not applicable for the parameter. The character Y indicates that macro expansion applies to the parameter.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
AddBusComp	Char	—	Business component name to use when adding a new record.
AddBusObj	Char	—	Business object name to use when adding a new record.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
AddField	Group	Y	Field names and predefined value when adding a new record.
AddLog	Char	—	Event object to execute if the application navigates to the view and applet specified by the AddRecordView and AddRecordApplet parameters. Determines the Siebel activity record to be created in response to communications events. For an example of using this parameter, see "Event Response Process Overview" on page 110. **NOTE:** Do not specify this parameter as an event response parameter. Rather, specify an event log of this type for the event response directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
AddRecordApplet	Char	—	Name of the applet that is displayed, in the view specified by AddRecordView, when adding a new record. If AddRecordApplet is not specified, the record is created in the first applet that can be inserted which has the same business component as specified with AddBusComp.
AddRecordView	Char	—	Name of the view that is displayed when adding a new record.
Command	Char	—	Optionally specifies the name of a command in the communications configuration that will be invoked when this event response is invoked.
ContextLog	Char	—	Event object to execute if screen-context data is used for screen display. Determines the Siebel activity record to be created in response to communications events. For an example of using this parameter, see "Event Response Process Overview" on page 110. **NOTE:** Do not specify this parameter as an event response parameter. Rather, specify an event log of this type for the event response directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
FindDialog	Char	—	Find Object name to determine what is displayed in the Search Center if no rows are returned.
FindField	Group	Y	Find Field names and default values to determine what is displayed in the Search Center if no rows are returned.
FindLog	Char	—	Event object to execute if the Search Center is displayed, as specified using FindDialog and FindField. Determines the Siebel activity record to be created in response to communications events. For an example of using this parameter, see "Event Response Process Overview" on page 110. **NOTE:** Do not specify this parameter as an event response parameter. Rather, specify an event log of this type for the event response directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
Log	Char	—	Event log to use if no other log is specified for an action. Determines the Siebel activity record to be created in response to a communications event. For an example of using this parameter, see "Event Response Process Overview" on page 110. **NOTE:** Do not specify this parameter as an event response parameter. Rather, specify an event log of this type for the event response directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
MultiLog	Char	—	Event object to execute if the view specified using the MultiView parameter is displayed. Determines the Siebel activity record to be created in response to communications events. For an example of using this parameter, see "Event Response Process Overview" on page 110. **NOTE:** Do not specify this parameter as an event response parameter. Rather, specify an event log of this type for the event response directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
MultiView	Char	—	Name of the view that will be displayed if the query result contains more than one row.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
OpenView	Char	—	Name of the view that will be displayed, regardless of query results. The view specified by OpenView is displayed only if a view specified by SingleView or MultiView is not displayed.
QueryAfterAnswer	Boolean	—	A parameter that, when set to TRUE, causes Communications Server to answer the call first and then execute the screen-pop query. When set to FALSE (the default setting), this parameter causes Communications Server to execute the query first and then answer the call. This parameter is valid only for an event response invoked by an event handler that follows the "handle immediately" model. For more information about screen pops, see "Handling an Inbound Call Received by an Agent" on page 109.
QueryBusComp	Char	—	Business component name for the query specified by QuerySpec.
QueryBusComp2	Char	—	Business component name for a second query, specified by QuerySpec2. The second query will be executed only if this parameter is specified and if the first query (specified by QuerySpec) returns exactly one row. This feature could be used to get screen pops related to campaigns. For example, the first query can locate the campaign that matches the call type—using DNIS to identify the number the caller dialed. The second query can locate the contact (within that campaign) whose work number matches that of the caller—using ANI to identify a caller's number.
QueryBusObj	Char	—	Business object name containing the business components specified by QueryBusComp and QueryBusComp2.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
QueryFields	Multi	—	Field names resulting from the business component query specified by QuerySpec. Only the fields specified here will be activated when the Siebel application performs the query. Since the same business component is used to navigate to the destination view, you can use this feature to specify the fields required for that view. If nothing is specified here, the navigation process will cause the business component to be queried again. QueryFields can speed up a screen-pop navigation by preventing additional queries of an entire business component.
QueryFields2	Multi	—	Field names resulting from the business component query specified by QuerySpec2. For more details, see the description of the QueryFields parameter.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
QuerySpec	Char	Y	Standard business component query. The query uses business component field names from the business object and business component specified with the QueryBusObj and QueryBusComp parameters. (No other fields can be specified.) Simple and complex queries are supported, using standard comparison operators, including: = LIKE AND OR EXISTS > < >= <= For example, if QuerySpec is defined as follows, then the query matches records in the current business component (assuming the business component contains the fields indicated) if either the work phone number or the home phone number matches the ANI value for the call: 'Work Phone #'='{ANI}' OR 'Home Phone #'='{ANI}' Alternatively, if QuerySpec is defined as follows, then the query matches records in the current business component (example is for Action business component) if the current user is one of the owners specified for the multivalue field Owned By: EXISTS('Owned By'='{@UserName}') The keyword EXISTS *must* be used when defining a query specification for a multivalue field.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
QuerySpec2	Char	Y	A second query that may be performed, after the query specified by the QuerySpec parameter. QuerySpec2 queries the business component specified by the QueryBusComp2 parameter. (Both QueryBusComp and QueryBusComp2 specify business components in the business object specified by the QueryBusObj parameter.) The result of the QuerySpec2 query may be used by Siebel SmartScript. For more information on how the query may be constructed, see the description for the QuerySpec parameter.
Script	Char	—	Siebel VB or Siebel eScript script name to invoke (if specified). Parameters are passed using the ScriptParam parameter. For more information about using Siebel VB or Siebel eScript, see "Integrating with Siebel Scripting Languages" on page 222.
ScriptParam	Group	Y	A group of subparameters that are parameters to the script method (if any) invoked using the Script parameter. You create each parameter you need in the form ScriptParam.*param_name*, then specify a parameter value that will be passed to the script. The parameter names themselves are not passed to the script. Order parameters in the sequence in which they are expected by the script: ScriptParam.Param1="value1" ScriptParam.Param2="name" Param1 and Param2 are subparameters of the ScriptParam parameter.
ServiceMethod	Char	—	The name of a Siebel business service and method to be called. The service and method are specified in the form *service.method*. Optionally, you can use the ServiceParam parameter to provide parameter names and values to pass to the method to be called. For more information, see "Using Business Services with Communications Server" on page 216.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
ServiceParam	Group	Y	A group of subparameters that are parameters to the Siebel business service method (if any) invoked using the ServiceMethod parameter. You create each parameter you need in the form ServiceParam.*param_name*, then specify a parameter value. The parameter name and value are both passed to the service method. Order parameters in the sequence in which they are expected by the service method: ServiceParam.Param1="value1" ServiceParam.Param2="name" Param1 and Param2 are subparameters of the ServiceParam parameter. For more information, see "Using Business Services with Communications Server" on page 216.
SingleField	Group	Y	When the result from the QuerySpec query is exactly one record, that record can be updated if you provide a field name and predefined field value, using this parameter. For example, in the following event response, QuerySpec will likely return a single record, then SingleField can update the Primary Owned By field to the current agent: [EventResponse:EmailWorkStarted] QueryBusObj="Email Response" QueryBusComp="Action" QuerySpec="Id='{ActivityID}'" SingleView="Communication Detail - Response View" SingleField.'Primary Owned By'="{@UserName}" Log=EmailWorkStarted
SingleLog	Char	—	Event object to execute if the view specified using the SingleView parameter is displayed. Determines the Siebel activity record to be created in response to communications events. For an example of using this parameter, see "Event Response Process Overview" on page 110. **NOTE:** Do not specify this parameter as an event response parameter. Rather, specify an event log of this type for the event response directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
SingleView	Char	—	Name of the view that will be displayed if the result from the QuerySpec query is exactly one row. QueryBusComp2, QuerySpec2, and QueryFields2 can be used to query a second business component of the QueryBusObj for a single record.
SmartScript	Group	Y	Parameter to be passed to Siebel SmartScript. Can include a script name or other data to pass to SmartScript. You can specify any of the following subparameters: SmartScript.CampaignId SmartScript.CampContactId SmartScript.ContactId SmartScript.LanguageCode SmartScript.ScriptId SmartScript.ScriptName Either SmartScript.ScriptName or SmartScript.ScriptId must be specified in order to launch a particular SmartScript. For more information, see "Integrating with Siebel SmartScript" on page 228.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
UseCtxData	Boolean	—	Enables or disables automatic use of screen-context data attached to an event. When UseCtxData is TRUE, and the event contains a Siebel screen bookmark, the bookmark will be used to perform a screen pop or screen transfer. The default is FALSE. **NOTE:** The size of a Siebel screen bookmark is defined by the middleware and communications driver and may vary according to each vendor.

Table 16. Event Response Parameters

Parameter	Type	Macro	Description
WorkObject	Char	Y	Specifies data identifying a work-item tracking database record that can be written to the database. This parameter can make work-item tracking possible even if your communications system does not fully support attaching data to a work item. The WorkObject parameter is not necessary for Siebel CTI Connect (using Intel NetMerge) or another CTI middleware package that supports call data attachment. This data is passed to the communications driver, which attaches the data to the call without requiring the WorkObject parameter. If you are using a custom driver developed for other middleware, you may need to use the WorkObject parameter in order to pass work-item tracking data from one agent to another, such as when a call is transferred or made as a conference call. Example: An agent receives a call and creates the activity record for it. When this agent transfers the call to a second agent, the second agent will want access to the same activity record. For the first agent's session, the row ID of the activity record is encoded, and the call is transferred. If the WorkObject parameter is defined appropriately, the second agent can continue tracking the call-activity record. Format values for this parameter as follows: *bus_obj_name*;*bus_comp_name*;*row_ID* For activity records, Action is the name of the business object and the name of the business component. Whether you require the WorkObject parameter, and how you define the parameter, depend on your communications system and driver. For example, with Aspect middleware, if variable E is used to store the row ID, define WorkObject as follows: Action;Action;{E} Whether this parameter definition works as intended depends on whether the communications driver expects to receive data in this form to pass to the receiving agent.

Event Response Example

Table 17 shows an example of an event response.

Table 17. Event Response: OnInboundCallReceived

Parameter Name	Parameter Value
QueryBusObj	Contact
QueryBusComp	Contact
QuerySpec	'Work Phone #'='{ANI}'
SingleView	Service Contact Detail View
MultiView	Contact List View
FindDialog	Service Request
FindField.CSN	Ask Caller
SingleLog	LogIncomingCallContactFound
MultiLog	LogIncomingCallMultiContactFound
FindLog	LogIncomingCallContactNotFound

In this example, the information specified with the business object and business component parameters is retrieved. Communications Server then performs a query on the phone number retrieved from the ANI event data field, to see if it matches a value in the Work Phone # field in the Contact business component. (SingleLog, MultiLog, and Log are specified using fields, rather than parameters.)

If QuerySpec returns a single record, the agent is navigated to the service requests view specified by SingleView. The event log LogIncomingCallContactFound, which is specified by the SingleLog parameter, creates a call activity record automatically.

If QuerySpec returns multiple records, the agent is navigated to the contacts view specified by MultiView. The event log LogIncomingCallMultiContactFound, which is specified by the MultiLog parameter, creates a call activity record automatically.

If QuerySpec returns no records, the Search Center appears and displays “Ask Caller” in the CSN field. The event log LogIncomingCallContactNotFound, which is specified by the FindLog parameter, creates a call activity record automatically.

Event Logs

Event logs in the communications configuration data define log-generation rules for communications activity. Some event responses have an associated event log.

NOTE: Each event log specified in a single event response must be unique. Define separate event logs for each such usage context, accordingly. For more information, see “Event Responses” on page 110.

Event Log Parameters

Table 18 describes the parameters available in event logs. A dash (—) in the Macro column indicates that macro expansion is not applicable for the parameter. The character Y indicates that macro expansion applies to the parameter.

Table 18. Event Log Parameters

Parameter	Type	Macro	Description
AfterWork	Group	Y	Field names and values, from the log business component, that will be used to update the work-tracking log record when the work item has ended or been released. **NOTE:** After the work item has been released, the log record that is modified according to the fields specified with this parameter is refreshed when the agent clicks off the record, then clicks on it again.
BusComp	Char	—	Log business component name.
BusObj	Char	—	Log business object name.
Display	Boolean	—	Parameter that, when it is set to TRUE and the current view contains a business component specified in the event log, uses the current business component to add a new log record. The log record appears on the screen immediately, and is selected. If Display is set to FALSE, the log record is generated but is not selected, and might not be displayed in the current record set. The default is FALSE. **NOTE:** While the work item is active, the agent can modify the log record. After the work item has been released, however, if AfterWork is set, an agent's changes may conflict with how the log record was written by the system at that time. The agent can click off the record, then click on it again in order to modify it.
LogField	Group	Y	Field names and values of the log business component.
QuerySpec	Char	Y	Parameter that can be used to retrieve the desired record in the business component: ■ If QuerySpec is not defined, a new record is created for the event log. ■ If QuerySpec is defined and returns a single record, this record is used for the event log. ■ If QuerySpec is defined and returns more than one record, a new record will be created for the event log.

Table 18. Event Log Parameters

Parameter	Type	Macro	Description
ServiceMethod	Char	—	The name of a Siebel business service and method to be called. The service and method are specified in the form *service*.*method*. Optionally, you can use the ServiceParam parameter to provide parameter names and values to pass to the method to be called. If ServiceMethod and ServiceParam are specified, they are executed first when the event log is invoked. If the specified business service method does not return the required result, such as a value of TRUE (1) for the output argument Continue, the event log will stop executing. For more information, see "Using Business Services with Communications Server" on page 216.

Table 18. Event Log Parameters

Parameter	Type	Macro	Description
ServiceParam	Group	Y	A group of subparameters that are parameters to the Siebel business service method (if any) invoked using the ServiceMethod parameter. You create each parameter you need in the form ServiceParam.*param_name*, then specify a parameter value. The parameter name and value are both passed to the service method. Order parameters in the sequence in which they are expected by the service method: ServiceParam.Param1="value1" ServiceParam.Param2="name" Param1 and Param2 are subparameters of the ServiceParam parameter. For more information, see "Using Business Services with Communications Server" on page 216.
WorkTrackingObj	Group	Y	Field name and values of a work-tracking object to create. You can retrieve the value of the tracking object from a command. For example, the following parameter defines a custom work-tracking object, called "ContactId", whose value is derived from the Id field of the Contact business component. WorkTrackingObj.ContactId="{Contact.Id}" For a selected work item, this data could then be accessed from a communications command, using the WorkTrackingObj command data parameter. In this example, ContactId becomes a custom work item attribute. Alternatively, if an event log includes a parameter definition like the following, then text you specify is displayed in the Work Items list in the communications toolbar. WorkTrackingObj.Description="your descriptive text goes here" For more information about using these and other attributes, see "Work Item Attributes" on page 190.

Event Log Examples

Table 19 shows an example of an event log for Siebel CTI Connect. This example shows how a new activity is created. The activity is linked with the current contact and its account because the current contact is the result of a screen pop.

Table 19. Event Log: LogIncomingCallContactFound

Parameter Name	Parameter Value
AfterWork.'ACD Call Duration'	{@WorkDuration}
AfterWork.'Planned Completion'	{@Now}
BusObj	Contact
BusComp	Action
Display	FALSE
LogField.Type	Call - Inbound
LogField.'Account Id'	{Contact.'Account Id'}
LogField.'Contact Id'	{Contact.Id}
LogField.Description	Inbound call
LogField.'Call Id'	{refId}
LogField.'Planned'	{@WorkStartTime}

Communications Simulator. Table 20 shows an example of an event log. A definition like this could be used when the inbound caller is not found in the database.

Table 20. Event Log: LogIncomingCallContactNotFound

Parameter Name	Parameter Value
BusObj	Contact
BusComp	Action
LogField.Type	Call - Inbound
LogField.Comment	{refId}
LogField.Description	Unknown Caller ({ANI})
AfterWork.'ACD Call Duration'	{@WorkDuration}

If you choose a View Call command after answering a simulated call, you go to the Activity Attachments view to see information about the current call. Notice that the description of the call starts with Unknown Caller, followed by the phone number from which the call was dialed.

You can change the LogField.Description parameter value to display other text here. For example, "Unknown Phone Number: {ANI}" displays the text "Unknown Phone Number:" followed by the phone number corresponding to the ANI event data field.

Commands

Commands in the communications configuration data define the appearance and availability of communications commands.

By using the DeviceCommand parameter in a command, you can invoke a command on the communications system or invoke a special command.

- Device commands that are executed by the communications driver. Such commands are specific to particular middleware vendors, or to how Siebel Systems supports communications toolbar interactivity. For more information about commands supported by Siebel-provided drivers, see "Siebel CTI Connect Commands" on page 346 and "Supporting Email Interactivity" on page 377.
- Device commands that are independent of communications drivers. For more information, see "Special Commands for Device Commands" on page 89.

Communications commands may instead invoke a business service method, a SmartScript, or a Siebel VB or Siebel eScript script. For commands, each of these mechanisms is mutually exclusive: invoke only one of these mechanisms in a given command. (Scripts and business service methods take precedence over device commands.) For more information:

- About invoking a business service method, see "Using Business Services with Communications Server" on page 216.
- About invoking a Siebel VB or Siebel eScript script, see Appendix B, "Communications Server Business Services."

Hierarchical Commands (Commands and Subcommands)

Communications commands can be defined in hierarchical fashion; you can define group commands that contain subcommands. These subcommands are invoked when the group command is invoked. Grouping or nesting commands using subcommands helps to define the communications user interface in the Siebel application.

For information about specifying subcommands, see "Defining Communications Commands" on page 70. For more information about how command groups are used in configuring the communications toolbar and menus, see Chapter 6, "Configuring User Interface Elements."

When you specify subcommands within group commands, note the following:

- Commands can have subcommands. A command containing subcommands is sometimes referred to as a group command.
- Do not define invalid recursive subcommand relationships, such as making a subcommand include its parent group command as a subcommand.

- The Subcommands list in the All Commands view specifies the order in which commands are executed. Alternatively, if the command parameter ExecuteAll is set to TRUE, then all subcommands can execute. As with any command, subcommands can execute only when they are enabled. Subcommands that are disabled are skipped and not executed.
- If a subcommand fails when executing, then the remaining subcommands are not executed. However, if the command parameter ExecUntilOK is set to TRUE, then the remaining subcommands are executed until one of them succeeds.
- Subcommands and their order are represented in .DEF files you export by using the command parameter SubCommand_*N*, where *N* is the order value for each subcommand.
- At runtime, commands that invoke device commands are enabled or disabled based in part on the status of the device command. For more information, see "SCCommandFlag" on page 410 and "CacheCommandInformation" on page 415.
- At runtime, commands that invoke business service methods rather than device commands are determined to be enabled or disabled by invoking from the applicable business service the method CanInvokeMethod("*method_name*"), where *method_name* is your business service method to be invoked. If TRUE is returned, the method can be invoked. If FALSE is returned, the method cannot be invoked and the command is disabled.
- Several command parameters are used to determine whether a group command or subcommand matches and can be executed. Such command parameters include AllViews, CmdChannelOnFocus, FilterSpec, OnEditControl, and View.
- Several command data parameters are also used to determine whether a group command or subcommand matches and can be executed. Such command data parameters include BusComp, BusObj, Filter, and RequiredField.
- Commands specifying subcommands cannot also specify a device command or business service method.
- If you require a command that executes both a device command and a business service method, define a group command where ExecuteAll is TRUE and where two subcommands are specified. Define one subcommand to execute a device command, such as a make-call command, and define the other subcommand to invoke your business service method. If you need command data parameters, create command data definitions and associate them with the subcommands.
- In some cases, a group command should not have command data associated with it. If a group command is at the top-level (is not itself a subcommand), or if a group command is also a subcommand and ExecuteAll is set to TRUE for its parent group command, then command data may be associated. However, in the latter case, if ExecuteAll is not set to TRUE for the parent group command, then do not associate command data with the child group command.
- The command parameter SelectApplet (and related parameters such as SelectBusObj, SelectBusComp, SelectView, and so on) can be used to display a dialog box that pertains to all subcommands in a group command, where ExecuteAll is TRUE. To configure this, you must define these parameters in the top-level command and not in the individual subcommands. The dialog box will apply to all subcommands.

Command Parameters

Table 21 on page 131 describes the available command parameters. A dash in the Macro column (—) indicates that macro expansion is not applicable for the parameter. The character Y indicates that macro expansion applies to the parameter. For more information about configuring the communications toolbar and Communications submenu commands, see Chapter 6, "Configuring User Interface Elements."

Table 21. Command Parameters

Parameter	Type	Macro	Description

AcceptReject	Boolean	—	Indicates whether the command is for prompting the agent to accept or reject a work item. Set to TRUE for such a command.
			
			Execution of the device command is subject to the Accept/Reject dialog box and subject to the agent's choices. If AcceptReject is TRUE, then this dialog box displays, with the title specified by the ARTitle parameter. The work item matches the definition of the ARWorkItemID parameter:
			
			■ If the agent accepts the work item, then the device command for this command can be executed.
			
			■ If the agent rejects the work item, then the command specified by the AROnRejectCmd parameter is executed.
			
			For example, the following command and command data definitions can be used for an accept/reject scenario for an email work item, where Siebel Universal Queuing is employed:
			
			`[Command:AcceptEmailGroup]`
			`Hidden="TRUE"`
			`DeviceCommand="AcceptEmailWork"`
			`CmdData="AcceptEmailGroup"`
			`AcceptReject="TRUE"`
			`ARTitle="Would You Like to Accept This Email Work Item?"`
			`ARWorkItemID="{$GetInboundWorkItemAttr (Email, WorkItemID)}"`
			`AROnRejectCmd="OnEmailWorkItemRejected"`
			`[CmdData:AcceptEmailGroup]`
			`Param.TrackingID="{$GetInboundWorkItemAttr (Email, DriverWorkTrackID)}"`
			`[Command:OnEmailWorkItemRejected]`
			`Hidden="TRUE"`
			`DeviceCommand="@UQRejectWorkItem"`
			`CmdData="OnEmailWorkItemRejected"`
			`[CmdData:OnEmailWorkItemRejected]`
			`Param.WorkItemID="{$GetInboundWorkItemAttr (Email, UQWorkItemID)}"`

AllViews	Boolean	—	Parameter that, when set to FALSE, disables the command on all views other than those specified using the View parameter. The default is TRUE.

Table 21. Command Parameters

Parameter	Type	Macro	Description
AROnRejectCmd	Char	—	Specifies the name of a command to be executed if the agent chooses to reject the work item. For example: AROnRejectCmd="OnEmailWorkItemRejected"
ARTitle	Char	—	Provides text to be presented to the agent when the agent is prompted to accept or reject the work item. For example, for a command used for the accept/reject case for email work items, the parameter could be defined as follows: ARTitle="Would You Like to Accept This Email Work Item?"
ARWorkItemID	Char	Y	Defines the ID of the work item to be accepted or rejected. For example, the parameter might be defined as follows: ARWorkItemID="{$InboundWorkItem(Email, WorkItemID)}"
CmdChannel OnFocus	Boolean	—	Specifies whether the command can only be enabled when the work item with the focus has the same channel as the device command for this command. In other words, if CmdChannelOnFocus is TRUE, the current work item is of a particular channel, and the device command is valid for a work item of this channel, then this command can be enabled. Otherwise, the command is disabled. For example, the command will be disabled if all of the following are true: ■ This parameter is set to TRUE. ■ The device command for this command is "HoldCall" (command is for Voice channel). ■ The work item that currently has the focus (that is, the current item displayed in the Work Items list in the communications toolbar) is for a channel other than Voice. An equivalent way to filter the command is to define the FilterSpec parameter as follows: FilterSpec=[@SelectedWorkItem:ChannelType]='Voice'"

Table 21. Command Parameters

Parameter	Type	Macro	Description
CmdData	Char	—	The command data definition name that describes parameter generation for this command to be passed to a device command (or to a business service method, for example). This parameter can be empty if the device command does not require any parameters from a command data parameters, or if no device command is specified. For more information, see "Command Data" on page 142. **NOTE:** Do not specify this parameter as a command parameter. Rather, specify a command data definition for the command directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
Description	Char	Y	Description string for the command. This string appears in the ToolTip area when the user points to a communications toolbar button for this command. For more information, see "About Communications Toolbar Configuration" on page 151 and "Configuring Communications Menu Commands" on page 160.
DeviceCommand	Char	—	The device command executed when this communications command is executed. Device commands that are executed by the communications driver are specific to particular communications systems. See the commands table for the relevant driver in Chapter 12, "Using Siebel CTI Connect." Device commands based on special commands are not executed by a communications driver. For more information, see "Special Commands for Device Commands" on page 89.

Table 21. Command Parameters

Parameter	Type	Macro	Description
ExecuteAll	Boolean	—	Default value is False. If you set ExecuteAll to True, do not set ExecUntilOK to True. When used within a group command (containing subcommands), specifies that all subcommands can be executed. For example, if autologin is in effect, and a default login command such as SignOnGroup is specified using the configuration parameter AutoLoginCmd, all of this group command's subcommands executes when the agent starts the Siebel session. In this example, the agent is logged in to all applicable communications systems, such as Siebel Universal Queuing or applicable ACD queues. See also "Hierarchical Commands (Commands and Subcommands)" on page 128 and "Configuring Communications Log In and Log Out" on page 201.
ExecUntilOK	Boolean	—	Default value is False. If you set ExecUntilOK to True, do not set ExecuteAll to True. When used within a group command (containing subcommands), specifies that each subcommand will execute in sequence until one of them succeeds. If a subcommand fails to execute, the next subcommand executes. See also "Hierarchical Commands (Commands and Subcommands)" on page 128.

Table 21. Command Parameters

Parameter	Type	Macro	Description
FilterSpec	Char	Y	Filter that supports simple or complex queries. Filter results are evaluated using a compound predicate that can include standard query operators to determine if the command matches. FilterSpec queries use standard comparison operators, including: = LIKE AND OR EXISTS > < >= <= Alternatively, FilterSpec can operate on the work item represented by the macro @SelectedWorkItem. For example, the following parameter definition matches voice work items: FilterSpec="[@SelectedWorkItem:ChannelType]='Voice'" See also the descriptions for the FilterSpec event handler parameter and the QuerySpec event response parameter.
Hidden	Boolean	—	Parameter that, when set to TRUE, hides the command from the Communications submenu. This command is typically set to TRUE for commands to appear only in the communications toolbar, or for commands that will not appear in the user interface. Set this parameter to FALSE (the default) or omit the parameter in order for the command to appear in the Communications submenu. Use the Description, HotKey, MenuPosition, and Title parameters to specify additional settings relevant to the command's appearance in the Communications submenu. For more information, see "About Communications Toolbar Configuration" on page 151 and "Configuring Communications Menu Commands" on page 160.

Table 21. Command Parameters

Parameter	Type	Macro	Description
HotKey	Char	—	Keyboard shortcut to assign to this command, for use in the Communications submenu. This can be a combination of CTRL, ALT, or SHIFT and one of the keyboard letter or function keys—or just a single letter or function key. For example, valid values for the HotKey parameter may include F12, CTRL+SHIFT+F, and so on. Keyboard shortcuts defined for communications commands do not use as input any data entered into the text entry field in the communications toolbar. **NOTE:** If you specify keyboard shortcuts, using the HotKey parameter, that conflict with keyboard shortcuts (accelerators) specified for commands in Siebel Tools, these shortcuts will take precedence over the shortcuts from Siebel Tools for the agent's session. For more information, see "Configuring Communications Menu Commands" on page 160.
HotKeyText	Char	—	Specifies locale-dependent text to represent the keyboard accelerator for the command's menu item. The keyboard accelerator itself is specified using the HotKey parameter. This locale-dependent text appears in the Communications submenu. If HotKeyText is not used, the text specified in HotKey is used instead. For more information, see "Configuring Communications Menu Commands" on page 160.

Table 21. Command Parameters

Parameter	Type	Macro	Description
IndicateActiveCmd	Boolean	—	Parameter that, when set to TRUE, specifies whether the toolbar button for this command should display the correct channel-specific icon for the device command for the active subcommand. This parameter is used in group commands containing subcommands for multiple channels—such as the AcceptWorkGroup command, for the Accept Work Item toolbar button. For example, if the active work item is a voice call, then the AcceptCallGroup subcommand is active, for which the device command is AnswerCall (for Siebel CTI Connect), or an equivalent command for your CTI middleware. The Accept Work Item button will display the phone-type icon in this case. The default is FALSE.
LocalMenu	Boolean	—	Parameter that, when set to TRUE, enables this command so it appears in the context-sensitive applet-level menu. This feature lets you, for example, define an applet-level menu item called Call Contact that can be invoked when working with contact records. The default is FALSE. For more information, see "Configuring Communications Menu Commands" on page 160.

Table 21. Command Parameters

Parameter	Type	Macro	Description
MenuPosition	Char	—	Parameter setting that determines the order in which the command will appear in the Communications submenu or in the applet-level menu. Commands appear in ascending order with respect to the MenuPosition values: those with lower MenuPosition values will appear above those with higher MenuPosition values. Values for the MenuPosition parameter can also specify additional menu command levels, within the Communications submenu of the Tools application-level menu. The applet-level menu commands, however, are all on a single level. For example, assume the SignOnGroupInMenu command (with Title parameter value "Log In") has a MenuPosition value of 20. This group command appears in the Communications submenu, below any items with lower number, and in turn has subcommands representing submenu items. The subcommands LoginToPBX and LoginToUQ have values of 20.1, 20.2, and 20.3—specifying that they appear as submenu items to SignOnGroupInMenu. LoginToPBX (with Title parameter value "Log In (Phone)") would be available to agents at the following path in the application-level menu: Tools > Communications > Log In > Log In (Phone) For more information, see "Configuring Communications Menu Commands" on page 160.
MultiActiveCmdIcon	Char	—	Specifies the name of the icon file to use for the toolbar button, such as the Accept Work Item button, if multiple work items have arrived, such as a phone call and an email message. For example, the parameter can be defined as follows: MultiActiveCmdIcon="misc_work.gif" For more information about toolbar button configuration, see "About Communications Toolbar Configuration" on page 151. For more information about icon files associated with communications drivers, see "Icon Files for Interactive Communications Drivers" on page 42.

Table 21. Command Parameters

Parameter	Type	Macro	Description
OnEditControl	Boolean	—	Parameter that, when set to TRUE, means that the command requires that the edit field in the communications toolbar has the focus and contains data. The default is FALSE.
Profile	Char	—	Name of the communications driver profile that supports the device command associated with this command. If a profile is associated with the command, then the command can only be sent to the communications driver for that profile. See also "Creating Commands" on page 72. **NOTE:** Do not specify this parameter as a command parameter. Rather, specify the profile for the command directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
Script	Char	—	Siebel VB or Siebel eScript script name to invoke (if specified). Parameters are passed using the ScriptParam parameter for the associated command data definition. For more information about using Siebel VB or Siebel eScript, see "Integrating with Siebel Scripting Languages" on page 222.
ServiceMethod	Char	—	The name of a Siebel business service and method to be called. The service and method are specified in the form *service.method*. Optionally, you can use the ServiceParam parameter for the associated command data definition to provide parameter names and values to pass to the method to be called. For more information, see "Using Business Services with Communications Server" on page 216.

Table 21. Command Parameters

Parameter	Type	Macro	Description
SubCommand_*N*	Multi	—	The name of a command that is specified as a subcommand for the current command. The commands may appear to an agent as one command because only one of them is active at any point, depending on the Siebel context and the specified Order values for the subcommands. Alternatively, all subcommands can execute, if the ExecuteAll parameter is set to TRUE. In a .DEF file (the only place this parameter appears), the suffix *N* represents the order value for the subcommand. For example, a call-transfer command could be either to transfer to an employee or to transfer to a service request owner, depending on the current business component. The commands that are invoked from the communications toolbar and the Communications submenu often invoke other commands, which are specified as subcommands. **NOTE:** Do not specify this parameter as a command parameter. Rather, specify subcommands for the command directly, using the Administration - Communications screen. This parameter is used in .DEF files that you export to or import from.
Title	Char	—	Name of the Communications submenu item associated with this command. If no Title parameter is specified, the name of the device command serves as the menu item name. For more information, see "Configuring Communications Menu Commands" on page 160.
View	Multi	—	Names of the views where this command is enabled if the AllViews parameter value is FALSE.

Command Example

Communications Simulator. Table 22 and Table 23 provide an example of a command showing how the Communications Simulator defines the hot key that simulates an incoming call from someone who is not in the database.

Table 22. Command: SimCallNotFound

Parameter Name	Parameter Value
DeviceCommand	SimulateCall
Hidden	TRUE
HotKey	CTRL+SHIFT+F11

Table 23. Command Data: SimCallNotFound

Parameter Name	Parameter Value
Param.ANI	6504775000

You might want to change something about this simulated incoming call. For instance, you might want the command to appear on the Communications submenu to provide Communications Simulator users with an alternative to pressing CTRL+SHIFT+F11.

For more information about the Communications Simulator, see "Enabling Session Communications and Simulation" on page 240.

To add a Simulate Call command to the Communications submenu

1. Start the Demo version of a Siebel application such as Siebel Call Center Demo.

   The demo versions of these applications use communications simulation and connect to the Sample Database.

2. Connect to the Sample Database as the Siebel administrator (SADMIN is both the login name and the password).
3. Navigate to Administration - Communications > All Commands.
4. In the Commands list, select the record for the SimCallNotFound command.
5. Change the command parameter Hidden from TRUE to FALSE.
6. Add a new command parameter called Title and give it the value Simulate Call.
7. Exit the Siebel client.

8 Restart the Siebel client.

A new menu item, Simulate Call, has been added to the Communications submenu.

If you do not specify a title, the menu item is based on the value for the DeviceCommand parameter; in this case, it is SimulateCall (with no space).

Command Data

Command data definitions in the communications configuration data define how communications command parameters are generated for use with a corresponding command.

For example, a command data definition can instruct one of the transfer commands to transfer the caller to the current service request owner, or to display a specified Siebel view, and extract the transfer destination from that view.

Command Data Parameters

Table 24 describes the parameters available in command data definitions. A dash (—) in the Macro column indicates that macro expansion is not applicable for the parameter. The character Y indicates that macro expansion applies to the parameter.

Table 24. Command Data Parameters

Parameter	Type	Macro	Description
AttachContext	Boolean	—	Parameter that, when set to TRUE, allows automatic passing of current screen-context data with the current work item. This is the easiest way of transferring screens between agents. AttachContext applies to voice work items only. The default is FALSE. The Send Screen Pop and Receive Screen Pop settings in the Communications options of the User Preferences screen override the setting of the AttachContext command parameter. For more information, see "Setting Communications User Preferences" on page 305. **NOTE:** The size of a Siebel screen bookmark is defined by the middleware and communications driver and may vary according to each vendor.
BusComp	Char	—	Business component name. If this parameter is empty, then the associated command is enabled on all views. If a business component name is specified, the associated command is enabled only on views that are based on the specified business component.

Table 24. Command Data Parameters

Parameter	Type	Macro	Description
BusObj	Char	—	Business object name. If this parameter is empty, then the associated command is enabled on all views. If a business object name is specified, the associated command is enabled only on views that are based on the specified business object.
OnField	Char	—	Field name that is required to be active. If this property is specified, then the related command is enabled only when this field is active (that is, when a cursor is in this field).
Param	Group	Y	Parameter names and values for the command. You can use this parameter to pass data to accompany the device command, whether for a communications driver or a special command. You create each parameter you need in the form Param.*param_name*, then specify a value for the parameter that you pass for the device command. Parameters specific to commands for particular middleware vendors, and possible values for these parameters, are listed in the command parameters table for the relevant driver in Chapter 12, "Using Siebel CTI Connect." The Param parameter can also be used to pass parameter values received from outside Communications Server, such as from a business service that invokes a Siebel communications command. For more information, see "About Using Business Services with Events and Commands" on page 218.
RequiredField	Group	Y	Field name/filter pair that will be used as a criterion to determine if this command will be active for this condition. RequiredField is always based on the currently selected row. For example, the following property will cause the specified command to be disabled if field A of the current (selected) row in the current (active) applet does not contain a value that ends with X2. RequiredField.A="*X2"

Table 24. Command Data Parameters

Parameter	Type	Macro	Description
ScriptParam	Group	Y	A group of subparameters that are parameters to the script method (if any) invoked using the Script parameter. You create each parameter you need in the form ScriptParam.*param_name*, then specify a parameter value that will be passed to the script. The parameter names themselves are not passed to the script. Order parameters in the sequence in which they are expected by the script: ScriptParam.Param1="value1" ScriptParam.Param2="name" Param1 and Param2 are subparameters of the ScriptParam parameter.
SelectApplet	Char	—	An applet from which an agent can make a selection, such as to select a recipient for a work item or to specify a reason code. You can specify the following applets: ■ Accept Reject Popup Applet ■ ACD Transfer Call Applet ■ Transfer Multiple LOV Popup Applet ■ UQ Transfer Route Popup Applet
SelectBusComp	Char	—	Business component for an applet from which an agent can make a selection.
SelectBusObj	Char	—	Business object specifying application elements from which an agent can make a selection.
SelectParam	Boolean	—	Parameter that, when set to TRUE, enables an applet in which the user can make a selection. For example, the applet can be used to select the person to whom to send, transfer, or conference a work item (such as to call or to transfer a call to) or used to specify a reason code. For example, for a MakeCall command, a dialog box can be displayed from which the agent can select an employee to call. The SelectBusObj, SelectBusComp, SelectApplet, SelectTitle, and SelectQuerySpec parameters determine the specific nature of the selection applet. The default is FALSE.

Table 24. Command Data Parameters

Parameter	Type	Macro	Description
SelectQuerySpec	Char	Y	Query specification for an applet, such as Transfer Multiple LOV Popup Applet, from which an agent can make a selection. For example, a command data definition can use SelectQuerySpec as follows: [CmdData:NotReadyWithPopup] SelectParam="TRUE" SelectTitle="Please select the reason for changing status to Not Ready" SelectApplet="Transfer Multiple LOV Popup Applet" SelectBusObj="List Of Values" SelectBusComp="List Of Values" SelectQuerySpec="[Type]='REASON_CODE' AND [Active]='Y'" Param.Reason="[Value]" In this example, SelectQuerySpec queries the List of Values business component to obtain the values to display in the Reason Codes list. If the applet allows you to select multiple records, as does Transfer Multiple LOV Popup Applet, then the selection values are concatenated as comma-separated values.
SelectTitle	Char	—	The title for the applet (dialog box) from which an agent can select a recipient for a work item. If this value is not specified, the applet will have the title that is assigned to the selection applet.

Table 24. Command Data Parameters

Parameter	Type	Macro	Description
ServiceParam	Group	Y	A group of subparameters that are parameters to the Siebel business service method (if any) invoked using the ServiceMethod parameter for the associated command. You create each parameter you need in the form ServiceParam.*param_name*, then specify a parameter value. The parameter name and value are both passed to the service method. Order parameters in the sequence in which they are expected by the service method: ServiceParam.Param1="value1" ServiceParam.Param2="name" Param1 and Param2 are subparameters of the ServiceParam parameter. For more information, see "Using Business Services with Communications Server" on page 216.

Table 24. Command Data Parameters

Parameter	Type	Macro	Description
WorkTrackingObj	Group	Y	Field name and values of a work-tracking object previously specified in an event log definition. The WorkTrackingObj command data parameter stores or updates this object for the currently selected work item. For example, the following command and command data definitions update the customer dashboard with a contact ID: [Command:UpdateDashboardFromContact] Hidden="TRUE" ServiceMethod="Persistent Customer Dashboard.Update Dashboard from CTI" CmdData="UpdateDashboardFromContact" [CmdData:UpdateDashboardFromContact] BusComp="Contact" ServiceParam.Field="Id" ServiceParam.Value="{Id}" WorkTrackingObj.ContactId="{Id}" The WorkTrackingObj parameter updates the attribute value of "ContactId" for the currently selected work item. In this example, ContactId is a custom work item attribute. You can also use WorkTrackingObj to update the Siebel bookmark. For example, using the following line in a command data definition for a command to resume a suspended work item (such as to remove a call from hold) prevents a screen pop from occurring when an agent resumes a work item. If this parameter is not defined as shown, a screen pop may display the view the agent was on when suspending the work item. WorkTrackingObj.ViewBookmark="" For more information, see "Work Item Attributes" on page 190.

Command Data Example

Table 25 shows an example of a command data definition for a Siebel CTI Connect configuration.

Table 25. Command Data: ConferenceTransferToPopupEmployee

Parameter Name	Parameter Value
BusObj	Service Request
BusComp	Service Request
AttachContext	TRUE
SelectParam	TRUE
SelectBusObj	Employee
SelectBusComp	Employee
SelectApplet	ACD Transfer Call Applet
SelectTitle	SR Conference To Agent Group
Param.PhoneNumber	[Phone #:6-10]
Param.CallNotifyText	Conference transfer from {@UserName}...

The command that uses this command data definition is enabled only on Service Request applets. First, the command opens the applet ACD Transfer Call Applet, for selecting an employee to whom to transfer a call (this applet is also used for placing and conferencing calls). Next, it dials the extension of the selected employee by retrieving the sixth through tenth digits of the phone number, for example, a five-digit extension that is part of a ten-digit phone field.

Communications Simulator. In the command and associated command data shown in Table 26 and Table 27, the phone number for the parameter Param.ANI is the same as the phone number specified in the Description field for the activity created for a simulated incoming call.

Table 26. Command: SimCallNotFound

Parameter Name	Parameter Value
DeviceCommand	SimulateCall
Hidden	FALSE
HotKey	CTRL+SHIFT+F11

Table 27. Command Data: SimCallNotFound

Parameter Name	Parameter Value
Param.ANI	6504775000
Param.Testing	My_Value_Here

6 Configuring User Interface Elements

This chapter provides information about configuring user interface elements for communications features. It includes the following topics:

About Communications Toolbar Configuration

This section explains how the communications toolbar buttons relate to commands defined for the communications configuration and provides information about modifying the communications toolbar for your agents.

For information about enabling communications activities through the communications toolbar, see "Enabling Session Communications and Simulation" on page 240.

For information about using the communications toolbar, see "Using the Communications Toolbar" on page 314.

The communications toolbar is defined and configured, in part, within Siebel Tools, like other toolbars in the Siebel applications. Within Siebel Tools, the Communication toolbar object definition contains, or is linked to, a series of other object definitions, including toolbar items, commands, and bitmaps.

These object definitions, combined with associated business service methods, communications commands, and bitmap image files, determine the functioning and appearance of the communications toolbar for your agents.

NOTE: The available communications toolbar functions are also determined by functionality supported by applicable communications drivers and external communications systems, such as CTI middleware.

The communications toolbar is a Java applet, and connects to the Web server using HTTP. Toolbar functions are determined by communications events and commands, which interact with communications drivers loaded by the Communications Session Manager component. Communications between the driver and applicable middleware is subject to the implementation of the driver.

For more information about configuring object definitions for toolbars, toolbar items, commands, and bitmaps, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

Communications Toolbar Items, Commands, and Methods

Table 28 displays information about elements associated with the toolbar items for the Communication toolbar object definition. From left to right, the table displays each toolbar item name, the associated Siebel Tools command name, and the business service method associated with the command (where applicable). The methods are those of the Communications Client business service.

The Default column indicates whether the toolbar button appears in the toolbar by default. An asterisk (*) indicates that the button is part of a submenu.

Each business service method name also corresponds to the name of a communications command. Communications configurations provided by Siebel Systems include commands with these names. For more information, see "Commands" on page 128.

Table 28. Communications Toolbar Items, Commands, and Methods

Toolbar Item	Tools Command (Display Name)	Business Service Method / Communications Command	Default
Accept Email	Accept Incoming Email	AcceptEmailGroup	Yes*
Accept Misc Work	Accept Miscellaneous Work Item	AcceptMiscWorkGroup	*
Accept Phone	Answer Phone Call	AnswerCallGroup	Yes*
Accept Work	Accept Work Item	AcceptWorkGroup	Yes
Blind Transfer	Blind Transfer Work Item	BlindTransferGroup	Yes
Consultive Conference	Consultative Conference Work Item	ConferenceTransferGroup	Yes
Consultative Transfer	Consultative Transfer Work Item	ConsultativeTransferGroup	Yes
Forward Work	Forward Work Item	ForwardWorkGroup	Yes
Hold Work	Hold Work Item	SuspendWorkGroup	Yes

Table 28. Communications Toolbar Items, Commands, and Methods

Toolbar Item	Tools Command (Display Name)	Business Service Method / Communications Command	Default
Initiate Email	Initiate Email	SendEmailGroup	Yes*
Initiate Fax	Initiate Fax	SendFaxGroup	Yes*
Initiate Phone	Initiate Phone Call	MakeCallGroup	Yes*
Initiate SMS	Initiate SMS	SendSMSGroup	Yes*
Initiate Work	Initiate Work Item	InitiateWorkGroup	Yes
InQueue Time	InQueue Time	N/A	Yes
Media Indicator	Media Indicator	N/A	Yes
Not Ready	Not Ready State	NotReadyGroup	Yes
Not Ready Email	Not Ready for Email	NotReadyForEmailGroup	Yes*
Not Ready Phone	Not Ready for Phone	NotReadyForPhoneGroup	Yes*
Release Work	Release Work Item	ReleaseWorkGroup	Yes
Resume Work	Resume Work Item	ResumeWorkGroup	Yes
Retrieve Work	Retrieve Work Item	RetrieveWorkGroup	Yes
SignOff	Communication Sign Off	SignOffGroup	Yes
SignOn	Communication Sign On	SignOnGroup	Yes
Work Item List	Work Item List	WorkItemList	Yes
Working Time	N/A	N/A	Yes

Communications Toolbar Commands and Bitmaps

Table 29 on page 154 displays the names of the bitmap object definitions that are associated with the Siebel Tools commands for the communications toolbar items.

From left to right, the table displays the same Siebel Tools command names shown in Table 28 on page 152, the name of the bitmap object definition specified as the HTML Bitmap, and the name of the bitmap object specified as the HTML Disabled Bitmap.

HTML Bitmap is for controls in an enabled state, while HTML Disabled Bitmap is for controls in a disabled state.

The applicable image file names can be found in Siebel Tools, by viewing the bitmap object definitions for the bitmap category HTML Command Icons.

The source image files are installed in the following locations:

- `webmaster\images\`*language_code*, in the Siebel Server installation directory

- public*language_code*\images, in the Siebel Developer Web Client installation directory

where *language_code* represents the language code for the installed software, such as ENU for U.S. English.

NOTE: New or updated files on the Siebel Server are automatically populated to the Web server whenever the Web server (where the Siebel Web Server Extension is installed) is restarted. Files can also be manually updated from the Siebel Server without restarting the Web server. For more information, see *Security Guide for Siebel Business Applications*.

Table 29. Communications Toolbar Commands and Bitmaps

Tools Command (Display Name)	HTML Bitmap	HTML Disabled Bitmap
Accept Incoming Email	Comm Email	Comm Email Disabled
Accept Miscellaneous Work Item	Comm Misc Work	Comm Misc Work Disabled
Accept Work Item	Comm Accept Work	Comm Accept Work Disabled
Answer Phone Call	Comm Phone	Comm Phone Disabled
Blind Transfer Work Item	Comm Blind Transfer	Comm Blind Transfer Disabled
Communication Sign On	Comm SignOn	Comm SignOn Disabled
Communication Sign Off	Comm SignOff	Comm SignOff Disabled
Consultative Conference Work Item	Comm Consultive Conference	Comm Consultive Conference Dis
Consultative Transfer Work Item	Comm Consultive Transfer	Comm Consultive Transfer Disab
Forward Work Item	Comm Forward Work	Comm Forward Work Disabled
Hold Work Item	Comm Hold Work	Comm Hold Work Disabled
Initiate Email	Comm Email	Comm Email Disabled
Initiate Fax	Comm Fax	Comm Fax Disabled
Initiate Phone Call	Comm Phone	Comm Phone Disabled
Initiate SMS	Comm SMS	Comm SMS Disabled
Initiate Work Item	Comm Initiate Work	Comm Initiate Work Disabled
InQueue Time	Comm InQueue Time	Comm InQueue Time
Media Indicator	Comm Phone	Comm Phone
Not Ready for Email	Comm Email	Comm Email Disabled
Not Ready for Phone	Comm Phone	Comm Phone Disabled
Not Ready State	Comm Not Ready	Comm Not Ready Disabled
Release Work Item	Comm Release Work	Comm Release Work Disabled

Table 29. Communications Toolbar Commands and Bitmaps

Tools Command (Display Name)	HTML Bitmap	HTML Disabled Bitmap
Resume Work Item	Comm Resume Work	Comm Resume Work Disabled
Retrieve Work Item	Comm Retrieve Work	Comm Retrieve Work Disabled
Work Item List	N/A	N/A

Modifying the Communications Toolbar

In implementing Siebel Communications Server, you can modify communications toolbar functionality in several ways. Several types of modifications that may apply to your implementation are described in this section.

For more information about configuring the communications toolbar in Siebel Tools, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

NOTE: Generally, it is best not to make changes in Siebel Tools unless it is strictly necessary or it is done with other application development work.

Modifying the Function of an Existing Toolbar Button

You can modify the function of an existing communications toolbar button in several different ways. You can:

- Change which communications command an existing communications toolbar button invokes
- Change which device command an existing communications toolbar button invokes
- Change which business service method an existing communications toolbar button invokes

Modifying Communications Command for Existing Toolbar Button

If you have created a command in the communications configuration and you want a communications toolbar button to invoke this communications command instead of an existing command, rename your commands so that the new command uses the name of the command that is currently associated with the toolbar button.

Renaming commands in the communications configuration has less impact on an upgrade effort than modifying object definitions in Siebel Tools.

Renaming commands in the communications configuration can be used whether you are only implementing a custom command or you are both implementing a custom command and invoking a device command supported by a custom communications driver, as described in "Modifying Device Command for Existing Toolbar Button."

For more information about communications commands, see "Commands" on page 128.

Modifying Device Command for Existing Toolbar Button

If you are using a custom communications driver, created using the Adaptive Communications API, in order to extend or replace the functionality of a driver provided by Siebel Systems, then you can reuse the existing communications toolbar configuration, and you can reuse existing commands from the configuration data provided by Siebel Systems. Of course, you can also create new communications commands, as described earlier in this section.

If your communications driver supports different device command names than do the drivers provided by Siebel Systems, specify your driver's device commands in your communications commands. Alternatively, for commands that do not specify a device command and do not execute something else such as a business service method or script, the application attempts to execute a device command with the same name as the communications command. For more information about device commands, see "Commands" on page 128. For more information about Adaptive Communications, see Appendix A, "Developing a Communications Driver."

Modifying Business Service Method for Existing Toolbar Button

If you have created a custom business service, and you want an existing communications toolbar button to invoke a method for this service, then you must use Siebel Tools to modify the command object definition that is associated with the toolbar item, to specify the new business service and method.

This approach is of limited practical value, however, because you can customize the function of a button without using Siebel Tools.

For more information about how business service methods are specified for the toolbar buttons, see "Communications Toolbar Items, Commands, and Methods" on page 152.

Modifying the Appearance of an Existing Toolbar Button

You can modify the appearance of an existing communications toolbar button in several different ways. You can:

- Change the image file or bitmap object definition for an existing toolbar button
- Change the icon file for an interactive communications driver

Changing Image File or Bitmap Object Definition for Existing Toolbar Button

If you want to customize the communications toolbar by using your own graphical elements in place of existing elements, you can do so by using your own graphics files in place of existing files.

You can change the image file specified for the HTML Bitmap and the image file specified for the HTML Disabled Bitmap. If you want to change the graphics used to represent a given channel, for example, you can simply identify the files used to represent the channel and change them to use your new images. All toolbar buttons that use the same bitmap object definitions will now use the new image files.

Suppose, however, that you want to change the appearance of a particular button, but you do not want to change the appearance of all buttons that may display the same image file. In this case, you can create a new bitmap object definition in Siebel Tools, associate it with your new image file, and associate the toolbar item object definition with the new bitmap object definition.

For more information about how bitmaps are specified for the toolbar buttons, and about the locations of the image files, see "Communications Toolbar Commands and Bitmaps" on page 153.

Changing the Icon File for Interactive Communications Driver

You can change the icon file for an interactive communications driver. The icon file represents communications work items of a particular channel, and is used for the Channel Type Indicator and to support blinking toolbar buttons.

Some communications toolbar buttons blink, based on event occurrence. For example, when a new work item arrives, the button for Accept Work Item blinks until the agent accepts the work item. When a work item has been paused, the button for Resume Work Item blinks until the agent resumes the work item.

For any buttons executing methods of the Communications Client business service, as do the default buttons on the communications toolbar, the blinking behavior is controlled by the applicable communications driver and cannot be configured. If a button is configured to execute a method of another business service, behavior is controlled by that service.

The names of the icon files are specified in the Icon File field in the Communications Drivers and Profiles view in the Administration - Communications screen. The files are in the same location as those for the bitmap object definitions.

Although different file names are used, the image content of the driver icon files is identical to those of some of the bitmaps used for the same communications channel. These bitmaps, and the locations of the associated image files are described in "Communications Toolbar Commands and Bitmaps" on page 153.

For more information about interactive drivers and icon files, see "Interactive Drivers" on page 41.

Moving, Adding, or Removing a Toolbar Button

You can customize the communications toolbar to add a new toolbar button or to remove an existing button. You can:

- Move a toolbar button or other control on the communications toolbar

  You can move a toolbar control, in relation to other controls, by modifying the value of the Position field for the applicable toolbar item in Siebel Tools.

- Remove a toolbar button or other control from the communications toolbar

  You can prevent a control from appearing in the communications toolbar by setting the Inactive flag to TRUE for the applicable toolbar item in Siebel Tools.

- Add a new button to the communications toolbar

  If you add a new toolbar button to the communications toolbar, you must also add, or reuse, all associated object definitions in Siebel Tools and image files. The scope of what you need to do depends on the purpose of the new button.

Communications Toolbar Buttons and Commands

This section provides an overview of the following:

- How communications commands are invoked from the communications toolbar
- About command parameters that affect toolbar commands
- About how group commands function to support the communications toolbar, especially in a multichannel environment

How Communications Toolbar Buttons Work

When an agent clicks a communications toolbar button, the Siebel client attempts to execute a command, corresponding to the method name, from the communications configuration.

If no matching command is found, the communications driver (a profile for which is associated with the configuration) that supports a device command with the same name as the method will execute that device command.

If a command invokes a subcommand, business service method, or script, then it need not execute a device command. All other commands generally execute a specified device command.

A communications command executing a device command can be explicitly associated with a profile for the driver that supports that device command. This can help to avoid possible collisions where the same device command is supported by multiple drivers, and can also help the administrator to keep track of the intended functions of commands in a complex, multichannel environment.

For more information about the DeviceCommand parameter, see "Command Parameters" on page 130. For available device commands that may apply for your deployment, see:

- "Siebel CTI Connect Commands" on page 346
- "User-interactive Email Driver Parameters" on page 378
- "Special Commands for Device Commands" on page 89

Command Parameters Affecting Communications Toolbar Buttons

The presence and visual appearance of items in the communications toolbar are determined in part by a set of related command parameters defined for the communications commands that are accessible through the toolbar:

- **Description.** Provides the text to be displayed as ToolTip text for the toolbar button for the command. Many references to toolbar buttons by name are based on the ToolTip text specified with this parameter. For more information, see "Communications Toolbar ToolTip Text" on page 159.
- **Hidden.** Although this parameter has no direct effect on the communications toolbar, it is typically set to TRUE for a command that is to appear in the communications toolbar (and that is not to appear in the Communications submenu).
- **LocalMenu.** Although this parameter has no direct effect on the communications toolbar, it is typically set to FALSE, or omitted from the command, for a command that is to appear in the communications toolbar (and that is not to appear in the applet-level menu).

For more information about these parameters, see "Command Parameters Affecting Communications Menu Items" on page 161 and "Command Parameters" on page 130.

Communications Group Commands in Toolbar

In the communications configurations provided by Siebel Systems, commands whose names end in "Group" are commands that may contain several subcommands. Most of the communications toolbar buttons invoke group commands. A group command can invoke another group command.

When a group command is executed, such as by clicking the associated toolbar button, one of its subcommands is typically invoked, based on the current context. Alternatively, a group command can be configured to execute all subcommands, where this is appropriate.

The order specified for each subcommand determines the sequence in which the subcommands are checked to find one that matches the context.

For more information about group commands and subcommands, see "Hierarchical Commands (Commands and Subcommands)" on page 128.

Communications Toolbar ToolTip Text

The ToolTips displayed when the agent points to a communications toolbar button are usually derived from the command in the communications configuration. They may also be obtained from the communications driver, or from the associated command object definition in Siebel Tools.

The ToolTip text is derived in this fashion:

- For the communications command (which may be a subcommand of another command) that the toolbar button represents, the value of the command parameter Description is used as the ToolTip text. Obtaining text from the context-appropriate subcommand allows the ToolTip text itself to vary by context. For the MakeCallToContact command, for example, "Make Call to Contact" is the ToolTip text.
- If the command parameter Description is not used for the communications command that the toolbar button represents, then any ToolTip text defined in the communications driver for the associated device command is used. Such text may also be context-sensitive, depending on the driver implementation.

  NOTE: In order to update the description of a device command, a driver must use the client handle method CacheCommandInformation. In turn, this description will be used for the ToolTip text.
- If neither of the previous two sources provides ToolTip text, then any ToolTip text is used that is defined for the command object definition in Siebel Tools. Such text cannot be context-sensitive, because only a single string is defined for the toolbar button.

Communications drivers and configurations provided by Siebel Systems provide ToolTip text. If you use a driver provided by Siebel Systems, ToolTip text derived from Siebel Tools is not used.

Configuring Communications Menu Commands

This section contains following information:

- Describes the Communications submenu and communications commands in the applet-level menu
- Explains how items in these menus relate to commands defined for the communications configuration
- Provides instructions and guidelines for modifying communications menu commands

For information about using communications menu commands and the Send commands, see "Using Communications Menu Commands" on page 321 and "Sending Email, Fax, and Page Messages" on page 324.

For more information about outbound communication requests, see Chapter 10, "Defining Outbound Communication Requests."

Communications Submenu and Applet-Level Menu

When communications are enabled for a user, the Communications submenu is displayed in the Tools application-level menu. The Communications submenu is configured using commands defined in the communications configuration.

The name and position of "Communications" as an item in the Tools application-level menu is determined in Siebel Tools by the Tools - Communications menu item definition.

Like the communications toolbar, the exact content of the Communications submenu varies according to the communications configuration you are using. All the sample communications configurations provided by Siebel Systems contain definitions to configure the Communications submenu.

The applet-level menu, which is accessed at the top of each applet, below the applet name, is generally available in the Siebel client. When session communications are enabled, communications-related items are added to it, according to the commands defined for the communications configuration.

Command Parameters Affecting Communications Menu Items

The presence and visual appearance of items in the Communications submenu and applet-level menu are determined by a set of related command parameters defined for the communications commands that are to appear in the affected menus. For more information, see "Command Parameters" on page 130.

The following list describes the command parameters:

- **Hidden.** Set to FALSE, or omitted from the command, for a command to appear in the Communications submenu. Set to TRUE for a command that is intended to appear in the applet-level menu or the communications toolbar, or that is not to appear directly in the user interface.
- **HotKey.** Specifies the key combination that serves as a keyboard accelerator for the command's menu item.

  Keyboard shortcuts defined for communications commands do not use as input any data entered into the text entry field in the communications toolbar.

  NOTE: If you specify keyboard shortcuts, using the HotKey parameter, that conflict with keyboard shortcuts specified in Siebel Tools, these shortcuts will take precedence over the shortcuts from Siebel Tools for the agent's session.
- **HotKeyText.** Specifies locale-dependent text to represent the keyboard accelerator for the command's menu item. The keyboard accelerator itself is specified using the HotKey parameter. This locale-dependent text appears in the Communications submenu.
- **LocalMenu.** Set to TRUE for a command to appear in the applet-level menu. Set to FALSE, or omitted from the command, for all other commands.
- **MenuPosition.** Specifies the position of the command's submenu item, relative to the other submenu items in the Communications submenu, or the position of the command's menu item relative to other menu items in the applet-level menu. This parameter supports multiple menu levels.
- **Title.** Provides the text of the command's menu item in the Communications submenu or in the applet-level menu.

Communications Group Commands in Menus

In the communications configurations provided by Siebel Systems, the commands whose names end in "GroupInMenu" or "GroupInLocalMenu" may specify several subcommands, one of which is invoked when the menu item is chosen, based on the current context.

The order specified for each subcommand determines the sequence in which the subcommands are checked to find one that matches the context.

The "GroupInMenu" commands are for the Communications submenu, and the "GroupInLocalMenu" commands are for the applet-level menu.

For example, MakeCallGroupInLocalMenu has several subcommands defined, each of which specifies another command, such as MakeCallToAccount, MakeCallToContact, and so on. When the agent chooses Make Call in the applet-level menu, one of these subcommands is invoked, according to the current context.

If the agent is viewing a contact record, for example, Contact is the current business component. Because this business component is specified in the command data defined for MakeCallToContact, a match is found in this command.

For more information about group commands and subcommands, see "Hierarchical Commands (Commands and Subcommands)" on page 128.

Communications Menu Items and Device Commands

When an agent chooses a communications command from the Communications submenu or the applet-level menu, the corresponding command is executed from the communications configuration. The communications driver (a profile for which is associated with the configuration) that supports the device command specified for the communications command will execute that device command.

If a command invokes a subcommand, business service method, or script, then it need not execute a device command. All other commands generally execute a specified device command.

A communications command executing a device command can be associated with a profile for the driver that supports that device command. This can help to avoid possible collisions where the same device command is supported by multiple drivers, and can also help the administrator to keep track of the intended functions of commands in a complex, multichannel environment.

For more information about the DeviceCommand parameter, see "Command Parameters" on page 130. For available device commands, see the commands table for the relevant driver in Chapter 12, "Using Siebel CTI Connect," or see "Special Commands for Device Commands" on page 89.

Configuring the Send Commands in the File Menu

This section describes configuring commands in the File menu for sending email, fax, or page messages. It also describes how the Send commands in the File application-level menu are configured. These commands are used for sending email, fax, or page messages.

End User Requirements for Using Send Commands

End users must have communications profiles available to choose from, if they are sending email or fax messages. End users must also have communications templates defined. For more information about end user operations, see:

- "Sending Email, Fax, and Page Messages" on page 324
- "Creating Communications Profiles for Personal Use" on page 323
- Chapter 9, "Configuring Communications Templates"

For more information about drivers and profiles, see Chapter 3, "Configuring Communications Drivers and Profiles."

Send Menu Commands Defined in Siebel Tools

The name and position of the Send commands in the File menu are determined by the properties of their corresponding menu item definitions in Siebel Tools.

These menu items invoke Siebel Tools commands that in turn invoke the SendCommunication method of the Communications Client business service—each with a different setting for the method argument CommType:

- For the Send Email command (File - Send EMail menu item), CommType is set to Email.
- For the Send Fax command (File - Send Fax menu item), CommType is set to Fax.
- For the Send Page command (File - Send Page menu item), CommType is set to Page.

For the Send Email and Send Fax commands, the SendCommunication method in turn invokes the business service methods of the Outbound Communications Manager.

The Send Page command offers similar functionality to the other Send commands, but uses the Page Manager server component, rather than Communications Server. For more information, see "Server Components for Communications Server" on page 237.

Send Commands in Communications Toolbar

The communications configurations provided by Siebel Systems include commands for choosing communications toolbar buttons for the business service methods Send Email, Send Fax, and Send Page commands. These commands invoke the same business service methods mentioned above. They do not, however, affect the Send commands in the File application-level menu.

Configuring Communications List of Values Types

Records defined for various List of Values (LOV) types determine what values can be selected in certain fields (drop-down lists) in the Administration - Communications screen.

LOV records of the same type and, where applicable, with the same Parent LIC (Language-Independent Code) value are listed together in particular drop-down lists. An LOV record must also be set to Active in order to appear in the user interface.

Typically, LOV records provided by Siebel Systems will serve your needs without alteration. However, depending on your implementation, you may need to include custom content in a particular LOV. To include custom content, you add an LOV record, or modify an existing LOV record to change its display value.

In particular, if you are implementing a custom communications driver, you must create records for several List of Values types.

CAUTION: For an existing LOV record, it is not recommended to modify values other than for the Display Value field. It is also not recommended to delete LOV records provided by Siebel Systems. If you are *certain* that an LOV record is not needed in any context, you can uncheck the Active flag in order to hide it in the user interface. However, note that some LOV records are required for internal operations not directly related to the user interface. For example, for inbound and outbound email, existing LOV records of type TODO_TYPE, with display values Email - Inbound, or Email - Outbound, must exist in unmodified form, and must be active.

Some of the List of Values types for which you may provide your own values are described in this section. Your deployment may require you to add or modify records for additional List of Values types not described here.

To add or modify List of Values records, you use the Administration - Data screen and the List of Values Explorer and List of Values views.

NOTE: For instructions for adding or modifying List of Values records, see *Applications Administration Guide*. Follow all documented guidelines and restrictions.

For information about configuring records for the List of Values type for recipient groups for outbound communication requests, see "Configuring Recipient Groups for Requests and Advanced Templates" on page 167.

List of Values Types for Channel Type

This section identifies List of Values types for specifying the channel type for communications drivers, templates, and outbound communication requests.

List of Values Type for Channel for Communications Drivers and Siebel Universal Queuing

If the available communications channel types do not meet your needs for your communications drivers and for Siebel Universal Queuing, and you want to add or modify channels, you must add or modify records to include the values you require. The applicable List of Values type is COMM_MEDIA_TYPE. The Parent LIC value depends on the context:

- For values to be displayed *only* in the Channel Type field in the Communications Drivers applet (in the Communications Drivers and Profiles view in the Administration - Communications screen), Parent LIC is set to COMM.
- For values to be displayed *only* in the Channel Type field in the Channels applet (in the Channels view in the Administration - UQ screen), Parent LIC is set to UQ.

- For values to be displayed *both* in the Channel Type field in the Communications Drivers applet and in the Channel Type field in the Channels applet (in the Channels view in the Administration - UQ screen), Parent LIC is set to COMMON.

For more information about defining drivers, see "Configuring Communications Drivers and Profiles" on page 44.

For more information about Siebel Universal Queuing, see *Siebel Universal Queuing Administration Guide*.

List of Values Type for Channel for Templates and Requests

If the available channel types do not meet your needs for your communications templates and outbound communication requests, you must add or modify records to include the values you require. The applicable List of Values type is OFFER_MEDIA. The applicable Parent LIC value is Package.

Such LOV records are used for:

- The Channel Type field in the All Templates view (Administration - Communications screen) and the My Templates view (Communications screen)
- The Default Preferences check box in the All Outbound Requests and Outbound Request Overview views (Administration - Communications screen) and the My Outbound Requests and My Outbound Request Overview views (Communications screen)

For more information about defining templates, see Chapter 9, "Configuring Communications Templates." For more information about defining communication requests, see Chapter 10, "Defining Outbound Communication Requests."

List of Values Type for ACD Queues

If you are supporting the voice channel (using Siebel CTI), and supporting ACD queues for your agents, then you may require particular values to appear in the ACD Queue field in the ACD Queues applet. The ACD Queues applet appears in the Agent General Profile view, in the Administration - Communications screen.

To specify your custom values, add or modify records for the List of Values type named CTI_ACD_QUEUES. For more information about specifying agents and ACD queues, see "Specifying Agents" on page 59.

List of Values Type for Reason Code

If you are supporting reason codes for your call or contact center agents to specify when they are not ready to receive communications work items, then you may require particular values to appear in the applet Transfer Multiple LOV Popup Applet. This applet appears when agents specify the Not Ready state for any or all supported communications channels. By default, this applet is multiselect enabled.

To specify your custom values, add or modify records for the List of Values type named REASON_CODE.

For more information about how agents specify the Not Ready state, see "Using the Communications Toolbar" on page 314.

List of Values Types for Event Parameters

This section identifies List of Values types for parameters for event handlers, event responses, and event logs.

NOTE: You need to add LOV records for these types only if you are using a custom communications driver.

For more information about specifying events, see "Defining Communications Events" on page 65.

List of Values Type for Event Handler Parameters

If you use a custom communications driver, you require particular values to appear in the Name field (drop-down list) in the Event Handler Parameters applet. This applet appears in the All Event Handlers view, in the Administration - Communications screen.

To specify your custom values, add or modify records for the List of Values type named COMM_EVTHNDLR_PARAM.

List of Values Type for Event Response Parameters

If you use a custom communications driver, you require particular values to appear in the Name field (drop-down list) in the Event Response Parameters applet. This applet appears in the All Event Responses view, in the Administration - Communications screen.

To specify your custom values, add or modify records for the List of Values type named COMM_EVTRESP_PARAM.

List of Values Type for Event Log Parameters

If you use a custom communications driver, you require particular values to appear in the Name field (drop-down list) in the Event Log Parameters applet. This applet appears in the All Event Logs view, in the Administration - Communications screen.

To specify your custom values, add or modify records for the List of Values type named COMM_EVTLOG_PARAM.

List of Values Types for Command Parameters

This section identifies List of Values types that parameters for commands and command data use.

NOTE: You need to add LOV records for these types only if you are using a custom communications driver.

For more information about specifying commands, see "Defining Communications Commands" on page 70.

Command Parameters

If you use a custom communications driver, you require particular values to appear in the Name field (drop-down list) in the Command Parameters applet. This applet appears in the All Commands view, in the Administration - Communications screen.

To specify your custom values, add or modify records for the List of Values type named COMM_CMD_PARAM.

Command Data Parameters

If you use a custom communications driver, you require particular values to appear in the Name field (drop-down list) in the Command Data Parameters applet. This applet appears in the All Command Data view, in the Administration - Communications screen.

To specify your custom values, add or modify records for the List of Values type named COMM_CMDDATA_PARAM.

Configuring Recipient Groups for Requests and Advanced Templates

This section describes configuring recipient groups for outbound communication requests and advanced communications templates. A *recipient group* is a way to specify a group of people you want to send communications to. Generally, you do not need to configure recipient groups unless existing recipient groups do not meet your needs:

- For outbound communication requests, recipient groups determine the actual recipients for communications. For example, for the Opportunity Contacts recipient group, recipients come from the contacts associated with one or more opportunities.
- For communications templates, which are used by requests and by other communications features, recipient groups determine which Siebel objects the template is associated with:
  - For advanced templates used for outbound communication requests, substitution fields for templates come from these objects. For example, for the Opportunity Contacts recipient group mentioned above, substitution fields come from the Opportunity and Contact business components.
  - (For simple templates used for the Send commands in the File menu and for email replies in Siebel Email Response, substitution fields come from a specified business object and no configuration is necessary other than for the template itself. See also "Configuring Recipients for Send Commands" on page 176.)

For a detailed description of recipient groups for communication requests and advanced templates, see "About Recipient Groups" on page 168.

Outbound communication requests use advanced templates in sending email or fax messages in an automated fashion. Communication requests can support high-volume processing and output.

Outbound communication requests can be created and submitted directly by a user, or can be created and submitted automatically by Siebel Workflow or other modules, by invoking methods of the Outbound Communications Manager business service.

For more information about working with recipient groups when creating advanced templates and requests, see Chapter 9, "Configuring Communications Templates," and Chapter 10, "Defining Outbound Communication Requests."

For more information about Communications Server business services, see "Using Business Services with Communications Server" on page 216 and Appendix B, "Communications Server Business Services."

For more information about Siebel Workflow, see *Siebel Business Process Designer Administration Guide*.

About Recipient Groups

This section describes concepts that will help you in either using or configuring recipient groups used with advanced templates and outbound communication requests.

For more information about working with business objects, business components, links, and other elements mentioned here, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

Each recipient group specifies a source business object, called the *recipient source*—such as Opportunity, Account, Action, Internal Division, and so on. The recipient group also specifies a recipient group business component, which must be a direct child of the primary business component for the recipient source business object.

The actual recipients for a communication request come from the recipient group business component, such as Contact, Employee, and so on. The link between the primary and child business components defines these recipients. For example:

- A recipient group could be based on a recipient source specifying one or more accounts and the contacts of these accounts (Account Contacts).
- A recipient group could be based on a recipient source specifying one or more opportunities and the sales-team members of these opportunities (Opportunity Sales Team Members).
- A recipient group could be based on a recipient source directly specifying one or more employees (Employees).

NOTE: Make sure that you select Link Specification for the Primary Contact ID field in the parent business component.

Recipient groups directly determine actual recipients only for outbound communication requests. For advanced communications templates, they determine which fields are available to be substituted with Siebel application data when the template is sent to each specific recipient for the request.

NOTE: If you create a new recipient group using a custom business component, and set the Create Activity flag for an advanced template that uses this recipient group, then you must create a System Activity object on the custom business component in order to specify fields that can be logged with activity logging. See also the description of the Create Activity flag in "Fields for Templates" on page 280.

Predefined Recipient Groups

Several predefined recipient groups, and their underlying business objects and business components, are provided by Siebel Systems. If the predefined recipient groups do not serve all your business needs, you must configure the Siebel application to extend or modify the available recipient groups. For more information, see "Configuring Additional Recipient Groups and Recipient Sources Applets" on page 172.

Table 30 on page 170 lists predefined recipient groups that users or administrators can choose in the Recipient Group drop-down list in the end-user and administrator views for working with outbound communication requests, or in the end-user and administrator views for working with communications templates (Advanced tab). These recipient groups can also be specified in workflow processes created using Siebel Workflow.

(In the Simple tab of the templates views, the Object drop-down list displays business objects for which a primary business component is defined. These entries are not the same as the recipient groups discussed in this section.)

NOTE: The Recipient Group drop-down list itself (for requests or for advanced templates) contains the most up-to-date list of recipient groups for the standard Siebel applications. The list of recipient groups in Table 30 on page 170 may not be comprehensive for your Siebel products.

Most of the recipient groups in the Recipient Group drop-down list apply to both requests and advanced templates, but some recipient groups apply only to templates. Recipient groups *not* applicable to requests display an asterisk (*).

Each recipient group is shown in Table 30, along with the name of the recipient source business object and recipient group business component. In the Administration - Data screen, List of Values view, for type COMM_RECIP_SRC, the Parent LIC column corresponds to the recipient source business object, and the Language-Independent Code column corresponds to the recipient group business component. These elements are described in "About Recipient Groups" on page 168.

Table 30. Predefined Recipient Groups

Recipient Group (Display Value)	Source Business Object (Parent LIC)	Recipient Business Component (Language-Independent Code)
Account Contacts	Account	Contact
Account Team Members	Account	Position
Activity Contact	Action	Comm Contact
Activity Contacts	Action	Contact(All)
Activity Owner	Action	Employee (MM)
Campaign Contacts/Prospects	Campaign	Campaign Recipient
Campaign Team Members	Campaign	Campaign Position
Contacts	Contact	Contact
Division Positions	Internal Division	Position
Email Activity Receiver*	Comm Outbound Email	Comm Outbound Email
Email Activity Sender	Action	Action
Employees	Employee	Employee
Expense Approver	Expense	Expense Report Approver
Expense Owner	Expense	Expense Report Owner
List Contacts/Prospects	List Mgmt	List Mgmt List Member
Messaging Activity*	Messaging	Messaging Activity
Messaging Activity Assignee	Action	Messaging Activity Assignee
Messaging Assignee*	Messaging	Messaging Assignee
Messaging Contacts*	Messaging	Messaging
Opportunity Contacts	Opportunity	Contact
Opportunity Key Contact	Opportunity	Opportunity Key Contact
Opportunity Sales Team Members	Opportunity	Position
Order Contact	Order Entry	Comm Contact
Order Contact (Purchasing Mgr)	Order Entry	Contact(Purchasing Manager)

Table 30. Predefined Recipient Groups

Recipient Group (Display Value)	Source Business Object (Parent LIC)	Recipient Business Component (Language-Independent Code)
Order Sales Team Members	Order Entry	Position
Project Team Members	Project	PS Project Team
Prospect Partner	Channel Partner	Prospect Partner
Quote Contact	Quote	Comm Contact
Quote Sales Rep	Quote	Position
Service Request Contact	Service Request	Comm Contact
Service Request Owner	Service Request	Comm Employee
Time Sheet Approver	Time Sheet	TS Approver
Time Sheet Owner	Time Sheet	TS Owner
Third Party Registration*	Third Party Registration	Training Third Party Registration
Training Class Registrants	Training Class	Training Class Registration
eEvent Attendee	eEvents Event Attendee	eEvents Event Attendee
eTraining Attendee	Training Class Registration	Training Class Registration

Configuring Substitution Fields for Recipient Groups

For the business components defined in the recipient group, all business component fields are generally listed as available substitutions. (System fields, hidden fields, and fields that have a dot in the name are not listed.) Fields are listed from both the recipient group business component and recipient source business component, if they are different.

For example:

- For the recipient group Contacts, the available substitution fields are from the Contact business component, in the form [Field Name].
- For the recipient group Account Contacts, the available substitution fields are from both the Contact business component, in the form [Field Name], and Account business component, in the form [Account.Field Name].

Although it is usually unnecessary to do this, you can explicitly specify the substitution fields that are to be listed for any business component. You use Siebel Tools to do this. The following procedure describes how to configure a substitution field.

To configure a substitution field for a Recipient Group

1. Start Siebel Tools, then select the business component on which the recipient group is based.
2. In the Object Explorer, select Business Component User Prop.

   The Business Component User Properties applet appears.
3. Choose Edit > New Record.
4. Enter details for the new record as follows:

In this field ...	Enter ...
Name	Substitution Field # Replace # by the next number in the list of substitution fields. For example, if the name of the last created substitution field is Substitution Field 17, then name the new field Substitution Field 18.
Value	Enter the name of the business component.

5. Step off the record to save your changes.

NOTE: If you add one or more Substitution Field user properties, only those business component fields identified by these user properties are listed as available substitution fields in the All Templates or My Templates view. No other fields are listed for this business component.

For the example recipient group Account Contacts, any Substitution Field user properties defined on the Accounts *or* the Contacts business component affect the substitution fields for that business component only.

For more information about specifying user properties, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

Configuring Additional Recipient Groups and Recipient Sources Applets

If the Recipient Group picklist does not contain the options you need, follow the instructions in this section. This section describes how to configure your Siebel application to add or modify recipient groups, and to modify user interface features for specifying recipients for outbound communication requests.

For additional information, see "About Recipient Groups" on page 168 and "Predefined Recipient Groups" on page 169. See also "Configuring Substitution Fields for Recipient Groups" on page 171.

If the predefined recipient groups do not serve all your needs, you must:

- Configure the List of Values for the Recipient Group drop-down list to add values you need, or to remove values you do not need. This is described in "Configuring the Recipient Group List of Values" on page 173.

If you are adding any items to the List of Values, you must also:

- Configure the Recipient Sources applet (applet name in Siebel Tools is Comm Source List Applet) to display records associated with the business objects and business components from which you want to specify recipients. This is described in "Configuring Recipient Sources Applets" on page 174. Do this for these views:
  - Outbound Request Overview view in Administration - Communications screen (view name in Siebel Tools is Comm Request Source Recipient Admin View)
  - My Outbound Request Overview view in Communications screen (view name in Siebel Tools is Comm Request Source Recipient Personal View)

  If you only want to remove an option from the List of Values, you do not need to configure the applet in Siebel Tools.

NOTE: If you customize a recipient group to include child business component fields, the business component user properties must exist in the child business component and not in the parent business component.

For additional information about any of the steps in the following procedures, see the relevant Siebel documentation. For configuring Lists of Values, see *Applications Administration Guide*. For configuring applets, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

Configuring the Recipient Group List of Values

This section describes configuring the List of Values for the Recipient Group drop-down list.

To configure the List of Values for the Recipient Group drop-down list

1. Navigate to Administration - Data > List of Values.
2. Optionally, create a new List of Values item of type COMM_RECIP_SRC, if one does not already exist that specifies the business object that will serve as the recipient source for the recipient group you require:
   a. Locate a record of type COMM_RECIP_SRC, for which the field Parent LIC is NULL (that is, contains no value). Parent LIC stands for Parent Language-Independent Code.
   b. Copy the record.
   c. Specify values for the new record. Set the Display Value and the Language-Independent Code fields to the business object name that represents the recipient source—for example, Campaign, Account, and so on.
3. Locate a record of type COMM_RECIP_SRC for which the field Parent LIC is *not* NULL (that is, contains a value).
4. Copy the record.
5. Specify values for the new record:
   a. Set the Parent LIC field to the business object name that represents the recipient source, such as one you specified in Step 2.
   b. Set the Display Value field to any appropriate value.

c Set the Language-Independent Code field to the business component specified as the recipient group business component.

Configuring Recipient Sources Applets

This section describes configuring applets, including association applets, for the Recipient Sources applet, which is described at the start of this section.

To configure Recipient Sources applets

1 Start Siebel Tools.

If an existing applet is defined, named Comm Source *bus_obj_name* List Applet, where *bus_obj_name* is the name of the business object that is to serve as the recipient source, then you do not need to create the applet in Step 2 and Step 3.

2 As necessary, create a new applet, or copy and modify an existing applet.

For example, you can copy the applet Comm Source List Applet, which is displayed as the Recipient Sources applet.

3 Provide a name for the new applet, in the form Comm Source *bus_obj_name* List Applet, where *bus_obj_name* is the name of the business object that is to serve as the recipient source.

NOTE: The applet Comm Source *bus_obj_name* List Applet must be specified for recipient groups for communication requests. The recipient groups listed for communication requests are only those for which applets with this naming convention have been defined. For recipient groups to be used only for communications templates, this naming convention is not required.

The class of the new applet must be CSSFrameListCommSrc.

4 Specify the business component associated with your new applet. If you copied Comm Source List Applet or another existing applet, then you must also modify which business component your new applet is associated with.

5 Add the new applet to the Applet Toggle list for the applet Comm Source List Applet.

When a user chooses the corresponding value from the Recipient Group picklist, as configured in the previous procedure, the Recipient Sources applet will automatically function as the new applet you are defining here. The recipient source business object is identified using a label displayed on the right side of the applet header area.

6 Create a new association applet for the new recipient source, or use an existing association applet, and set it into the new list applet.

An association applet is the applet that appears in a dialog box when you add a new record in a fixed applet that allows association. For example, in this case, it could be for associating an account with a communication request. Every list applet has an Assoc Applet property that you can set to an association applet.

7 Create a link between the Comm Request business component and the new recipient source business component, using S_COMM_REQ_SRC as the intersection table.

For examples, refer to existing links defined for the Comm Request business object.

8 Add the new recipient source business component to the Comm Request business object, specifying the link created in Step 7.

9 For outbound communication requests for email, or fax, or page, define the following user properties on the business component that you plan to use as the recipient group:

- **Recipient First Name Field.** The business component field containing the recipient's first name.
- **Recipient Last Name Field.** The business component field containing the recipient's last name.
- **Recipient Preferred Medium Field.** The business component field containing the recipient's communications channel preference.

  If the setting Only Send Preference is specified for a communication request, a communications template is sent to a recipient if the template's channel type corresponds to the value in the field indicated by this user property.

  If this user property is not set, the preference is retrieved from the Preferred Communications field, if the business component includes such a field.

  For more information about Only Send Preference, see "Fields for Outbound Communication Requests" on page 297.
- **Recipient Email Address Field.** The business component field containing the recipient's email address data.

  For example, in the Contact business component, your application may use the Email Address field for email address data. In this case, Recipient Email Address Field should specify the Email Address field. (This field is specified by default.)
- **Recipient Fax Address Field.** The business component field containing the recipient's fax address data.

  For example, in the Contact business component, your application may use the Fax Phone # field for fax address data. However, Recipient Fax Address Field should specify the Fax Address field (this field is specified by default). Fax Address is a calculated field that prepares fax addressing data obtained from the Fax Phone # field for use with your fax server.

  The formatting for fax-related fields depends on your fax server requirements. By default, the calculated field value for the Fax Address field contains fax formatting that applies to Captaris RightFax. All such field values must be modified as appropriate for your specific fax integration.

  For example, for RightFax, the calculated field value should specify the email account that is monitored by RightFax. For more information, see "Integrating with Fax Systems" on page 357 and "Configuring Fax Integration Elements in Siebel Tools" on page 358.
- **Recipient Pager Address Field.** The business component field containing the recipient's pager address data. See the Recipient Pager Address Field in the Employee business component for the correct pager address format.

Configuring Recipients for Send Commands

This section describes how to specify user properties to configure generic recipients for messages sent using the Send Email and Send Fax commands.

The Send Page command does not use generic recipients, and the user properties identified in this section do not apply.

For information about using the Send commands, see "Sending Email, Fax, and Page Messages" on page 324.

About Generic Recipients

A generic recipient is associated with a business component and identifies a potential recipient related to a specific record. Generic recipients are of two types: nonjoined and joined:

- **Nonjoined generic recipients.** With nonjoined generic recipients, addressing information is obtained directly from fields in the current business component. For details, see "Configuring Nonjoined Generic Recipients" on page 177.

  For example, the Contact business component uses the nonjoined generic recipient type. If a contact record is selected when the user invokes the Send Email command, the Send Email window appears, preaddressed to the selected recipient. Recipient data comes directly from the contact record.

- **Joined generic recipients.** With joined generic recipients, addressing information is obtained from fields in another business component that is joined to the current business component. For details, see "Configuring Joined Generic Recipients" on page 177.

  For example, the Service Request business component uses the joined generic recipient type. If a service request record is selected when the user invokes the Send Email command, the Recipient field allows the user to choose Service Request Owner or Service Request Contact as the email recipient (or the user can choose None and specify the recipient directly). After the user selects the recipient (and other settings), the Send Email window appears, preaddressed to the appropriate recipient.

  For Service Request Owner, recipient data comes from an employee record. For Service Request Contact, recipient data comes from a contact record. In each case, recipient data comes from a record that is joined to the service request record.

User Interface Context for the Recipient Field

The user interface context for the Recipient field is as follows for the Send Email and Send Fax commands:

- For Send Email using the native Siebel email client or for Send Fax, the Recipient field is part of the Pick Recipient dialog box.

  If invoking the Send command does not require the user to select from this field—such as when the current business component uses nonjoined generic recipients for Send Email or Send Fax—the Pick Recipient dialog box does not appear. The Send Email or Send Fax window appears.

- For Send Email using Lotus Notes or Microsoft Outlook, the Recipient field is part of the Recipient/Template dialog box.

  If invoking the Send Email command does not require the user to select from this field—such as when the current business component uses nonjoined generic recipients for Send Email—the Recipient/Template dialog box appears without the Recipient field. After the user selects a template, the Lotus Notes or Microsoft Outlook email window appears.

Configuring Nonjoined Generic Recipients

To configure nonjoined generic recipients for a business component, specify values for the business component user properties listed as follows:

For Contact, Employee, and certain other business components, default values are provided.

- **Recipient Email Address Field.** The business component field from which email address data will be obtained for the Send Email command.

  For example, in the Contact business component, your application may use the Email Address field for email address data. In this case, Recipient Email Address Field should specify the Email Address field. (This field is specified by default.)

- **Recipient Fax Address Field.** The business component field from which fax address data will be obtained for the Send Fax command.

  For example, in the Contact business component, your application may use the Fax Phone # field for fax address data. However, Recipient Fax Address Field should specify the Fax Address field (this field is specified by default). Fax Address is a calculated field that prepares fax addressing data obtained from the Fax Phone # field for use with your fax server.

  The formatting for fax-related fields depends on your fax server requirements. By default, the calculated field value for the Fax Address field contains fax formatting that applies to Captaris RightFax. All such field values must be modified as appropriate for your specific fax integration.

  For example, for RightFax, the calculated field value should specify the email account that is monitored by RightFax. For more information, see "Integrating with Fax Systems" on page 357 and "Configuring Fax Integration Elements in Siebel Tools" on page 358.

Additional Requirement for Nonjoined Generic Recipients

When you configure nonjoined generic recipients, the business component must belong to the Messaging Recipient business object. (The Contact and Employee business components already belong to this business object.)

Configuring Joined Generic Recipients

To configure joined generic recipients for a business component, specify values for the business component user properties listed as described below. For Service Request and certain other business components, default values are provided.

- **Recipient Id Field *N* (where *N* is an integer).** A comma-delimited list of three required values, and an optional fourth value, that points to records in the joined business component. This user property is used by the Send Email and Send Fax commands.

  For example, the Service Request Contact joined generic recipient type is defined for the Service Request business component, using the value "Contact Id, Contact, Service Request Contact." The value for this user property includes the following elements:

  - The foreign key field in the parent business component that points to records in the joined business component. In our example, this field is Contact Id, a field in the Service Request business component that corresponds to the Id field in the Contact business component.
  - The joined business component. In our example, this is Contact.
  - The text of the label to appear for this generic recipient in the Recipient field. In our example, this is Service Request Contact.
  - An optional fourth element that identifies the primary key field in the target business component to join to. By default, the field name is assumed to be Id. Include this element in the user property value if the field name is other than Id. (In our example, this element is unnecessary because the Contact business component uses the Id field.)

You can specify multiple joined generic recipients for a business component. That is, you can define the user property Recipient Id Field 1 to specify one joined generic recipient (such as Service Request Contact), Recipient Id Field 2 to specify another recipient (such as Service Request Owner), and so on. The order of the integers in the user property names determines the order in which the joined generic recipients are listed in the Recipient field.

Additional Requirements for Joined Generic Recipients

When you configure joined generic recipients, also consider the following requirements:

- For Send Email and Send Fax, the user properties of the nonjoined type described in "Configuring Nonjoined Generic Recipients" on page 177 must be set in the joined business component—such as in the Contact business component, joined from the Service Request business component.
- The joined business component, such as Contact, must belong to the Messaging Recipient business object. (The Contact and Employee business components already belong to this business object.)
- If a business component has user properties defined to specify both nonjoined and joined generic recipients, the joined recipients take precedence.
- If any user property definitions of Recipient Id Field *N* are specified as inactive in Siebel Tools, then all such definitions with higher numbers are also inactive.

Configuring Default Templates for Send Email Command

This section describes how to configure a default communications template for an applet, for use with the Send Email command (F9 shortcut). A default template is preselected and populates the message area when the Send Email command is invoked, based on the current (active) applet at the time the command was invoked.

For more information about communications templates, see Chapter 9, "Configuring Communications Templates."

To specify that a specific email template is the default for a given applet, specify the template name as the value for the Mail Template applet property for the applet. You do this in Siebel Tools.

For example, by default, the email template Service Request Assignment has been associated in this manner with the applet Service Request List Applet and with other applets that display service request records in the Service screen.

When the Mail Template applet property has been set for an applet, and the repository changes have been compiled and distributed, then users do not need to explicitly select the template from the Body drop-down list when they use the Send Email command when this applet is active.

Setting a default template for the Send Email command, as described above, overrides template visibility by channel. For example, if a template is specified as the default email template for Change Request applets, this same template will also be the default template for these applets for the fax channels (using the Send Fax command).

For more information about using Siebel Tools, see *Using Siebel Tools*, *Configuring Siebel Business Applications*, and *Siebel Developer's Reference*.

7 Configuring Advanced Communications Features

This chapter provides information about advanced configuration for communications features (mostly for session communications). It includes the following topics:

Using Macro Expansion for Character Fields

Some parameter values for events or commands, communications drivers, or configuration parameters can contain macro-expansion characters, which allow the actual values of those fields to be calculated based on contextual data such as the current business component or other runtime data.

For information about macros that represent runtime values such as the current agent's login or a hoteling agent's extension, see "Macros for Parameter Values" on page 182.

For examples illustrating macros and macro-expansion, see "Macro-Expansion Examples" on page 194.

Macro-Expansion Syntax Elements

Macro expansion uses the following syntax elements:

- Brackets ([ ]) may contain a field name from the record that is currently selected in the Siebel application—for example, a record displayed in a dialog box. Brackets are applicable to event and command parameters only.
- Braces ({ }) may contain a field name from the current business object, business component, work item, or session data, or may contain one of the macros described in "Macros for Parameter Values" on page 182. The Communications Server processes the content of braces after executing a pop-up applet. This applies to Command Data only.
- A colon (:) following a field name functions as a separator to introduce modifying keywords, attributes, subfields, or numeric ranges that determine what data to extract and substitute. For information about how this element is used in phone number processing, see "Macro Expansion with Phone Numbers" on page 188.
- A backslash (\) functions as an escape character, allowing you to place a literal bracket, brace, or colon in a value without its being subject to macro expansion.

Macros for Parameter Values

Several macros (sometimes referred to as special fields) can be used within values for applicable event or command parameters, configuration parameters, or, in limited cases, driver parameters.

NOTE: Unless otherwise stated, the macros described in this section cannot be used within values for communications driver parameters.

When the parameter value is read into memory (such as when the event or command is invoked), particular values are substituted for the macro name. These macros are not specific to any particular communications system, such as a specific CTI middleware product.

Some macros require work item attributes to be provided. Notations for specifying attributes are shown in examples. For details, see "Work Item Attributes" on page 190.

Event attributes are also available for use in parameter values that support macro expansion. For example, the event attribute "SiebelChannelType" represents the channel type of the current work item. For more information, see "Driver Event Attributes" on page 408.

The macros that start with the at sign (@) are static variables with values that are defined at runtime (values for some of these macros may change during a user session). The macros that start with the dollar sign ($) are actually functions that return a value. The macros, which have names that start with the at sign (@) or the dollar sign ($), are listed as follows:

- **@ACDDNList.** The list of ACD DNs (extensions of type "A") associated with the current agent.

  This macro may be used, for example, in a value for the driver parameter Service:ACDDNList, for the Siebel CTI Connect driver.

  The values are separated by commas.
- **@AgentId.** Agent login for the current agent.

- **@AgentPin.** Password for the agent login for the current agent.
- **@ClientHostName.** Host name of the agent's computer.
- **@ClientIP.** IP address of the agent's computer.
- **@Configuration.** The current configuration for the agent.
- **@CountryCode.** The country code applicable to the agent's location.

  If the agent's country is the U.S., this macro returns an empty value.

  If the agent's country is other than the U.S., the macro returns the applicable country code, preceded by a plus (+). For example, if the agent's country is France, the macro returns +33.
- **@DeselectedWorkItem.** The item in the Work Items list in the communications toolbar that had the focus just prior to the item that currently has the focus. (The item with the focus is represented by the @SelectedWorkItem macro.)

  This macro must be specified with an attribute value, as follows:

  @DeselectedWorkItem:*attribute_name*

  where *attribute_name* is the work item attribute you are providing.

  For more information, see "Work Item Attributes" on page 190.
- **$DialingRuleMethod.** Used within parameter values for the DialingFilter.Rule*N* configuration parameters, this macro calls a custom business service method to determine how to translate phone numbers when the dialing filter is in effect.

  The custom business service must recognize the input arguments Filter and PhoneNumber. Filter is the filter in effect for the parameter that invokes the macro. PhoneNumber is the number that needs to be translated. For example, for the following dialing rule, Filter is 650:

  DialingFilter.Rule2="650->$DialingRuleMethod(*myservice.mymethod*)"

  The business service must also support an output parameter called PhoneNumber. For example, if your dialing filter rule is to be applied to the U.S. number (650) 123-4567, where numbers with prefix 123 in the area code (650) should be changed to use prefix 555, the business service method might return the PhoneNumber output parameter with the value "916505554567".

  After a dialing filter rule is applied that uses the macro $DialingRuleMethod, no other filters are applied.

  For more information about using this macro with dialing filters, see "Working with Dialing Filters" on page 195.
- **@DNList.** The list of DNs (standard extensions of type "S") associated with the current agent.

  This macro may be used in a value for the driver parameter Service:DNList, for the Siebel CTI Connect driver.

  The values are separated by commas.

- **@EditControl.** The data in the edit field (text box) control in the communications toolbar.

  This macro yields a value only when the edit field in the communications toolbar has the focus and contains a value. Use @EditControl when agents will input values other than phone numbers into the communications toolbar's edit field—such as to log in to the switch.

- **$GetCommandStatus.** Obtains the status of the device command for a communications command. Possible status values returned are:

  - Checked – command is in toggled-down state, button displays (for example, if the agent indicates Not Ready for new work items, the button is toggled down)
  - Blinking – command is available, button is blinking
  - Enabled – command is available, button is enabled

  NOTE: This macro can be used within command parameters, but not within event parameters. Events are unable to use this macro to determine the status of a device command.

  - Disabled – command is not available, button is disabled

  You can obtain the status with an expression like this:

  `$GetCommandStatus(`*`device_command`*`)`

  where *device_command* is the name of the device command for the command.

  For example, a command parameter to determine the status of the device command may be defined as follows:

  FilterSpec="[$GetCommandStatus(ChangeNotReadyState)]='Checked'"

  In this case, if the status of ChangeNotReadyState is Checked, then FilterSpec matches, and the command containing this FilterSpec parameter can execute.

- **$GetInboundWorkItemAttr.** Obtains the attributes of an inbound work item.

  You can obtain the work item attributes with an expression like this:

  $GetInboundWorkItemAttr(*channel_name*, *attribute_name*)

  where *channel_name* is the name of the locale-independent channel for the work item, and *attribute_name* is the work item attribute you are interested in.

  For example, a command data parameter to obtain the tracking ID of an inbound voice work item may be defined as follows:

  Param.TrackID="{$GetInboundWorkItemAttr(Voice, DriverWorkTrackID)}"

  For more information, see "Work Item Attributes" on page 190.

- **$GetWorkItemAttr.** Obtains the attributes of a work item—for example, in an event handler evaluating a new work item, or in a command.

  You can obtain the work item attributes with an expression like this:

  $GetWorkItemAttr(*workitem_ID*, *attribute_name*)

  where *workitem_ID* is the ID number of the work item, if known, and *attribute_name* is the work item attribute you are interested in.

  For example, an event handler parameter to verify that the ContactId attribute (a custom work item attribute) has a value may be defined as follows:

  FilterSpec="[$GetWorkItemAttr(SiebelWorkItemID, ContactId)] IS NOT NULL"

  For commands operating on the selected work item, you can also obtain work item attributes using @SelectedWorkItem:*attribute_name*.

  For more information, see "Work Item Attributes" on page 190.

  For more information about the SiebelWorkItemID attribute, see "Driver Event Attributes" on page 408.

- **$HotelingPhone.** Retrieves an agent's extension if the agent is using a hoteling teleset.

  The field value is calculated at run time according to these rules:

  - Uses the login name of the currently selected agent (employee), or the name of a service request owner, to find the agent's runtime extension, if the agent is using a hoteling teleset. In the first example below, "Login Name" is the Login Name field of the current business component, which corresponds to the LOGIN column in the S_USER table.
  - If the extension is not found, $HotelingPhone retrieves the employee's extension number from the S_USER table.

  The following example parameter definition in a make-call command (command data definition) may generate the desired extension, based on the Login Name field:

  Param.PhoneNumber="{$HotelingPhone(Login Name):Lookup}"

  Alternatively, the following example parameter definition for a make-call command may generate the desired extension, based on the Owner field. Use a parameter value like this for a command invoked when the current business component is Service Request, to call the owner of the current service request record.

  Param.PhoneNumber="{$HotelingPhone(Owner):Lookup}"

  For more information, see "Configuring Telesets for Hoteling" on page 197.

  NOTE: The macro @HotelingPhone is still supported, for compatibility with release 6.*x*, but it is recommended that you use $HotelingPhone instead.

- **@Language.** The language code applicable to the agent's Siebel application session.

  For example, the language code for U.S. English is "ENU".

- **@Now.** Returns the current timestamp, using the following format:

  %month/%day/%Year %H:%M:%S

- **@Phone.** Represents a phone number, whose value is calculated at run time according to these rules:
  - If the Phone # field in the communications toolbar has the focus and contains a value, @Phone is equal to the value of this field.
  - If the Phone # field has no value, but the currently active applet field is of the type DTYPE_PHONE, @Phone is equal to the value of this field.
  - If the Phone # field has no value and the currently active field is not of the type DTYPE_PHONE, @Phone is equal to the value of the field referred to by the Primary Phone Field user property for the business component.

  In all other cases, the @Phone macro contains no data.

  For more information, see "Macro Expansion with Phone Numbers" on page 188.
- **@PrimaryQueueList.** The list of ACD queues associated with the agent and designated as primary.

  For example, a login command may include a parameter like this to identify which ACD queues to log the agent into:

  Param.ACDQueue="{@PrimaryQueueList}"

  The values are separated by commas.
- **@QueueId.** The list of ACD queues associated with the agent and designated as primary.

  NOTE: @PrimaryQueueList replaces @QueueId, but you can still use @QueueId for compatibility purposes.
- **@QueueList.** The list of ACD queues associated with the agent.

  This list includes all such queues, including those marked as primary and those not marked as primary. The values are separated by commas.
- **@Random.** A string containing 10 random digits.

  This macro may be used for testing purposes.
- **$RemoteConnectStr.** Name of the remote call center. This macro, which can be used with transfers and conference calls between call centers, derives the name of a remote call center's communications configuration from either:
  - The ConnectString configuration parameter (if defined), or
  - The extension number of the agent to be called

  For example, a command parameter in a transfer or conference command may generate the desired configuration name in this fashion:

  Param.RemoteConnectStr="[$RemoteConnectStr(@Phone)]"

  For more information, see "Configuring Remote Transfers and Conferences" on page 204.
- **$RemoteConnectStr2.** Name of the remote call center. This macro, which can be used with transfers and conference calls between call centers, derives the name of a remote call center's communications configuration from either:

- The ConnectString configuration parameter (if defined), or
- The employee ID of the agent to be called

For example, where the SelectBusObj and SelectBusComp parameters are set to "Employee," a command parameter in a transfer or conference command may generate the desired configuration name in this fashion:

Param.RemoteConnectStr="[$RemoteConnectStr2(Id)]"

The Employee business component uses the Id field to uniquely identify an employee record. A command that references another business component might need to use another field name to uniquely identify a record.

For more information, see "Configuring Remote Transfers and Conferences" on page 204.

- **@SelectedDN.** The currently selected extension in the Communications options of the User Preferences screen.
- **@SelectedQueue.** The currently selected ACD queue record in the Agent Queues list in the Communications options of the User Preferences screen.
- **@SelectedText.** Text the user has selected in the browser, such as in a field in an applet within the Siebel application. The text could be used, for example, to provide a phone number to call or to provide some other input data for a command.
- **@SelectedWorkItem.** The item in the Work Items list in the communications toolbar that has the focus. (The item that had the focus just prior to this item is represented by the @DeselectedWorkItem macro.)

  This macro must be specified with an attribute value, as follows:

  @SelectedWorkItem:*attribute_name*

  where *attribute_name* is the work item attribute you are providing.

  For more information, see "Work Item Attributes" on page 190.
- **@UserId.** Siebel login ID of the current Siebel user.
- **@UserName.** Login name of the current Siebel user.
- **@WorkDuration.** Length of time, in seconds, of the current work item—the time elapsed since the work item started.
- **@WorkObjectID.** Row ID of the work-item tracking object that was transferred to the agent or that was created as a result of creating an event log and defining a value for the AfterWork event log parameter.
- **@WorkStartTime.** Date and time (timestamp) when the work item arrived or was initiated. The timestamp is in the following format:

  %month/%day/%year %H:%M:%S

@Phone Macro Example

The example command and associated command data shown in Table 31 and Table 32 use the @Phone macro, and can be used to dial any phone field on any business component.

Table 31. Command: MakeCallToCurrentPhone

Parameter Name	Parameter Value
Description	Make Call to "{@Phone}"
DeviceCommand	MakeCall
Hidden	TRUE

Table 32. Command Data: MakeCallToCurrentPhone

Parameter Name	Parameter Value
AttachContext	TRUE
Param.CallNotifyText	Call from {@UserName}...
Param.PhoneNumber	{@Phone:PhoneTypeLookup}
RequiredField.@Phone	?*

Macro Expansion with Phone Numbers

Macro expansion provides many options for retrieving and manipulating phone number data for use with communications events or commands. This section applies to event and command parameters only.

The macro @Phone is often used to retrieve a phone number; for a detailed description of this macro and its encoded logic, see "Macros for Parameter Values" on page 182. Alternatively, you can explicitly specify the name of a field from which to extract phone number data.

NOTE: When you work with phone number fields in event and command parameter definitions, you must be aware how phone number data is formatted in your deployment of the Siebel Business Applications.

For information about phone number formatting, including formatting for international numbers, see *Applications Administration Guide*.

You can use the following special macro-expansion features when retrieving phone number data:

- A keyword to specify how to apply dialing filters to the field value. See "Keywords to Specify Dialing-Filter Behavior" on page 189.
- A keyword to indicate the part of the phone number that you want to extract. See "Keywords to Extract Part of a Phone Number" on page 189.

- A specific range of numerals to extract from the field value. See "Numeric Ranges to Extract" on page 190.

Keywords to Specify Dialing-Filter Behavior

Follow the field name with a colon (:) and one of the following keywords to specify whether to filter the field data using dialing filter rules specified using DialingFilter.Rule*N* configuration parameters. Dialing filters are applied in numeric sequence until a match is found.

Dialing filter rules allow you to perform an intelligent translation of a phone number, in order to optimize dialing. The macro $DialingRuleMethod can be used within a dialing filter rule definition to invoke a business service method to specify custom logic in the application of the dialing filter rule to the results.

You can use the following keywords with either the @Phone macro or a field whose name you explicitly specify:

- **Lookup.** Applies the dialing filters to the field data.

  You can use Lookup with fields that you know to contain phone number data that can be successfully filtered with your dialing filters.

- **PhoneTypeLookup.** Checks if a field is of type DTYPE_PHONE (as defined in Siebel Tools). If it is of this type, the dialing filters are applied to the field data. If it is not of this type, the dialing filters are not applied to the field data.

CAUTION: Do not modify the type for any field defined as DTYPE_PHONE, or phone number data will not be correctly handled in your application.

For examples, see "Macro-Expansion Examples" on page 194. For more information about DialingFilter.Rule*N* configuration parameters, see "Specifying Parameters for Communications Configurations" on page 52.

Keywords to Extract Part of a Phone Number

Follow a field name with a colon (:) and one of the following keywords to specify the portion of the phone number field to be extracted and substituted. You can use these keywords only with fields of type DTYPE_PHONE. If none of these keywords is specified after a field name, then the phone number from the field corresponds to IntlNumber. IntlNumber may also be specified explicitly.

NOTE: The plus sign (+) is included only in the value for IntlNumber, when this keyword represents an international number. In the following keyword descriptions, references to the country code do not include the plus sign (+) character:

- **IntlNumber.** Represents *one* of the following:
  - If the country code for the number matches the current locale for the Siebel Server (Application Object Manager): represents the domestic phone number (number without the country code or extension).
  - If the country code for the number does *not* match the current locale for the Siebel Server (Application Object Manager): represents the international phone number ("+" and country code and number, without the extension).

- **Digits.** The full phone number as stored in the database field (country code, phone number, and extension).
- **Country.** The country code only (without the phone number or extension).
- **Number.** The phone number only (without the country code or extension).
- **Extension.** The phone number extension only (without the country code and phone number). This keyword is applicable when the phone number field contains an extension number prefaced with an "x" or another localized character).

Numeric Ranges to Extract

Follow the field name or a keyword with a colon (:) and a range of numerals to be extracted from the field value. A substring numeric range can be specified directly after the field name or after a keyword indicating the part of the phone number.

Primary Phone Field User Property in Business Components

Any business component can have a named user property called Primary Phone Field. The property value is the field name of the same business component that contains the phone number and that should be dialed when a make-call command is invoked on an applet that uses this business component.

For example, the Account business component can have a Primary Phone Field user property set to the value Main Phone Number. This tells Communications Server that Main Phone Number is the field that should be dialed by default when dialing is requested on an account.

Under certain circumstances, the value of the @Phone macro is equal to the field designated as the Primary Phone Field.

For more information about specifying user properties, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

Work Item Attributes

Table 33 on page 191 describes work item attributes that can be specified in values for parameters that support macro expansion. Communications events and commands can reference these attributes as subparameters of the parameter Param, or as elements within other parameter values where macro expansion is supported. The following macros, when used in parameter values, can process these attributes: @SelectedWorkItem, @DeselectedWorkItem, and $GetWorkItemAttr.

In addition, custom work item attributes can be stored and updated by using the WorkTrackingObj parameter in event log or command data definition. Database records that store work item attributes can be updated after work items are released, by using the AfterWork event log parameter. (The WorkObject event response parameter can also be used to update database records for tracking work items.)

For example, if an event log includes a parameter definition like the following, then "ContactId" becomes a custom work item attribute that is available for use wherever work item attributes can be accessed. For example, commands that update the customer dashboard can access such attribute values:

```
WorkTrackingObj.ContactId="{Contact.Id}"
```

Alternatively, if an event log includes a parameter definition like the following, then text you specify is displayed in the Work Items list in the communications toolbar:

```
WorkTrackingObj.Description="your descriptive text goes here"
```

NOTE: Of the predefined work item attributes described in Table 33 on page 191, only ParentWorkItemID and Description can be updated.

For more information about the WorkTrackingObj and AfterWork event log parameters, see "Event Logs" on page 123. For more information about the WorkTrackingObj command data parameter, see "Command Data" on page 142. For more information about the WorkObject event response parameter, see "Event Responses" on page 110.

For additional attributes that can be specified in event handlers, see "Driver Event Attributes" on page 408.

Table 33. Work Item Attributes

Attribute Name	Description
ChannelType	The language-independent value representing the channel type of this work item.
ChannelTypeLocale	The locale-dependent value for the channel type of this work item.
Description	A description of this work item. The value for this attribute comes primarily from the client handle methods WorkItemStarted and IndicateNewWorkItem. For more information, see Appendix A, "Developing a Communications Driver."
DriverWorkTrackID	The ID number of this work item as tracked by the driver.
IsTransferred	Indicates whether the work item has been transferred using Siebel Universal Queuing. Possible values are TRUE or FALSE (the default is FALSE). If the value is TRUE, then the Reject button on the Accept/Reject dialog box is disabled. This dialog box allows an agent to accept or reject a work item.
IsRevoked	Indicates whether the work item has been rejected by an agent (using the Accept/Reject dialog box) or has been taken back by Siebel Universal Queuing. Possible values are TRUE or FALSE (the default is FALSE). This attribute is TRUE whenever the service handle method RevokeQueuedWorkItem has been invoked for the work item.

Table 33. Work Item Attributes

Attribute Name	Description
ParentWorkItemID	The ID number of the parent of this work item. Note the following behavior for parent and child work items: ■ When a parent work item is released, all child work items are released. ■ When a parent work item is suspended, all child work items are suspended. ■ When a parent work item is resumed, all child work items are resumed. ■ When a child work item is started or resumed, the parent work item gains the focus. ■ Agent can use the Work Items list in the communications toolbar to select a child work item and operate on this child work item individually. ■ Parent and child work items can be of different channels, such as when an outbound voice call is placed to a customer who initiated an email work item. See the sample communications configurations provided by Siebel Systems for examples.
ProfileName	The communications driver profile for which this work item is applicable.
UQWorkItemID	The ID number of this work item from Siebel Universal Queuing, if this work item was originally assigned by Siebel Universal Queuing.
ViewBookmark	The bookmark for the view which was current when this work item was suspended. The bookmark data is serialized and compressed into a string.
WorkDuration	The duration of this work item. The value is the same as the @WorkDuration macro.
WorkItemID	The ID number of this work item.
WorkObjectID	The work tracking object ID number. The value is the same as the @WorkObjectID macro.

Table 33. Work Item Attributes

Attribute Name	Description
WorkStartTime	The start time of this work item. The value is the same as the @WorkStartTime macro. The value for this attribute comes primarily from the client handle method WorkItemStarted. For more information, see Appendix A, "Developing a Communications Driver."
WorkState	The state of this work item. Possible values are Created, Active, Suspended, and Released. The value for this attribute comes primarily from client handle methods. ■ When IndicateNewWorkItem is invoked, the WorkState value is Created. ■ When WorkItemStarted is invoked, the WorkState value is Active. ■ When WorkItemSuspended is invoked, the WorkState value is Suspended. ■ When WorkItemResumed is invoked, the WorkState value is Active. ■ When WorkItemReleased is invoked, the WorkState value is Released. For more information, see Appendix A, "Developing a Communications Driver."

Macro-Expansion Examples

The examples in Table 34 on page 194 demonstrate macro-expansion usage. From the nominal parameter value shown in the second column, macro expansion calculates the resulting value shown in the third column. Each example is explained to show how the result was generated.

Table 34. Macro-Expansion Examples

Parameter Name	Parameter Value	Result	Explanation
Param.PhoneNumber	#8[Phone #: 7-10]	#85000	If the value of the Phone # field in the current business component is 6504775000, macro expansion extracts the seventh through tenth digits of the phone number. The characters "#8" prefix the result. Because no keyword is specified, the Number part is assumed, and the country code and extension are excluded. Because the macro-expansion characters representing the phone number are enclosed in brackets, this Phone # field could be in a dialog box or in the current business component.
Param.PhoneNumber	{Phone #: Number:6-10}	75000	Assume the same value for the Phone # field as in the previous example. Macro expansion extracts the sixth through tenth digits of the phone number. In this example, the Number keyword is explicitly specified. Because the macro-expansion characters representing the phone number are enclosed in curly braces, this Phone # field could not be in a dialog box.
Param.GetId	{Id}	10-CSAE	If the Id field for a record in the current business component is 10-CSAE, macro expansion yields this string as the result.
Param.Greet	\{Hi\}	{Hi}	The backslashes escape the curly brace characters, allowing them to be included literally as part of the result. Because "Hi" is not enclosed in curly braces, it is not interpreted as a field name.

Table 34. Macro-Expansion Examples

Parameter Name	Parameter Value	Result	Explanation
Param.ExtField	{Extension}	5000	For a field called Extension, macro expansion extracts the value of this field—for example, 5000. Because the Extension field is not of type DTYPE_PHONE, you cannot use keywords to specify a phone number part to extract.
Param.Ext	{Work Phone #: Extension}	6000	If the value of the Work Phone # field of the current business component (for example, Contacts) is (650)4775000 x6000 for a U.S. phone number and extension, only the extension number after the "x" is extracted.
Param.PhoneNumber	{Work Phone #: Lookup}	*phone num*	Macro expansion generates a phone number based on applying the dialing filters to the data in the Work Phone # field of the current business component.
Param.PhoneNumber	{@Phone: PhoneTypeLookup}	*phone num*	Macro expansion generates a phone number based on the logic described for the @Phone macro, in "Macros for Parameter Values" on page 182. Dialing filters are applied only if a field is of type DTYPE_PHONE.

Working with Dialing Filters

Dialing filters are used by the Siebel application to manipulate telephone numbers for voice calls made, transferred, or for conference calls made.

Dialing filters specify a set of phone-number translation rules that are invoked when the Lookup or PhoneTypeLookup keyword is specified in macro-expanded text in a communications command for the voice channel (using Siebel CTI). Dialing filters are defined as configuration parameters in the All Configurations view in the Administration - Communications screen.

In each of the examples in this section, the first set of numbers is searched for. If there is a match, the searched numbers are translated to the numbers after the -> symbols.

Filter rules are checked for matches in numerical sequence. The last rule in the sequence should always be defined to match any number.

The numeric elements in the names of the filter rule parameters are applied by string comparison, from small to large. For example, this means that a dialing filter named DialingFilter.Rule11 will be applied *before* one named DialingFilter.Rule2. Consequently, use the same number of digits for each rule, according to your total anticipated number of rules. For example, the numeric elements in the filter rules could use a sequence starting with 000, followed by 001, and so on, up to 999.

NOTE: How dialing filters actually function is subject to the command data definition for the command that is performing the make-call operation.

By default, the Lookup and PhoneTypeLookup keywords used in command data definitions do the following:

- Treat a phone number as domestic (that is, omit the country code) if the country code matches the current locale.
- Treat a phone number as international (that is, include the country code) if the country code does not match the current locale.

For CTI implementations supporting multiple call centers, consider the dialing filter requirements among the factors in deciding how to organize your agents in communications configurations. You may decide, for example, to ensure that all agents included in a single configuration are located within a single location for which the same set of dialing filters will apply. For example, this grouping may be defined as a particular call center facility, or a particular locale corresponding to a country code.

For more information about configuration parameters, including DialingFilter.Rule*N*, see "Specifying Parameters for Communications Configurations" on page 52.

For more information about the Lookup and PhoneTypeLookup keywords, keywords to represent parts of phone numbers, phone-number formatting, and the $DialingRuleMethod macro, see "Using Macro Expansion for Character Fields" on page 181.

For more information about international telephone numbers, see *Applications Administration Guide*.

Dialing Filter Examples

The following are examples of dialing filter rules:

- DialingFilter.Rule001="650477->" – This rule takes a ten-digit domestic phone number (for example) and translates it into four digits for dialing an internal extension.

  You can use such a rule for internal calls, call transfers, or call conferences, where, for example, the U.S. area code 650 and prefix 477 apply to the call center. Or use such a rule for calls, transfers, or conferences to another call center, where this area code and prefix apply to the destination call center. For more information, see "Configuring Remote Transfers and Conferences" on page 204.

- DialingFilter.Rule002="650->9" – This rule takes a ten-digit domestic phone number (for example) and translates it into eight digits for dialing an outside local call in the U.S.
- DialingFilter.Rule003="+33->901133" – This rule takes a French phone number (any number for which the country code is 33, when the current locale is *not* France) and prefaces it with the digits that are required for dialing an international number when your locale is the U.S. or Canada.

- DialingFilter.Rule004="+->9011" – This rule takes any international phone number (that is, any number for which the country code does not match the current locale) and prefaces it with the digits required for dialing an international number when your locale is the U.S. or Canada.
- DialingFilter.Rule005="->91" – This rule takes any domestic phone number (that is, any number that did not match the preceding rule) and prefaces it with the digits required for dialing an outside call to another area code in the U.S.

For the preceding examples:

- If a telephone number matches 650477****, then Rule001 will be applied.
- If the number does not match this pattern, but matches 650*******, then Rule002 will be applied.
- If the number does not match this pattern, but matches a French number (when the current locale is not France), then Rule003 will be applied.
- If the number does not match this pattern, but matches any international number (for which the country code does not match the current locale), then Rule004 will be applied.
- If the number does not match any of these patterns, then Rule005 will be applied. As noted, the last defined rule should match any number.

The following example is an alternative definition for DialingFilter.Rule002.

- DialingFilter.Rule002="650->$DialingRuleMethod(*service.method*)"—This rule takes a ten-digit domestic phone number (for example) and translates it into a number that is determined by the specified business service method. You can use such a rule to invoke a method that will provide a correct phone number to dial, where other filter rules may not be able to filter unambiguously.

  For example, if internal numbers use a prefix, such as 477, that is also used outside the company, the business service invoked using this macro can determine the correct number to dial by identifying the specific number as internal or external.

The following example is an alternative definition for DialingFilter.Rule004.

- DialingFilter.Rule004="+->$DialingRuleMethod(*service.method*)" – This rule takes an international phone number and translates it into a number that is determined by the specified business service method.

Configuring Telesets for Hoteling

You can configure the telesets in your call center for hoteling, which allows an agent to log in to the Siebel application from any one of a pool of telesets and computers that have been configured for this purpose, and to be able to use voice communications features.

You configure hoteling by associating a teleset with the host name of the computer that is located at the same station as the teleset. Do this in the All Telesets view in the Administration - Communications screen.

For instructions for configuring telesets, see "Specifying Telesets" on page 62.

When you have specified a host name to enable hoteling for a teleset, any agent logging in to the Siebel client on the hoteling computer will use the standard extension of the hoteling teleset that is associated with that computer's host name.

Hoteling Requirements and Issues

Note the following requirements and issues for implementing hoteling:

- In order for multiple call-center agents to be able to log in to one computer, the computer or browser may need to be prepared for this purpose, for instance by creating Windows profiles for all hoteling users. For instructions for such tasks, refer to your computer system documentation.
- After each agent is added to the communications configuration, the agent can then be associated with a teleset. Associating a host name with a teleset overrides any such association of the teleset with an agent.
- If an agent logs into the Siebel application on a computer that is not associated with a hoteling teleset, voice communications are enabled for the agent only if that agent is explicitly associated with the teleset at the new location.
- Hoteling can be configured only for computers that are always expected to be located at the same station as the hoteling teleset. A laptop computer that an agent connects to the network at any of several locations, for example, can be configured for hoteling only for a single location where a particular teleset is located.
- Agents who log in to a hoteling computer cannot specify any other teleset in the Communications options of the User Preferences screen. The agent can specify a different extension for this teleset, however, if more than one extension is defined.
- Commands for internal calls to employees who are using hoteling telesets can be configured to retrieve the runtime extension of the employee by using the $HotelingPhone macro. For more information, see "Macros for Parameter Values" on page 182.

Supporting Multitenancy

Siebel Communications Server supports the multiple-organization feature in the Siebel Business Applications. This feature, called multitenancy for call centers, helps provide the contact center, including those supporting multiple channels, with flexibility to organize its work and agent resources and to control visibility of data.

Contact centers that provide outsource services to other companies, or to multiple internal entities, can support these multiple entities using multitenancy.

In an outsource contact center, agents with expertise supporting products or services from Company A, for example, are assigned positions in the organization defined for Company A, while agents supporting Company B are assigned positions in the organization defined for Company B.

Agents can be assigned to multiple positions in multiple organizations, according to their expertise and the contact center's operational requirements.

For more information about positions, responsibilities, and organization visibility for Siebel applications, see *Security Guide for Siebel Business Applications* and *Applications Administration Guide*.

Setting the MultiTenancy Configuration Parameter

The Siebel Communications Server configuration parameter MultiTenancy allows you to specify whether organization-visibility rules should apply:

- Set this parameter to TRUE if your Siebel implementation uses multitenancy. The parameter is set to TRUE by default.
- Set this parameter to FALSE if your Siebel implementation does not use multitenancy.

For more information, see "Specifying Parameters for Communications Configurations" on page 52.

Organization Visibility and Positions

Some Siebel data records can be viewed only by users whose current position is within a particular organization. Other records are visible to users whose positions are in different organizations. Records may be associated with one or more organizations.

Each user can have only one position active at a time. In some situations, a user may be allowed to see data for all of that user's assigned positions.

Organization visibility is defined for both business components and views.

Organization Visibility for Business Components

Organization visibility is defined for each business component, determining which of the following applies to the business component:

- Visibility is for one organization.
- Visibility is for multiple organizations.
- Organization visibility does not apply.

A business component for which organization visibility applies is said to be position-dependent.

Organization Visibility for Views

Organization visibility is enforced for each view. Multiple views that display data for the same business component may enforce visibility differently. For example, in the Contacts screen, the All Contacts view allows a user to view contact records within one organization, while the All Contacts across Organizations view (not available for all users) allows a user to view all contact records for all organizations.

You can determine whether a view enforces organization visibility by using Siebel Tools. For each view object definition:

- Organization visibility is enforced for the view if the field Visibility Applet Type is set to Organization *and* the field Admin Mode Flag is not checked (FALSE).
- Organization visibility is *not* enforced for the view if the field Admin Mode Flag is checked (TRUE).

Changing Position Manually or Automatically

For the agent to view some data, such as for certain screen pops, the agent's position must be changed, either manually or automatically. At any time, an agent can manually change the current position in the Siebel application by using the Change Position view in the User Preferences screen.

If the agent is the recipient of a screen pop for an incoming or transferred work item or a conference call, the agent's position is changed automatically, as appropriate for the context and for the screen-pop data. When an agent retrieves a paused work item, the original screen pop and position are both restored. As described earlier, the business component determines the organization visibility for the screen-pop data, enabling the agent's position to be changed, as necessary.

What screen-pop data to display is determined by the current business component, which is specified in the QueryBusComp parameter in the event response. See also "Event Responses" on page 110.

After an agent's position has been automatically changed, when a work item is concluded, the agent's position remains what it was changed to. The agent must manually change the position again if the new current position is not appropriate.

Scenarios for Generating Screen Pops and Changing Position

When a position is changed, a message is displayed to the user. If the position is not changed, no message appears, except in certain error conditions. Application behavior for generating screen pops and changing positions varies for each of the scenarios is described as follows:

- If the agent's current position matches the screen-pop data, the screen pop is displayed and the agent's position is not changed.
- If the agent's current position does not match the screen-pop data, and the agent has one other position that matches the data, the screen pop is displayed and the agent's position is changed.
- If the agent's current position does not match the screen-pop data, and the agent has more than one position that matches the data, the screen pop is displayed. The agent's position is changed to the first position that matches the data, and the agent is allowed to view all data that is defined to be visible for all the agent's assigned positions.

  The Siebel client allows the agent to view this data by using the VIEW_ALL mode, which ignores organization-visibility rules. A message advises the agent to manually change to a position appropriate for the screen-pop data, so that the agent can subsequently navigate to related records without hindrance.
- If the agent does not have a position that allows the agent visibility to the screen-pop data, no screen pop will be displayed.

  NOTE: Routing inbound work items using Siebel Universal Queuing should take into account the way organizations and positions are defined for your company. This way, you can make sure that a screen pop can always be displayed for an incoming work item.

- If the business component for a view does not have enough data to determine organization visibility, the screen pop will occur without restriction. The position will not be changed. No error message will be displayed.
- Organization-visibility rules for screen pops apply to inbound work items and to transfers and conferences between contact-center users. For inbound work items, the Siebel application always attempts to set the user's position automatically. For transfers and conferences, the position is set only if the screen-pop view enforces organization visibility.
- When a view has no records displayed, organization-visibility rules do not apply for screen pops for call transfers or conferences.
- When organization visibility is not enforced for a view, organization-visibility rules are not used when changing an agent's position.
- Multitenancy affects screen pops differently for different views. Siebel application developers, contact-center managers, and end users need to understand how this issue relates to screen pops.

  Some views, such as administration views, do not have organization-visibility rules. Some views, such as views whose names include the phrase "Across Organizations," allow users to view records for multiple organizations. Organization-visibility rules are applied to screen pops according to the business component data. Verify organization-visibility behavior, changing position manually as needed, for the views involved in your screen pops.

Configuring Communications Log In and Log Out

Agents can log in to call-center communications systems that work with Siebel Communications Server, either automatically or manually. See the following for more information about login issues:

- For end user login procedures, see "Using the Communications Toolbar" on page 314.
- For a description of logging in to selected ACD queues, and of the end user preference Auto Login to Call Center at Startup (in the Communications options in User Preferences), see "Setting Communications User Preferences" on page 305.
- For descriptions of the configuration parameters AutoLogin, AutoLoginCmd, PreferenceLoginCmd, and PreferenceLogoutCmd, see "Specifying Parameters for Communications Configurations" on page 52.
- For a description of the Siebel CTI Connect driver parameter Service:AutoLogout, see "Siebel CTI Connect Driver Parameters" on page 339.

Log In and Log Out Command Configuration

This section documents the command and command data definitions from a Siebel-provided communications configuration file (.DEF), that determine the availability of log in and log out functions for the communications toolbar and for menus. For more information:

- About all command and command data parameters shown below, see Chapter 5, "Configuring Events and Commands."
- About configuring the communications toolbar and communications menu items, see Chapter 6, "Configuring User Interface Elements."
- About macros such as @PrimaryQueueList, @UQAgentLogon, and so on, see "Macros for Parameter Values" on page 182.

In the following commands, SignOnGroup and SignOnGroupInMenu both specify the same set of subcommands, which control login and log out functionality in the communications toolbar and in the Communications submenu of the Tools application-level menu, respectively:

```
[Command:SignOnGroup]
    Hidden          = "TRUE"
    Description     = "Log in"
    ExecuteAll      = "TRUE"
    SubCommand_1    = "LoginToPBX"
    SubCommand_3    = "LoginToUQ"

[Command:SignOnGroupInMenu]
    MenuPosition    = "20"
    Title           = "Log In"
    Description     = "Log in"
    SubCommand_1    = "LoginToPBX"
    SubCommand_3    = "LoginToUQ"

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

    [CmdData:LoginToPBX]
    Param.ACDQueue   = "{@PrimaryQueueList}"
    Param.AgentId    = "{@AgentId}"
    Param.AgentPin   = "{@AgentPin}"

[Command:LoginToUQ]
    MenuPosition     = "20.3"
    Title            = "Log In (Universal Queuing)"
    DeviceCommand    = "@UQAgentLogon"
```

Automatic and Manual Login

This section provides a brief overview of the automatic and manual login capabilities provided for communications-enabled Siebel applications.

Automatic Login

With automatic login, after starting the Siebel application, the end user is automatically logged into all applicable communications systems. Autologin can be enabled or disabled for all users. Alternatively, administrators can allow each user to specify whether to log in automatically.

When autologin is in effect, using the Connect command in the File application-level menu in order to log in to a different Siebel Database, or in order to change the Siebel login you are using, logs you out of the communications systems and then automatically logs you in again.

Agents who use ACD queues can manually log in to or out of any of the queues with which they are associated. Agents do this in the Communications options of the User Preferences screen.

Manual Login

With manual login, after starting the Siebel application, the end user clicks the Log In button on the communications toolbar in order to log in to the applicable communications systems. Autologin must be disabled for all users or for the individual user.

When manual login is in effect, using the Connect command in the File application-level menu in order to log in to a different Siebel Database, or in order to change the Siebel login you are using, logs you out of all applicable communications systems and you must log in again manually.

Agents who use ACD queues can manually log in to or out of any of the queues with which they are associated. Agents do this in the Communications options of the User Preferences screen.

Configuring Automatic Log Out Using Server Script

This section describes an example server script that will provide automatic log out for communications-enabled Siebel applications. This script will be invoked under certain circumstances to automatically log a user out of an ACD queue or out of Siebel Universal Queuing.

To configure this functionality, create a server script that overwrites the Service_PreInvokeMethod event for the ShellUIExit method of the Communications Client business service. The script uses the runtime events ApplicationUnload, WebLogout, and WebTimeout.

The ShellUIExit method may be triggered in any of the following cases:

- The user logs out of the Siebel application by choosing File > Log Out from the application-level menu.
- The user exits the browser.
- The browser fails and the application timeout is reached.

Define the following server script to include this functionality:

```
function Service_PreInvokeMethod (MethodName, Inputs, Outputs)


{
   if (MethodName == "ShellUIExit")
    {
      var CommSrvr_BS = TheApplication().GetService("Communications Client");
      var ip = TheApplication().NewPropertySet();
```

```
        var op = TheApplication().NewPropertySet();
        var m_pValue;
        m_pValue = Inputs.GetProperty("m_pMediaStatusMgr");

           if (m_pValue !="")
              {
                CommSrvr_BS.InvokeMethod("IsTheLastSession", ip, op);
                var bLast = op.GetProperty("LastSession");
                if (bLast == "1")
                 {
                  CommSrvr_BS.InvokeMethod("LogOutCommandName", ip, op);
                 }
              }
         }
    return (ContinueOperation);
    }
```

For more information about Service_PreInvokeMethod, see *Siebel Object Interfaces Reference*. For more information about runtime events, see *Siebel Personalization Administration Guide*.

For more information about methods of the Communications Client business service, see "Communications Client Methods" on page 427.

Configuring Remote Transfers and Conferences

You can configure Siebel Communications Server to allow agents to initiate, transfer, and conference voice calls between call centers—with attached data to support screen pops. This capability is supported for Siebel CTI Connect (using Intel NetMerge) and may also apply to other middleware.

NOTE: For Intel NetMerge, the Call Information Manager server connected to each switch must be configured for multisite operation. For more information, refer to Intel NetMerge documentation provided on *Siebel Business Applications Third-Party Bookshelf*.

Follow the additional guidelines in this section to implement this functionality.

Creating Communications Configurations

You must create separate communications configurations for each call center. For example, for call centers in San Mateo and Emeryville, create a configuration called San Mateo and a configuration called Emeryville.

Each configuration must define the agents applicable to that physical call center. All configurations must exist in the Siebel Database at all call centers. In the example, both the San Mateo and Emeryville configurations exist at both call centers.

NOTE: Session-based communications functionality, such as Siebel CTI, is supported only for single-database environments. Session communications capability is not supported for users of a regional database, to which data is replicated using Siebel Replication Manager.

Specifying Dialing Filters

Each communications configuration must contain configuration parameters specifying dialing filters, in order to support dialing for both locations. These parameters may have names such as DialingFilter.Rule1, DialingFilter.Rule2, and so on. For more information, see "Specifying Parameters for Communications Configurations" on page 52.

For example, assume that you are configuring communications between two call centers located in the U.S. The San Mateo call center is located in area code 650 and has exclusive use of the prefix 477. The Emeryville call center is located in area code 510 and has exclusive use of the prefix 788. Five-digit dialing is supported between these call centers.

With dialing filters such as the examples shown in Table 35, phone numbers for call transfers and call conferences initiated from Siebel Communications Server from either site to the other site (or internal to one site) are converted from ten-digit numbers, as they are stored in the Siebel Database, to four-digit numbers, then prepended with a numeral such as "1" or "2" for five-digit dialing.

Table 35. Dialing Filter Examples

Parameter Name	Parameter Value
DialingFilter.Rule1	510788->1
DialingFilter.Rule2	650477->2

In these examples, the resulting five-digit numbers match extensions defined (with the same number of digits) in the organization's switches and match the extensions defined in the Administration - Communications screen.

The macro $DialingRuleMethod can help you to implement advanced logic for your dialing filters. For details, see "Macros for Parameter Values" on page 182.

Using Macros to Identify Remote Call Centers

This section describes how you can use parameters to obtain a call center configuration name.

In the communications configurations for San Mateo and Emeryville, you add a command parameter like one of the examples below to the command data definition for any applicable commands—such as commands used for initiating a call transfer or conference call to another call center.

The command data parameter Param.RemoteConnectStr can include one of two related macros in its parameter value in order to obtain the desired configuration name:

- **$RemoteConnectStr.** This macro derives the name of a remote call center's communications configuration, using the agent's extension number as a parameter. Table 36 shows examples:

  Table 36. $RemoteConnectStr Examples

Parameter Name	Parameter Value
Param.RemoteConnectStr	[$RemoteConnectStr(@Phone)]
Param.RemoteConnectStr	{$RemoteConnectStr(Owner Phone:Lookup)}

  For more information, see "Macros for Parameter Values" on page 182.

- **$RemoteConnectStr2.** This macro derives the name of a remote call center's communications configuration, using the agent's employee ID as a parameter.

  For example, where the Employee business object and business component are specified, the parameter definition uses the business component field Id to get the name of the configuration. Table 37 shows an example:

  Table 37. $RemoteConnectStr2 Example

Parameter Name	Parameter Value
Param.RemoteConnectStr	[$RemoteConnectStr2(Id)]

Example Command Using $RemoteConnectStr

The example command and command data definitions shown in Table 38 on page 207 and Table 39 on page 207, using the $RemoteConnectStr macro, use the Owner Phone field in the Service Request business component to determine which call center has the phone number.

In this example, the Param.RemoteConnectStr command parameter gets the extension number for the call recipient by macro-expanding Owner Phone:Lookup. The Siebel application looks up which configuration has this extension and returns the name of the call center with this extension.

Table 38. Command: BlindTransferToSROwner

Parameter Name	Parameter Value
Description	Blind Transfer to Service Request Owner
DeviceCommand	TransferMute
Hidden	TRUE

Table 39. Command Data: BlindTransferToSROwner

Parameter Name	Parameter Value
BusComp	Service Request
RequiredField.'Owner Phone'	?*
Param.PhoneNumber	{Owner Phone:Lookup}
AttachContext	TRUE
Param.CallNotifyText	Blind transfer from {@UserName} about SR {Id}...
Param.RemoteConnectStr	{$RemoteConnectStr(Owner Phone:Lookup)} (In this example, Owner Phone is a field from the Service Request business component.)

Using Device Event to Enhance Screen Pop Performance

The Siebel CTI Connect communications driver, used as part of the Siebel CTI Connect product (which is based on Intel NetMerge), includes an event that provides a way to enhance CTI screen pop performance for your call center agents.

The EventAnswerCall event is triggered whenever the AnswerCall device command is invoked by the Siebel CTI Connect driver, such as when an agent answers a call using the communications toolbar.

To use this feature, create an event handler definition in your communications configuration that is based on the EventAnswerCall device event. This event handler invokes an event response that generates a screen pop and invokes the appropriate event logs.

The EventAnswerCall event occurs after the driver sends the AnswerCall command, but before the Siebel CTI Connect middleware sends the corresponding TpAnswered event. In this manner, extra time is available in which to generate the screen pop.

The event handler definition using EventAnswerCall causes the screen pop to be generated for the agent upon answering the call from the communications toolbar. This screen pop may occur before the call is actually connected. Event logging should still be based on the TpAnswered device event. See "Event Examples for Agents Answering Call from Communications Toolbar" on page 208.

For the case where the agent answers the call from the physical teleset instead of the communications toolbar, do not use the EventAnswerCall event. Instead, both the screen pop and event logging should be based on the TpAnswered device event. See "Event Examples for Agents Answering Call from Teleset" on page 209.

For additional information about handling inbound calls, see "Handling an Inbound Call Received by an Agent" on page 109.

Dependency for Multiple Event Handlers

With this feature, it is possible to configure event handlers where the completion of one event handler is dependent on the completion of another event handler. (Otherwise, the execution of event handlers is independent.)

For example, your call center may require an automatic call-answering function, such as to invoke the AnswerCall command using a script initially invoked by the InboundCall event handler.

As described, the AnswerCall command invokes the EventAnswerCall event. If you have defined an event handler to be triggered by EventAnswerCall, then your original InboundCall event handler will not complete until the EventAnswerCall event handler has completed. If the EventAnswerCall event handler performs some lengthy operation (such as to invoke another script), then it will delay completion of the InboundCall event handler from which the initial script was invoked.

Event Examples for Agents Answering Call from Communications Toolbar

The following examples show event handler, event response, and event log definitions based on receiving the EventAnswerCall device event. Use examples like these for when the agent answers the call using the communications toolbar:

```
[EventHandler:AnswerCallReceived]
    DeviceEvent = "EventAnswerCall"
    Response = "OnAnswerCallReceived"
    Filter.ANI = "*"
    Order = "1"

[EventResponse:OnAnswerCallReceived]
    QueryBusObj = "Contact"
    QueryBusComp = "Contact"
    QuerySpec = "'Work Phone #'='{ANI}'"
    SingleView = "Service Contact Detail View"
    MultiView = "Visible Contact List View"
    FindDialog = "Service Request"
    FindField.CSN = "Ask Caller"
    Log = "LogIncomingCallContactAnswerCall"
```

```
[EventLog:LogIncomingCallContactAnswerCall]
    WorkTrackingObj.PopView = "YES"

[EventHandler:InboundCallReceived-no-pop]
    DeviceEvent = "TpAnswered"
    Response = "InboundCallReceived-no-pop"
    Filter.ANI = "*"
    FilterSpec = "[$GetWorkItemAttr(SiebelWorkItemID, PopView)] = 'YES'"
    Order = "2"

[EventResponse:InboundCallReceived-no-pop]
    QueryBusObj = "Contact"
    QueryBusComp = "Contact"
    QuerySpec = "'Work Phone #'='{ANI}'"
    SingleLog = "LogIncomingCallContactFound"
    MultiLog = "LogIncomingCallMultiContactFound"
    Log = "LogIncomingCallContactNotFound"
```

Event Examples for Agents Answering Call from Teleset

The following examples show event handler and event response definitions based on receiving the TpAnswered device event. Use examples like these for when the agent answers the call using the physical teleset:

```
[EventHandler:InboundCallReceived-pop]
    DeviceEvent = "TpAnswered"
    Response = "InboundCallReceived-pop"
    Filter.ANI = "*"
    Order = "3"

[EventResponse:InboundCallReceived-pop]
    QueryBusObj = "Contact"
    QueryBusComp = "Contact"
    QuerySpec = "'Work Phone #'='{ANI}'"
    SingleView = "Service Contact Detail View"
    MultiView = "All Contacts across Organizations"
    FindDialog = "Service Request"
    FindField.CSN = "Ask Caller"
    SingleLog = "LogIncomingCallContactFound"
    MultiLog = "LogIncomingCallMultiContactFound"
    Log = "LogIncomingCallContactNotFound"
```

Using Push Keep Alive Driver for Session Connections

The Push Keep Alive communications driver provides a heartbeat message that can maintain connections for your communications sessions in some environments. You use this driver with Communications Session Manager, such as when you are using Siebel CTI, or using Siebel Email Response with Siebel Universal Queuing.

The Push Keep Alive driver helps to solve problems that may be experienced in some environments, where push communications sessions between the Application Object Manager and a user's Web browser are sometimes dropped. A connection may be dropped if messages have not been pushed to an agent for a particular length of time.

The Push Keep Alive driver sends a heartbeat message to the Application Object Manager, and each agent user's browser, at a regular interval. This heartbeat message allows the push connection applicable to each agent's communications toolbar not to be dropped in such cases. The heartbeat interval is specified using a driver parameter.

NOTE: When you use the Push Keep Alive driver, you *must* also set the ChannelCleanupTimer configuration parameter to make sure that abandoned connections for communications sessions are dropped promptly. For more information about the ChannelCleanupTimer configuration parameter, see "Parameters for Communications Configurations" on page 53. For information about how to set this parameter, see "Specifying Parameters for Communications Configurations" on page 52.

To use the Push Keep Alive driver

1. Start the Siebel application, such as Siebel Call Center, logging in as the system administrator (for example, as the user SADMIN).
2. Navigate to Administration - Communications > Communications Drivers and Profiles.
3. Create a new record in the Communications Drivers list and complete the fields as described in the following table.

Parameter	Value
Name	Enter `Push Keep Alive`
Channel Type	Select Push Keep Alive from the drop-down list.
Interactive	Select this check box.
Library Name	Enter `sscmimed`

4. Click the Driver Parameters tab and enter the values as described in the following table.

Parameter	Value
LogDebug	FALSE
LogFile	push_{@UserName}.log
MaxLogKb	128
PushKeepAliveTimer	0
ReleaseLogHandle	TRUE

5. Click the Profiles tab and enter a new record with a name that identifies the Profile with the Push Keep Alive communications driver. For example, enter `Push Keep Alive`.

6 In the Profile Parameter Overrides child applet, enter a record with the following values.

Parameter	Value
Name	Select PushKeepAliveTimer from the drop-down list.
Value	Enter the time interval (in seconds) between the Push Keep Alive driver sending messages. For example, enter 180 to have the driver send a heartbeat message every 180 seconds.

7 Click All Configurations in the application link bar and select your configuration in the displayed list.

8 Click the Profiles screen tab.

The list of profiles currently associated with your configuration appears.

9 Click New.

The Add Communications Profiles dialog box appears. This dialog box displays the list of currently available profiles.

10 Select Push Keep Alive, then click OK.

This associates the Push Keep Alive communications driver with your configuration. It means that a heartbeat message is sent at the interval that you specify (for example, every 180 seconds).

11 If you use the server component *Communications Configuration Manager* to cache your configuration, shut down and restart your Siebel Server to apply the changes made in steps Step 1 to Step 10. If you do not use *Communications Configuration Manager*, log out and log into the application for these changes to take effect. For more information about the Communications Configuration Manager, see "Administering Communications Configuration Manager" on page 255.

Simulating a Communications Environment

This section describes how to use the Communications Simulator to simulate basic capabilities for handling work items of different channel types.

The Communications Simulator can help you to understand how Siebel Communications Server works. It allows you, for example, to experiment with event responses such as those that generate screen pops or populate the customer dashboard.

The Communications Simulator simulates:

- Inbound calls
- Inbound call (received from a predictive dialer)
- Inbound email

Any changes to communications profiles, configuration parameters, agents, telesets, commands, or events that you make as part of using the Communications Simulator must be made using the applicable communications configuration.

NOTE: The Communications Simulator can simulate CTI voice call-handling only for communications configurations using profiles for the communications driver provided by Siebel Systems to support Intel NetMerge middleware.

Setting Up and Running the Communications Simulator

The Communications Simulator is included as part of the Siebel Business Applications, when communications modules are licensed. Users can access the Communications Simulator with any of the Siebel client types. Configuration varies somewhat for each client type.

For instructions and requirements for enabling and running the Communications Simulator, see "Enabling Session Communications and Simulation" on page 240.

Using the Communications Simulator

Using the Communications Simulator, you can begin trying out basic communications features, such as accessing functions through the communications toolbar, or viewing Siebel records based on data associated with an inbound call. These features are described in more detail in the rest of this section.

You can simulate communications activity and experiment with associated screen-pop behavior.

Communications-Simulation Keys

The following sets of simulation keys have been preconfigured in the Communications Simulator. Consult the file predefined communications configurations for additional simulation commands:

- SHIFT+F11 simulates an incoming new voice call. In this case, the call contains Caller ID data known as ANI (Automatic Number Identification). The caller exists in the Siebel Sample Database and has presumably called before. See "Simulated Inbound Call with Caller ID" on page 213.
- CTRL+SHIFT+F11 simulates an incoming new voice call, in which the caller either does not have Caller ID or is not in the Siebel Database. See "Simulated Inbound Call from an Unknown Caller" on page 214.
- SHIFT+F6 simulates an inbound campaign call, such as initiated by a predictive dialer. See "Inbound Campaign Call Simulation" on page 215.
- CTRL+F11 simulates an incoming new email work item.

Inbound Call Simulations

This section describes two inbound call scenarios.

Simulated Inbound Call with Caller ID

When you press SHIFT+F11 in the Communications Simulator, you hear a ring or a beep, and you see the Accept Work Item button blink on the communications toolbar. The sound depends on how Enable Sound is set in the Communications options of the User Preferences screen. These indicators signal an incoming call.

Clicking the Accept Work Item button on the toolbar to answer the call initiates the following:

- A screen pop displays contact information about the caller.
- An activity record is automatically created, of type Call - Inbound.

The Siebel application performs the screen pop using ANI. By default, the Communications Simulator matches the contact's phone number to the setting of the Param.ANI parameter for the call simulation command. Find a contact in the Sample Database and enter the phone number as the value for Param.ANI.

Table 40 and Table 41 show the values used in the simulation. These values are displayed and can be edited in the All Commands and All Command Data views in the Administration - Communications screen (when you are logged in as SADMIN).

Table 40. Command: SimCallFound

Parameter Name	Parameter Value
DeviceCommand	SimulateCall
Hidden	TRUE
HotKey	SHIFT+F11

Table 41. Command Data: SimCallFound

Parameter Name	Parameter Value
Param.ANI	*phone_number*

You can modify the value for Param.ANI to specify the phone number, and therefore the contact, that the Communications Simulator uses for the screen pop. Many contacts in the Sample Database have work phone numbers defined. Enter another contact's work phone number to select that contact during simulation.

You can also change the view that will be displayed for the screen pop:

- The event response parameter SingleView specifies the view to display if the telephone number of the inbound call matches a single contact record in the Siebel Database.
- The event response parameter MultiView specifies the view to display if the telephone number of the incoming call matches more than one contact record in the database.

The example event response in Table 42 shows values that are used in simulating screen pops.

Table 42. Event Response: OnInboundCallReceived

Parameter Name	Parameter Value
QueryBusObj	Contact
QueryBusComp	Contact
QuerySpec	'Work Phone #'='{ANI}'
SingleView	Service Contact Detail View
MultiView	All Contacts across Organizations
FindDialog	Service Request
FindField.Owner	Ask Caller

Two event logs are associated with this event response:

- An event log of type SingleLog, named LogIncomingCallContactFound
- An event log of type Log, named LogIncomingCallContactNotFound

Simulated Inbound Call from an Unknown Caller

When you press CTRL+SHIFT+F11 in the Communications Simulator, you hear a ring or a beep, depending on how the Enable Sound option is set, and you see the Accept Work Item button blink.

Click the Accept Work Item button on the communications toolbar to answer the call. No information about the caller is retrieved, because the caller's telephone number was not in the database; the Search Center appears.

Table 43 and Table 44 show values used for inbound call simulation. This operation is similar to that for the simulated inbound call with ANI. In this case, however, the phone number in the Param.ANI setting does not exist in the Sample Database.

Table 43. Command: SimCallNotFound

Parameter Name	Parameter Value
DeviceCommand	SimulateCall
Hidden	TRUE
HotKey	CTRL+SHIFT+F11

Table 44. Command Data: SimCallNotFound

Parameter Name	Parameter Value
Param.ANI	6504775000

Once the Siebel application determines that the phone number does not exist in the Sample Database, the Search Center appears on the right side of the screen. In the Search drop-down list, the Service Request option is highlighted.

You can change the option that is highlighted in the Search Center by changing the value for the FindDialog parameter. For details, see Table 42 on page 214.

Other possible values for the FindDialog parameter include Opportunity, Service Account, Corporate Contact, Service Product, and Consumer.

Inbound Campaign Call Simulation

When you press SHIFT+F6, the Accept Work Item button on the communications toolbar blinks. Click the Accept Work Item button to connect to the call. You will then get a screen pop that displays the campaign associated with the simulated call.

Accepting the call automatically creates an activity record of type Call - Inbound. When the call ends, the date and time are entered in the activity's Description field.

NOTE: Typically, a campaign call like this originates from a predictive dialer, and is considered as an outbound call—from the call center to the consumer. However, as it relates to the communications configuration, the call functions as an inbound call, due to how it arrives at the agent's communications toolbar.

Table 45 and Table 46 show values used for inbound campaign call simulation.

Table 45. Command: SimCampaignCall

Parameter Name	Parameter Value
DeviceCommand	SimulateCall
Hidden	TRUE
HotKey	SHIFT+F6

Table 46. Command Data: SimCampaignCall

Parameter Name	Parameter Value
Param.CampID	1-CQZ
Param.CampContactID	FAKE_ID

Using Business Services with Communications Server

This section describes some ways in which you can use Siebel business services with Siebel Communications Server.

Business services represent encapsulated functionality in the form of methods that can be called by Siebel application modules or by scripts within the Siebel application environment. Many types of Siebel Business Applications functionality, including Communications Server functionality, can be accessed as methods of Siebel-provided business services. Siebel applications running on all client types support Siebel business services.

This section describes using business services in two general ways:

- Access Siebel Communications Server functionality outside of the standard means offered by the Siebel application user interface. For example, Siebel Workflow invokes Communications Server business services, and Siebel VB and Siebel eScript scripts can invoke Communications Server business services.
- Integrate Siebel Communications Server events and commands with Siebel business services, in particular with custom business services or business services other than those provided for Communications Server.

Siebel Communications Server supports multiple business services that support the different functional areas of interactive session communications, outbound communications, and inbound communications.

The Outbound Communications Manager business service runs on the Siebel Server on which the Communications Outbound Manager server component is running.

For information about business services related to inbound communications, see *Siebel Email Response Administration Guide* and *Siebel Universal Queuing Administration Guide*.

For additional information about Siebel business services, see:

- *Siebel Business Process Designer Administration Guide*
- *Configuring Siebel Business Applications*
- *Siebel Developer's Reference*
- *Overview: Siebel Enterprise Application Integration* (and other books for Siebel EAI)

Invoking Communications Server Business Service Methods

Siebel Communications Server includes two business services that support session communications for agents:

- **Communications Client.** Supports the communications user interface features such as the communications toolbar and communications menu commands. This business service aggregates the Communications Session Manager business service. It is not involved with server-based communications functionality.
- **Communications Session Manager.** Provides an interface to session-based communications functionality at the server level. Through Server Request Broker and Server Request Processor, this business service communicates with the Communications Session Manager server component. It does not deal with the communications user interface.

  NOTE: Because the Communications Session Manager business service is aggregated by the Communications Client business service, you can access all Communications Session Manager business service methods from the Communications Client business service. For configuring communications events or commands, the communications toolbar, or the Application Object Manager, you *must* invoke all methods through the Communications Client business service.

The first two of these business services work together, and may be run on the Application Object Manager supporting instances of the Siebel Web Client. They run together with the channel manager, depending on how you have deployed interactive session communications. For more information, see "Enabling Session Communications and Simulation" on page 240.

Siebel Communications Server includes one business service that supports outbound communications:

- **Outbound Communications Manager.** Provides an interface to the outbound communications functionality of the Communications Outbound Manager server component.

Methods of Communications Server business services can be invoked from outside Communications Server. You can invoke a method from an applet, a script, a workflow, or another business service, or invoke it in some other way. Siebel Workflow and scripts for Siebel VB or Siebel eScript can invoke Communications Server business services.

For detailed information about Communications Server business services, methods, and arguments, see Appendix B, "Communications Server Business Services."

About Using Business Services with Events and Commands

This section describes scenarios for integrating business services with communications event and commands. These scenarios are applicable to a contact center using Siebel CTI and related modules, Siebel Email Response, and Siebel Universal Queuing. These scenarios are as follows:

- Invoking a business service method from an event handler
- Invoking a business service method from an event response

  A business service method can also be invoked from an event log, but this scenario is not described here.
- Invoking a communications command from a business service method
- Invoking a business service method from a communications command

Invoking a Command Through the Business Service Model

You can set up a custom business service that will invoke a command from a communications configuration. To set this up, you do the following:

- Create a communications command (and a corresponding command data definition) that is intended to be invoked from outside of Siebel Communications Server, and
- Modify or create an applet, script, or business service to invoke a particular communications command

An example is provided below for a make-call command and for a business service method that will invoke this command.

Table 47 and Table 48 show an example of a command that will be called by a custom business service. This example executes a MakeCall device command.

Table 47. Command: MakeCallFromCustomService

Parameter Name	Parameter Value
Description	Make Call from Custom Service
DeviceCommand	MakeCall
Hidden	TRUE

Table 48. Command Data: MakeCallFromCustomService

Parameter Name	Parameter Value
Param.PhoneNumber	{Call Recipient's Phone Number}
Param.DisplayText	{My Display Text}

The values for the Call Recipient's Phone Number and My Display Text will be passed from the custom business service to the command when the command is invoked.

A business service must be implemented that will invoke the command MakeCallFromCustomService. Communications Server retrieves the values of Callee Phone Number and My Display Text from the input arguments of the business service method, and assign these values to the command data parameters Param.PhoneNumber and Param.DisplayText. The device command MakeCall is executed, using the values for these parameters.

NOTE: By default, communications commands are invoked from business service methods associated with communications toolbar buttons. For more information, see "About Communications Toolbar Configuration" on page 151.

Invoking a Business Service Method from a Command

You can create or modify a command and a corresponding command data definition in the communications configuration to specify a Siebel business service and method to be invoked. To do this, you do the following:

- Use the command parameter ServiceMethod in a command to specify the name of the business service and method to invoke.
- Use the command parameter ServiceParam in the associated command data definition to specify argument names and values to pass to the business service method.

For more information about using these command parameters, see "Commands" on page 128 and "Command Data" on page 142.

In the example shown in Table 49 on page 220 and Table 50 on page 220, Communications Server invokes the method MyMakeCall of the business service MyMakeCallService, and passes arguments to the service method. The value of the ServiceParam.PhoneNumber parameter is from the @Phone macro; the value of the ServiceParam.AgentID parameter is from the @AgentId macro.

At runtime, commands that invoke business service methods are determined to be enabled or disabled by invoking from the applicable business service the method CanInvokeMethod("*method_name*"), where *method_name* is your business service method to be invoked. If TRUE is returned, the method can be invoked. If FALSE is returned, the method cannot be invoked and the command is disabled.

No DeviceCommand parameter is included in this example, because the communications command is not designed to call a function on the external communications system.

Table 49. Command: MakeCallInService

Parameter Name	Parameter Value
Description	Make Call In My Service
ServiceMethod	MyMakeCallService.MyMakeCall
Hidden	TRUE

Table 50. Command Data: MakeCallInService

Parameter Name	Parameter Value
ServiceParam.PhoneNumber	{@Phone}
ServiceParam.AgentID	{@AgentId}

Communications Server can invoke the methods of any business service, not only those, like this example, that pertain to communications.

NOTE: By default, several communications commands invoke business service methods. For examples, see the communications configuration data provided by Siebel Systems.

Invoking a Business Service Method from an Event Handler

You can create or modify an event handler in the communications configuration to specify a Siebel business service and method to be invoked. The business service method returns data to determine whether this event handler matches. In addition, it may perform other functions desired by the customer.

To configure this, you use the event parameters ServiceMethod and ServiceParam to specify the name of the business service and method to invoke, and to specify argument names and values to pass to the business service method. For more information about using these event parameters, see "Event Handlers" on page 104.

In the example shown in Table 51, Communications Server invokes the method MyMethod of the business service MyService, and passes arguments "attribute1" and "attribute2" to the method, using the values of the ServiceParam.Param1 and ServiceParam.Param2 parameters.

Table 51. Event Handler: OnInboundCallReceived

Parameter Name	Parameter Value
ServiceMethod	MyService.MyMethod
ServiceParam.Param1	{attribute1}
ServiceParam.Param2	{attribute1}

The example event handler has the Device Event field set to the appropriate device event for the driver, such as InboundCall for the Siebel CTI Connect driver.

You can use the ServiceMethod and ServiceParam parameters to supplement or replace filter mechanisms using the Filter or FilterSpec parameters. You can also use this method to execute custom code before executing a specified event response. The business method should set a parameter of Result to a value 1 or 0 (or TRUE or FALSE):

- 1 (TRUE) indicates that this event handler matches and is executed, and the associated event response is executed.
- 0 (FALSE) invalidates this event handler; the next event handler will then be evaluated.

Invoking a Business Service Method from an Event Response

You can create or modify an event response in the communications configuration to specify a Siebel business service and method to be invoked whenever this event response is invoked by an event handler.

To configure this, you use the event parameters ServiceMethod and ServiceParam to specify the name of the business service and method to invoke, and to specify argument names and values to pass to the business service method. For more information about using these event parameters, see "Event Responses" on page 110.

The business service method may also alter the event data fields that were associated with the work item, such as to add new fields.

In the example event response shown in Table 52 (for Siebel CTI Connect), the method MyMethod of the business service MyService is invoked to handle the event. The value of the ServiceParam.CallingDN parameter is from the ANI event data field; the value of the ServiceParam.Connection parameter is from the refId event data field.

Table 52. Event Response: OnInboundCallReceived

Parameter Name	Parameter Value
ServiceMethod	MyService.MyMethod
ServiceParam.CallingDN	{ANI}
ServiceParam.Connection	{refId}

When a business service method is invoked from an event response, the value for the output argument Continue will be either TRUE (1) or FALSE (0). If Continue is TRUE, event handling as specified in the event handler that called this event response will proceed (such as to generate a screen pop).

Integrating with Siebel Scripting Languages

Siebel VB and Siebel eScript scripts can be used in many ways to extend your communications-enabled Siebel applications.

For information about scripting methods from Siebel CTI release 6.*x*, see "Using Siebel Scripts from Release 6.x" on page 492.

You can use Siebel VB and Siebel eScript scripts in the following ways:

- Communications events or commands can invoke Siebel VB or Siebel eScript scripts:
  - Communications events can invoke Siebel VB or Siebel eScript scripts, for instance on the arrival of an incoming work item or upon releasing a work item.
  - Communications commands can invoke Siebel VB or Siebel eScript scripts.
- Siebel VB or Siebel eScript scripts can invoke communications commands (using business service methods) and access event data fields or work item attributes:
  - Siebel VB or Siebel eScript scripts can invoke communications commands. Commands such as transferring a work item can be invoked from Siebel VB–based menu items, buttons, or toolbar buttons.
  - Siebel VB or Siebel eScript scripts can access data attached to a work item, such as event data fields like ANI, DNIS, digits collected from an IVR system, or Siebel work item attributes.
  - Siebel VB or Siebel eScript scripts can access methods and arguments of Communications Server business services. For more information, see "Using Business Services with Communications Server" on page 216 and Appendix B, "Communications Server Business Services."

- Siebel VB or Siebel eScript scripts can access runtime data provided by the macros described in "Macros for Parameter Values" on page 182.

Siebel Communications Server integration with Siebel VB and Siebel eScript allows customers to write event handler functions and advanced communications commands in Siebel VB or Siebel eScript.

The methods of all Communications Server business server methods are exposed and can be called with any applicable arguments. All event data fields (such as ANI, DNIS, and so on) are accessible. A Siebel VB or Siebel eScript script can be invoked on any communications event.

A major use of scripting is to allow communications event handling (such as screen-pop logic). But you can also use it to take complete control of the communications technology: to obtain information about the current work item, to intercept events and perform the necessary handling, or to control communications work items (initiate work items, transfer work items, and so on).

NOTE: Scripts can be defined in the general section of the application object or in a custom business service. They will be compiled into the Siebel repository (.srf) file for the Siebel Application Object Manager.

Integrating Using Server and Browser Scripts

Siebel eScript scripts are generally supported only for the Object Manager layer, not for the browser layer. For Siebel Web Client, the Object Manager is on the Siebel Server (Application Object Manager, such as SCCObjMgr_enu for Siebel Call Center in an ENU environment).

However, Siebel eScript scripts defined on the browser layer can be supported indirectly by using both server scripts and the @InvokeSWECommand special command. See the example below for creating a communications command, a command for the Siebel Web Engine (SWE), and related server and browser scripts.

NOTE: Transferring data from a server script to a browser script is not supported, due to limitations of SWE commands.

For more information about the @InvokeSWECommand special command, see "Special Commands for Device Commands" on page 89.

Command Example

Table 53 and Table 54 show an example command and associated command data definition that invokes the AlertTest SWE command.

Table 53. Command: Alert

Parameter Name	Parameter Value
DeviceCommand	@InvokeSWECommand
CmdData	Alert

Table 54. Command Data: Alert

Parameter Name	Parameter Value
Param.SWECommand	AlertTest

SWE Command Example

In Siebel Tools, create a command named AlertTest with the properties shown in Table 55. This SWE command is executed by the communications command shown above, and in turn invokes the CTI_Test.Alert business service method.

Table 55. SWE Command Example

Property Name	Value
Name	AlertTest
Business Service	CTI_Test
Force Enable	Y
Method	Alert
Target	Browser

Server and Browser Script Examples

Define the following *server* script in the CTI_Test business service. The PreCanInvokeMethod server script in the CTI_Test business service allows the SWE command to be enabled that in turn will invoke the browser-layer scripting business service method Alert:

```
function Service_PreCanInvokeMethod (MethodName, &CanInvoke)
{
   var ReturnVal = ContinueOperation;
   if (MethodName == "Alert")
   {
      CanInvoke = "True";
      ReturnVal = CancelOperation;
```

```
   }
   return ReturnVal;
}
```

Define the following *browser* scripts in the CTI_Test business service. The PreInvokeMethod browser script invokes the Alert method to create an alert box in the browser.

```
function Service_PreCanInvokeMethod (methodName)
{
   if (methodName == "Alert")
      return true;
   else
      return ("ContinueOperation");
}

function Service_PreInvokeMethod (methodName, inputPropSet, outputPropSet)
{
   if (methodName == "Alert")
   {
      alert("CTI browser script test");
      return ("CancelOperation");
   }
   else
      return ("ContinueOperation");
}
```

In the previous steps, you have successfully created a new communications command named Alert. This command can be executed, for example, from the Communications submenu of the Tools menu, or from a keyboard shortcut. Another option is to further integrate this command with a server script, as illustrated in the following server script example:

```
function BrowserScriptTest()
{
   // Invoking Browser Script
   var ctibs = TheApplication().GetService("Communications Client");
   var ip = TheApplication().NewPropertySet();
   var op = TheApplication().NewPropertySet();
   ctibs.InvokeMethod("Alert", ip, op);
}
```

Integrating Using Server Scripts

Here are some examples of using business services and Siebel scripting with your communications configurations, such as for Siebel CTI Connect. These examples are for server-side scripting using Siebel eScript.

Event Response Example

Table 56 shows an example event response that invokes the server-layer scripting business service method CTI_Test.EvtBSGotoView. This event response retrieves work item data and generates a screen pop to the view Contact List View, with all contacts whose last name starts with the letter S.

Table 56. Event Response: OnInboundCallReceived

Parameter Name	Parameter Value
ServiceMethod	CTI_Test.EvtBSGotoView
ServiceParam.myWorkItemID	{SiebelWorkItemID}
ServiceParam.ANI	{ANI}

Command Example

Table 57 and Table 58 show an example communications command and associated command data that invokes the server-layer scripting business service method CTI_Test.CmdBSGotoView. This command makes an outbound phone call to the number the user specified in the communications toolbar's text input field.

Table 57. Command: MakeCallToPhone

Parameter Name	Parameter Value
ServiceMethod	CTI_Test.CmdBSGotoView
CmdData	MakeCallToPhone
OnEditControl	TRUE

Table 58. Command Data: MakeCallToPhone

Parameter Name	Parameter Value
ServiceParam.PhoneNumber	{@Phone:PhoneTypeLookup}
RequiredField.@Phone	?*
Param.CallNotifyText	Call from {@UserName}...

Server Script Examples

Define the following *server* scripts in the CTI_Test business service. These scripts are invoked by the event response and command described in "Event Response Example" and "Command Example":

```
function Service_PreCanInvokeMethod (MethodName, &CanInvoke)
{
   var ReturnVal = ContinueOperation;
   switch (MethodName)
   {
      case "EvtBSGotoView":
```

```
      case "CmdBSGotoView":
      CanInvoke = "True";
      ReturnVal = CancelOperation;
   }
   return (ReturnVal);
}

function Service_PreInvokeMethod (MethodName, Inputs, Outputs)
{
   var ReturnVal = ContinueOperation;
   switch (MethodName)
   {
      case "EvtBSGotoView":
         EvtBSGotoView(Inputs, Outputs);
         ReturnVal = CancelOperation;
         break;

      case "CmdBSGotoView":
          CmdBSGotoView(Inputs, Outputs);
          ReturnVal = CancelOperation;
          break;
   }
   return (ReturnVal);
}

function EvtBSGotoView(Inputs, Outputs)
{
   // Getting input arguments from Event Response parameters
   var itemID = Inputs.GetProperty("myWorkItemID");
   var ANI = Inputs.GetProperty("ANI");

   // Invoking Business Service "Communications Session Manager" method
   var ctibs = TheApplication().GetService("Communications Session Manager");
   var ip = TheApplication().NewPropertySet();
   var op = TheApplication().NewPropertySet();
   ip.SetProperty("WorkItemID", itemID);
   ctibs.InvokeMethod("GetWorkItemInfo", ip, op);

   // Generate a screen pop to "Contact List View" with
   // all the contacts whose last name start with letter "S"
   var boGlobal = TheApplication().GetBusObject("Contact");
   var bcContact = boGlobal.GetBusComp("Contact");
   bcContact.SetViewMode(AllView);
   bcContact.ClearToQuery();
   bcContact.SetSearchSpec("Last Name", "S*");
   bcContact.ExecuteQuery(ForwardBackward);
   TheApplication().GotoView("Contact List View", boGlobal);
}

function CmdBSGotoView(Inputs, Outputs)
{
   // Getting input arguments from Command Data parameters
   var number = Inputs.GetProperty("PhoneNumber");

   // Invoking Business Service "Communications Client" method
   var ctibs = TheApplication().GetService("Communications Client");
   var ip = TheApplication().NewPropertySet();
   var op = TheApplication().NewPropertySet();
   ip.SetProperty("PhoneNumber", number);
   ip.SetProperty("ProfileName", "Siebel CTI Connect for San Mateo");
   ctibs.InvokeMethod("MakeCall", ip, op);
}
```

More Information About Scripting

For more information about Siebel scripting, see the following documents:

- *Siebel Object Interfaces Reference*
- *Siebel eScript Language Reference*
- *Siebel VB Language Reference*
- *Configuring Siebel Business Applications*
- *Siebel Developer's Reference*
- *Siebel Installation Guide* for the operating system you are using
- *Siebel System Administration Guide*

Integrating with Siebel SmartScript

You can integrate Siebel Communications Server with Siebel SmartScript. For more information about SmartScript, see *Siebel SmartScript Administration Guide*.

Invoking SmartScript Through Communications Server

Siebel Communications Server can invoke Siebel SmartScript in order to execute SmartScripts. To enable this behavior, you define parameters for the appropriate event response, using the views in the Administration - Communications screen. If the event response is executed in response to a corresponding event handler matching the call data parameters, the script will be launched.

The following parameter and value must be added to the event response in order to enable the link between Communications Server and Siebel SmartScript:

- Parameter name: SmartScript.ScriptName
- Parameter value: *Your_script_name*

Any parameters received that are prefixed with "SmartScript" will be passed to the SmartScript that is invoked. The available parameters are:

- SmartScript.ScriptName
- SmartScript.ScriptId
- SmartScript.LanguageCode
- SmartScript.CampaignId
- SmartScript.CampContactId
- SmartScript.ContactId

Either ScriptName or ScriptId must be specified in order to start a specific SmartScript. Otherwise, the agent will be prompted, through the Choose Script dialog box, for the name of the script to run. In addition, either Siebel VB or Siebel eScript must be invoked to set the correct focus for the applicable business component.

For more information about defining event response parameters, see "Event Responses" on page 110.

Example Events to Invoke SmartScript Script

Table 59 shows an example event response that invokes a Siebel SmartScript script. The event handler that invokes this event response has the Device Event field set to the appropriate device event for the driver, such as TpAnswered for Siebel CTI Connect.

Table 59. Event Response: OnInboundCallReceived

Parameter Name	Parameter Value
SmartScript.ScriptName	Customer Service

Displaying Communications Parameter Data in SmartScript

If a SmartScript is invoked from Communications Server, call data parameters are also available to be passed to SmartScript. These parameters, which have the same names in SmartScript as in the communications configuration data, can be accessed through either Siebel VB or Siebel eScript by using the GetParameter function against the SmartScript object. The parameters can be included among the variables displayed in the customer dashboard by using the prefix CTI.

For example, a variable using the GetParameter function may be defined as follows:

```
var s = GetParameter("CTI.ANI")
```

For more information about variables and programming for SmartScript, see *Siebel SmartScript Administration Guide*.

Integrating with the Customer Dashboard

The customer dashboard, located near the top of the application window for Siebel Call Center and other applications, contains fields with customer-related data for an agent to view. The customer dashboard typically displays contact information, but can be configured in Siebel Tools to display other fields.

For more information about configuring the customer dashboard using Siebel Tools, see *Configuring Siebel Business Applications*.

Customer dashboard fields can be populated, or cleared, automatically upon the execution of a communications event or command. For example, the fields may be populated from an event log when an inbound work item arrives, or cleared from an event log if the caller is not found in the database.

The fields in the dashboard are populated with customer data by invoking a business service and method and passing values from business component fields or event data fields, such as the ANI phone number of a caller. If a match is produced, the dashboard fields can be populated.

Example Events for Updating and Clearing Customer Dashboard

For example, the following sequence of event handler, event response, and event logs causes the dashboard to be populated with data, or cleared.

In these examples, when a single matching contact record is found, the dashboard is populated. When no contact records are found, or multiple records are found, the dashboard is cleared.

Adjust your event definitions as necessary, such as if you are using a different business component than Contact:

```
[EventHandler:InboundCallReceived]
    Filter.ANI = "*"
    Profile = ""
    Comments = ""
    Order = "1"
    Response = "OnInboundCallReceived"
    DeviceEvent = "TpAnswered"

[EventResponse:OnInboundCallReceived]
    MultiView = "All Contacts across Organizations"
    QueryBusComp = "Contact"
    QueryBusObj = "Contact"
    QuerySpec = "'Work Phone #'='650477{ANI}'"
    SingleView = "Service Contact Detail View"
    SingleLog = "LogIncomingCallContactFound"
    Log = "LogIncomingCallContactNotFound"
    MultiLog = "LogIncomingCallMultiContactFound"
    Comments = ""
```

If contact data is found, the event log below updates the customer dashboard:

```
[EventLog:LogIncomingCallContactFound]
    AfterWork.'ACD Call Duration' = "{@WorkDuration}"
    AfterWork.'Done' = "{@Now}"
    AfterWork.'Planned Completion' = "{@Now}"
    BusComp = "Action"
    BusObj = "Contact"
    Display = "FALSE"
    LogField.'Account Id' = "{Contact.'Account Id'}"
    LogField.'Call Id' = "{refId}"
    LogField.'Contact Id (Thin)' = "{Contact.Id}"
    LogField.'Planned' = "{@WorkStartTime}"
    LogField.'Started' = "{@WorkStartTime}"
    LogField.Description = "Inbound call"
    LogField.Type = "Call - Inbound"
    ServiceMethod = "Persistent Customer Dashboard.Update Dashboard from CTI"
    ServiceParam.Field = "Id"
    ServiceParam.Value = "{Contact.Id}"
    WorkTrackingObj.ContactId = "{Contact.Id}"
    Comments = ""
```

If no contact record is found, the event log below clears the customer dashboard:

```
[EventLog:LogIncomingCallContactNotFound]
    AfterWork.'ACD Call Duration' = "{@WorkDuration}"
    BusComp = "Action"
    BusObj = "Contact"
    LogField.'Call Id' = "{refId}"
    LogField.Description = "Unknown Caller({ANI})"
    LogField.Type = "Call - Inbound"
    ServiceMethod = "Persistent Customer Dashboard.CleanDashBoard_UI"
    Comments = ""
```

If multiple contact records are found, the event log below clears the customer dashboard:

```
[EventLog:LogIncomingCallMultiContactFound]
    AfterWork.'ACD Call Duration' = "{@WorkDuration}"
    BusComp = "Action"
    BusObj = "Contact"
    LogField.'Call Id' = "{refId}"
    LogField.Description = "Inbound call({ANI})"
    LogField.Type = "Call - Inbound"
    ServiceMethod = "Persistent Customer Dashboard.CleanDashBoard_UI"
    Comments = ""
```

Generating Communications Reports

Communications administrators and call center managers can generate reports tracking inbound and outbound activity for Siebel Communications Server.

For information about configuring additional reports, see *Siebel Reports Administration Guide*.

To generate a communications report

1. Navigate to Administration - Communications > Reports.
2. As instructed, click the Reports button in the application toolbar (not in the communications toolbar).
3. Click to select one of the following preconfigured call-center reports, or another report you have previously configured:
   - Call Center Volume
   - Call Center Volume (Inbound)
   - Call Center Volume (Outbound)

   The selected report is generated for you to view or print.

Requirement for Defining Fields in Event Logs

Generating the reports identified in this section requires that event log definitions in your communications configuration include parameters that specify field names as shown in the examples in Table 60, Table 61 on page 232, and Table 62 on page 233. The field names are specified using the parameters LogField and AfterWork.

Table 60. Event Log: LogIncomingCallContactFound

Parameter Name	Parameter Value
Display	FALSE
BusObj	Contact
BusComp	Action
LogField.Type	Call - Inbound
LogField.'Account Id'	{Contact.'Account Id'}
LogField.'Contact Id (Thin)'	{Contact.Id}
LogField.Description	Inbound call
LogField.'Call Id'	{ConnID}
LogField.'Planned'	{@WorkStartTime}
LogField.'Started'	{@WorkStartTime}
AfterWork.'Planned Completion'	{@Now}
AfterWork.'Done'	{@Now}
AfterWork.'ACD Call Duration'	{@WorkDuration}
ServiceMethod	Persistent Customer Dashboard.Update Dashboard from CTI
ServiceParam.Field	Id
ServiceParam.Value	{Contact.Id}
WorkTrackingObj.ContactId	{Contact.Id}

Table 61. Event Log: LogIncomingCallContactNotFound

Parameter Name	Parameter Value
BusObj	Contact
BusComp	Action
LogField.Type	Call - Inbound
LogField.Description	Unknown Caller({ANI})
LogField.'Call Id'	{ConnID}

Table 61. Event Log: LogIncomingCallContactNotFound

Parameter Name	Parameter Value
AfterWork.'ACD Call Duration'	{@WorkDuration}
ServiceMethod	Persistent Customer Dashboard.CleanDashBoard_UI

Table 62. Event Log: LogIncomingCallMultiContactFound

Parameter Name	Parameter Value
BusObj	Contact
BusComp	Action
LogField.Type	Call - Inbound
LogField.Description	Inbound call({ANI})
LogField.'Call Id'	{ConnID}
AfterWork.'ACD Call Duration'	{@WorkDuration}
ServiceMethod	Persistent Customer Dashboard.CleanDashBoard_UI

Viewing Communications Status Data

System administrators and call-center administrators can display status data about agents who are engaged in communications activity in Siebel applications. Status data is displayed in:

- All Active Agent Status view
- All Channel Items view

Viewing Agent Status Data

You can use the All Active Agent Status view to view status data for each agent who is currently logged in and using a Siebel application configured to support session communications. For each agent, the view displays:

- The agent's login, agent ID, and first and last name
- The communications configuration that is active for the agent
- The teleset the agent is using, for voice agents
- Currently active work items for the agent, for each channel
- The agent's state (such as Ready, Not Ready) for each channel for the agent's configuration

To view agent status data

1. Navigate to Administration - Communications > All Active Agent Status.
2. Click Refresh to update the displayed data.

NOTE: The All Active Agent Status view is always updated by the system, and does not require specific parameter settings or event log definitions.

Viewing Channel Status Data

You can use the All Channel Items view to view status data about current communications work items for agents. For each work item, the view displays the:

- Work item's channel, such as voice, email, and so on
- Channel target address
- Time the work item started
- Agent's Siebel login, agent log, and agent password

NOTE: For each work item, the value displayed in the Start Time field is accurate only to the extent that the system time is synchronized between the various computers running instances of Communications Session Manager or Application Object Manager.

To view channel status data

1. Navigate to Administration - Communications > All Channel Items.
2. Click Refresh to update the displayed data.

Modifying How Channel Status Data Is Displayed

Channel status data is logged for display in the All Channel Items view only if the UpdateChannelStatusTable configuration parameter is set to TRUE. In the Siebel-provided communications configurations, this parameter is set to TRUE. For more information, see "Specifying Parameters for Communications Configurations" on page 52.

NOTE: Records in the All Channel Items view reflect current activity only. If any records persist in this view for past work items, then you may need to delete such records manually in this view.

Distributing Status Data to Agents

Call-center administrators may want to distribute call-center statistics from the switch or CTI middleware to call-center agents for display in the message broadcast area of the Siebel client.

For example, it may be useful for an agent to know how long a call has been in an ACD queue before taking the call, how many calls are currently in an ACD queue, or the average talk time for a call.

Consult the documentation for your switch or CTI middleware for information about how call-center statistics can be retrieved.

You configure message broadcasting in the Message Broadcasts view, located in the Administration - Communications screen. For more information, see *Applications Administration Guide*.

8 Administering Siebel Communications Server

This chapter describes how to configure and run Siebel Communications Server components, including how to enable session communications and simulation for your applications. It includes the following topics:

Siebel Server Requirements for Communications Server

In order for you to be able to use Siebel Communications Server within your enterprise, note the following basic requirements relating to Siebel Server:

- The Siebel Server installation must include the Communications Management (CommMgmt) component group.
- After installation, the Communications Management component group must be enabled.

The rest of this section describes the Siebel Communications Server components and describes additional Siebel Server requirements.

NOTE: For more information about running, configuring, or monitoring Siebel Server components, see *Siebel System Administration Guide*.

Server Components for Communications Server

The Communications Management component group includes these server components:

- **Communications Session Manager (CommSessionMgr).** Supports multichannel user-interactive sessions for agents using the communications toolbar for voice, email, or other types of work items. For details, see "Enabling Session Communications and Simulation" on page 240 and "Administering Communications Session Manager" on page 254.

- **Communications Configuration Manager (CommConfigMgr).** Loads and caches communications configuration data, for agents using functions supported by the Communications Session Manager. For details, see "Administering Communications Configuration Manager" on page 255.
- **Communications Inbound Receiver (CommInboundRcvr).** Receives and queues inbound work items, and queues them for processing by Communications Inbound Processor. Work items may include email messages (for Siebel Email Response) or voice work items that are to be routed using Siebel Universal Queuing (for Siebel CTI).
  - For nonreal-time work items, such as email messages for most deployments of Siebel Email Response, Communications Inbound Receiver queues work items it has received for further processing by Communications Inbound Processor.
  - For real-time work items, such as phone calls for Siebel CTI or email messages for some deployments of Siebel Email Response, Communications Inbound Receiver processes work items it has received. Communications Inbound Processor is not used.

  For details, see "Administering Communications Inbound Receiver" on page 256.
- **Communications Inbound Processor (CommInboundProcessor).** Processes inbound work items that were queued by Communications Inbound Receiver. For details, see "Administering Communications Inbound Processor" on page 260.
- **Communications Outbound Manager (CommOutboundMgr).** Processes outbound communications, for email, fax, or page channels. Supports communication requests, whether directly or through Siebel Workflow. Also supports outbound capabilities for Siebel Email Response, Send Email, and Send Fax commands. For details, see "Administering Communications Outbound Manager" on page 261.
- **Page Manager (PageMgr).** Used to send pages by the Send Page command. Also used by some workflow processes in Siebel Workflow. For more information about setting up and using Page Manager, see *Siebel Business Process Designer Administration Guide*.
- **Email Manager (MailMgr).** Used to send email by some workflow processes in Siebel Workflow. For more information about setting up and using Email Manager, see *Siebel Business Process Designer Administration Guide*.
- **Smart Answer Manager (SmartAnswer).** Supports the Siebel Smart Answer product option. For more information, see *Siebel Smart Answer Administration Guide* and *Siebel Email Response Administration Guide*.
- **Message Broadcast Manager (MsgBroadcastMgr).** Supports the message broadcasting feature. For more information, see *Applications Administration Guide*.

Running Communications Server in Heterogeneous Server Environments

Siebel Communications Server supports heterogeneous server environments. Because Communications Server components such as Communications Session Manager load driver library files, the applicable components must be enabled on the same platform that the driver was developed for. For heterogeneous operation, the Application Object Manager can run in parallel on a different platform.

For more information, see *System Requirements and Supported Platforms* on Siebel SupportWeb, and see also "About Communications Drivers and Profiles" on page 35.

Using Siebel Server Load Balancing with Communications Server

If you are using Siebel Server Load Balancing to implement load balancing among multiple instances of Siebel Server, then, for each Siebel Server participating in load balancing, you must:

- Enable the Communications Management component group, as noted, and enable or start any applicable Communications Server components.

If you are also using Communications Session Manager, then you must:

- Set the Enable Communication parameter to TRUE for each Application Object Manager instance. For more information about enabling communications, see "Enabling Session Communications and Simulation" on page 240.

NOTE: Siebel Server Load Balancing does not load balance requests for Siebel Communications Server. Rather, in this case the Application Object Manager from which a Communications Server request is made is load-balanced, and requires that the necessary Communications Server components be enabled and running.

For more information about installing and configuring Siebel Server and Siebel Server Load Balancing, see *Siebel Installation Guide* for the operating system you are using and *Siebel System Administration Guide*.

Synchronizing Batch-Mode Server Components

Communications Inbound Receiver, Communications Inbound Processor, and Communications Outbound Manager are batch-mode components. You need to synchronize batch-mode server components between the Siebel Gateway Name Server and the Siebel Database whenever you:

- Create new server components
- Create new component jobs
- Modify existing server components
- Delete server components
- Enable server components

- Disable server components

For more information about synchronizing batch-mode server components, see *Siebel System Administration Guide*.

Enabling Session Communications and Simulation

Interactive session-based communications can be enabled for employees who use the Siebel Web Client or the Siebel Developer Web Client (the Siebel Mobile Web Client in connected mode). Optionally, you can also configure the Communications Simulator for your agents.

NOTE: Siebel Systems support for the Siebel Developer Web Client is restricted to administration, development, and troubleshooting usage scenarios only. Siebel Systems does not support the deployment of this client to end users.

Session-based Communications Server functionality can be accessed either through server components or business services, depending on your Siebel client deployment choices.

NOTE: This section applies only to enabling communications activities that use the communications toolbar or menu commands for handling voice and email channels for interactive communications.

For information about using the communications toolbar and menu items, see "Using the Communications Toolbar" on page 314 and "Using Communications Menu Commands" on page 321.

For information about the communications architecture, see "About Communications Server Architecture" on page 20.

About Communications Session Modes

Supported modes of operation for session communications are as follows:

- **Siebel Web Client.** Server-based communications session. (Most customers are likely to use this mode.)
- **Siebel Developer Web Client.** Two modes are supported:
  - Server-based communications session
  - Local communications session

Server-based communications sessions use the Communications Session Manager server component. Local communications sessions do not use this component, but rather perform communications processing locally, similar to Siebel CTI release 6.*x*.

The Communications Client and Communications Session Manager business services run on the Application Object Manager for Siebel Web Client deployments. These business services run locally for Siebel Developer Web Client deployments.

To enable communications, you configure the Siebel Web Client through the Application Object Manager. You configure Siebel Developer Web Client through the configuration file, such as uagent.cfg for Siebel Call Center.

About Communications Simulator

You can configure communications simulation, for any type of Siebel client, for individual users or groups of users. In addition to the client scenarios described earlier in this section for enabling communications, the Communications Simulator also supports the Siebel Mobile Web Client, connected to a local database.

The Communications Simulator enables you to test parts of your communications configuration, such as event and command definitions. It can also help you to train agents in handling communications work items in the Siebel applications.

NOTE: The Communications Simulator does not simulate outbound email, such as may be sent using the Send Email button on the communications toolbar. (The Send Email button is available through the Initiate Work Item button.)

For details, see "Enabling Communications Simulation" on page 252. See also "Simulating a Communications Environment" on page 211.

Prerequisites for Enabling Session Communications

In order for agents to be able to use communications features, including simulation, you must set up your communications environment. For more information, see "Process of Configuring Communications Server" on page 28. In particular:

- Siebel communications-related products (such as Siebel CTI, Siebel Email Response, or Siebel Universal Queuing) must be licensed, configured, and available. Applicable third-party communications systems also must be configured and available.
- For scenarios requiring the Communications Session Manager or Communications Configuration Manager server component, these Siebel Server components must be enabled and running. Also, they must be available for the instances of the Application Object Manager that connect to them for each agent's communications session.
- Call center users must have been defined as agents within a communications configuration, as described in "Specifying Agents" on page 59. (Demo users are predefined in the sample communications configuration data.)
- For CTI (voice-call handling), telesets and extensions must be defined, and the telesets must be associated with actual users who will run the Siebel application.

  Alternatively, demo telesets and extensions must be defined and associated with demo users associated with Demo applications that support the Communications Simulator. In the Sample Database, demo users such as CCHENG are associated with demo telesets in order to support communications simulation.

  The Demo applications are the shortcuts that start with the word *Demo* in the Siebel application program group for a Siebel Mobile Web Client installation. These shortcuts start the indicated application, such as Siebel Call Center Demo, in a demonstration-only mode and log in to the Sample Database as a demonstration user.

NOTE: Activating the communications toolbar is a prerequisite for being able to access the Communications Simulator. For details, see "Enabling Communications Simulation" on page 252.

Parameters for Application Object Manager and Siebel Developer Web Client

This section describes the parameters for the Application Object Manager (for Siebel Web Client) and Siebel Developer Web Client. These parameters are described in Table 63 on page 243.

NOTE: Siebel Systems does not support the Siebel Developer Web Client in production environments. It is recommended that you restrict the use of this client to development and test environments. Support for this client in these environments is restricted.

The type of client that you use determines how the parameters are specified:

- For deployments of the Siebel Web Client, parameters are specified as server component parameters for the Application Object Manager that is to support communications users.
- For deployments of the Siebel Developer Web Client (where applicable), parameters are specified in the [Communication] section of the configuration file, such as uagent.cfg for Siebel Call Center, for each communications user's Siebel client.

NOTE: For the parameters listed in Table 63 on page 243, if the parameter value is changed while the Application Object Manager is running, the changed value will affect communications sessions for all users who log in subsequently.

Table 63. Object Manager and Siebel Developer Web Client Parameters for Communications Sessions

Parameter Name	Display Name	Default Value	Description
CommConfigCache	Communication Configuration Cache	FALSE	Enables or disables the caching of communications configuration data on the Application Object Manager. **NOTE:** This parameter applies to the Siebel Web Client only, not to the Siebel Developer Web Client. ■ When this parameter is FALSE, communications configuration data is loaded separately for each agent session, with no caching on the Object Manager. ■ When this parameter is TRUE, the Application Object Manager loads and caches communications configuration data. When CommConfigCache is TRUE, after the first agent logs in, subsequent agent logins for the same configuration will download the configuration data from the cache. This speeds up the login process and reduces memory usage per agent session. This parameter may be used with or without the Communications Configuration Manager component and the CommConfigManager parameter. See also the description of the CommConfigManager parameter.

Table 63. Object Manager and Siebel Developer Web Client Parameters for Communications Sessions

Parameter Name	Display Name	Default Value	Description
CommConfigManager	Communication Configuration Manager	FALSE	Enables or disables use of the Communications Configuration Manager (CommConfigMgr) for loading and caching communications configuration data to reduce configuration downloads. ■ When this parameter is FALSE, communications configuration data is loaded separately for each agent session, with no caching. ■ When this parameter is TRUE, Communications Configuration Manager is employed to load and cache communications configuration data. After the first agent logs in, subsequent agents using the same configuration will download the configuration data from the cache, thereby speeding up the login process. For more information, see "Administering Communications Configuration Manager" on page 255.
CommConfigManager Name	Communication Configuration Manager Name	CommConfigMgr	Specifies the name of the Communications Configuration Manager server component to use when the parameter CommConfigManager is TRUE. A global deployment may, for example, use different components for different languages within the same enterprise.

Table 63. Object Manager and Siebel Developer Web Client Parameters for Communications Sessions

Parameter Name	Display Name	Default Value	Description
CommEnable	Enable Communication	FALSE	Enables or disables session-based communications for agents. ■ When this parameter is TRUE, the communications toolbar and menu elements are displayed and activated for agents who are associated with a configuration. ■ When this parameter is FALSE, the communications toolbar and menu elements are not displayed. If an agent is not associated with a configuration, the communications toolbar is not displayed, even if CommEnable is TRUE. Enabling communications has many additional requirements, which are described in Chapter 4, "Configuring Session Communications."

Table 63. Object Manager and Siebel Developer Web Client Parameters for Communications Sessions

Parameter Name	Display Name	Default Value	Description
CommLocalDriver	Local Communication Driver	FALSE (as component parameter) TRUE (as configuration file parameter for Siebel Developer Web Client)	Specifies on which machine interactive communications drivers are loaded by the channel manager for each agent communications session: ■ For either the Siebel Web Client or the Siebel Developer Web Client: when this parameter is FALSE, interactive drivers are loaded on the Siebel Server machine where Communications Session Manager is running. ■ For the Siebel Developer Web Client: if this parameter is TRUE in the configuration file, interactive drivers are loaded on each agent's Siebel client. ■ For the Siebel Web Client: if this parameter is TRUE on the Application Object Manager, interactive drivers are loaded on the Application Object Manager. For more information, see "Administering Communications Session Manager" on page 254.
CommLogDebug	Log Debug Message	FALSE	Specifies whether the log file specified with the CommLogFile parameter should contain extra detail.

Table 63. Object Manager and Siebel Developer Web Client Parameters for Communications Sessions

Parameter Name	Display Name	Default Value	Description
CommLogFile	Log File	SComm.log	Specifies the name of the log file to create for each agent's communications session. For example, where the parameter is set to SComm.log, files named SComm_*user*.log are created, where *user* is each agent's Siebel login name. These files are written in the log subdirectory of the Siebel Server or Siebel Developer Web Client installation directory.
CommMaxLogKB	Maximum Log Size (KB)	1024	Specifies the maximum size, in Kilobytes, of log files specified using CommLogFile.

Table 63. Object Manager and Siebel Developer Web Client Parameters for Communications Sessions

Parameter Name	Display Name	Default Value	Description
CommMaxMsgQ	Maximum Message Queue	64	The maximum number of messages to be queued for the agent's communications toolbar. If the number of messages exceeds the maximum, the oldest message is discarded. For example, with the default value of 64, the most recent 64 messages are queued. If a 65th message arrives, the oldest message in the queue is discarded in order for the new message to be received. It is recommended that you use the default setting unless you require changing it. If you set this value too low, older messages might be deleted in some cases, such as in a slow network environment. If you set this value too high, server resources may be overused, by storing many messages for agents who experience browser problems that interrupt the session. The behavior is also subject to the general session time-out defined for Siebel Web Client deployments. If an agent's session times out, all toolbar messages for the session are cleared.

Table 63. Object Manager and Siebel Developer Web Client Parameters for Communications Sessions

Parameter Name	Display Name	Default Value	Description
CommReleaseLogHandle	Release Log Handle	TRUE	Indicates that the log file handle for each agent is released periodically, after logs have been generated. The default setting of TRUE provides better performance. When CommReleaseLogHandle is FALSE, some log files may not be generated if the number of concurrent users exceeds the file-handle capacity provided by the operating system.
CommReqTimeout	Request Timeout	600	Specifies the number of seconds to wait for a response from the Communications Session Manager for a particular communications interaction.
CommSimulate	Simulate Communication	FALSE	Enables or disables the Communications Simulator. ■ When this parameter is TRUE, communications simulation is enabled for agents. ■ When this parameter is FALSE, communications simulation is disabled for agents. The Communications Simulator can be enabled in other ways as well. Additional requirements are described in "Enabling Communications Simulation" on page 252.

Parameters for Communications Session Manager

Parameters relating to log files can be specified as server component parameters for Communications Session Manager. These parameters are described in Table 64 on page 250.

NOTE: For the parameters listed in Table 64 on page 250, if the parameter value is changed while the Communications Session Manager is running, the changed value will affect communications sessions for all users who log in subsequently.

Table 64. Communications Session Manager Parameters

Parameter Name	Display Name	Default Value	Description
LogDebug	Log Debug Message	FALSE	Specifies whether the log file specified with the LogFile parameter should contain extra detail.
LogFile	Log File	SComm.log	Specifies the name of the log file to create for each agent's communications session. For example, where the parameter is set to SComm.log, files named SComm_*user*.log are created, where *user* is each agent's Siebel login name. These files are written in the log subdirectory of the Siebel Server installation directory.
MaxLogKB	Maximum Log Size (KB)	1024	Specifies the maximum size, in Kilobytes, of log files specified using LogFile.
ReleaseLogHandle	Release Log Handle	TRUE	Indicates that the log file handle for each agent is released periodically, after logs have been generated. The default setting of TRUE provides better performance. When ReleaseLogHandle is FALSE, some log files may not be generated if the number of concurrent users exceeds the file-handle capacity provided by the operating system.

Enabling Communications Sessions for Siebel Web Client

End users can access communications using the Siebel Web Client by connecting the Web browser to an Applications Object Manager for which server component parameter values are set as described in this section.

For more information about setting server component parameter values, see *Siebel System Administration Guide*.

To enable or disable session communications for Siebel Web Client users

- On each Application Object Manager that is to support users for whom session communications must be available, set the server component parameter CommEnable (Enable Communication) to TRUE. Alternatively, to disable communications, set it to FALSE. To set this parameter, do *one* of the following:
  - Navigate to Administration - Server Configuration > Servers, specify the Siebel Server, and select Call Center Object Manager (for example) in the Components list. Set a value for the Enable Communication parameter in the Parameters list (for this component).
  - On the command line, run Siebel Server Manager, specifying your Siebel Gateway Name Server, Siebel Enterprise Server, and Siebel Server, as appropriate. In Server Manager, enter (on one line) a command like the following, as appropriate for your Application Object Manager component:

    ```
    change parameter CommEnable=true for component SCCObjMgr_enu
    ```

After you set the CommEnable parameter value, the change will take effect for each subsequent user login. Optionally, you may request existing users to log out and log in again.

For more information about using Siebel Server Manager (GUI or command line), see *Siebel System Administration Guide*.

Enabling Communications Sessions for Siebel Developer Web Client

This section describes how you can enable communications using the Siebel Developer Web Client—the Siebel Mobile Web Client in connected mode. Enabling communications is subject to your having the appropriate license keys. Siebel Systems does not support the Siebel Developer Web Client in production environments. It is recommended that you restrict the use of this client to development and test environments. Support for this client in these environments is restricted.

NOTE: If simulation is also used, this section also applies to using the Siebel Mobile or Developer Web Client with the Sample Database.

Enabling Communications Using Command-Line Option

You can enable communications in the Siebel Developer Web Client by including the command-line option /ct at the end of the command line, such as in the application shortcut. Any of the Siebel Business Applications can support this command-line option.

Enabling Communications by Modifying Configuration File

You can enable communications in the Siebel Developer Web Client by setting the following parameter in the [Communication] section of the application's configuration file, such as uagent.cfg for Siebel Call Center:

```
CommEnable=TRUE
```

Enabling Communications Simulation

This section describes how to enable the Communications Simulator for end users. You can simulate communications for voice and email channels.

An end user can access communications simulation using any of the Siebel clients, depending on how simulation is enabled. Users commonly access simulation in the Siebel Mobile Web Client, using the Sample Database.

For information about enabling session communications, which is a prerequisite for communications simulation, see "Prerequisites for Enabling Session Communications" on page 241.

For information about end-user operations using communications simulation, see "Simulating a Communications Environment" on page 211.

NOTE: In order to test communications simulation, make sure that any configuration in which you are defining simulation functionality is the primary configuration for the applicable users.

Role of the Simulate Parameter

Enabling communications simulation as described in this section automatically sets the value of the Simulate driver parameter to TRUE, in the memory for each agent session, for all profiles for interactive communications drivers.

To specify communications simulation for users, administrators or agents that follow the procedures in this section do *not* need to explicitly set the value of the Simulate driver parameter to TRUE for communications drivers or profiles.

Simulate Parameter and Custom Drivers

If you are implementing a custom communications driver developed using the Adaptive Communications API, if you define a driver parameter Simulate to support communications simulation, enabling simulation as described in this section can also apply to this driver. Otherwise, do not define a parameter with this name.

Enabling Simulation for Siebel Mobile Web Client

End users can access communications simulation using the methods described in this section.

Enabling Using Command-Line Option

An agent can enable communications simulation in the Demo versions of the Siebel applications, which use the Siebel Mobile Web Client with the Sample Database, by including the command-line option /ctsim at the end of the command line, such as in the application shortcut. Any of the Siebel Business Applications can support this command-line option.

Enabling by Modifying Configuration File

An agent can enable communications simulation in the Demo version of the Siebel applications, which use the Siebel Mobile Web Client with the Sample Database, by setting the following parameters in the [Communication] section of the application's configuration file, such as uagent.cfg for Siebel Call Center:

```
CommEnable=TRUE
CommSimulate=TRUE
```

Enabling Simulation for Siebel Web Client

Users can access communications simulation from the Siebel Web Client by connecting the Web browser to an Applications Object Manager for which an administrator has activated simulation by setting server component parameter values. You can use this method of enabling simulation to provide a training environment for your contact-center agents.

As described in "Parameters for Application Object Manager and Siebel Developer Web Client" on page 242, the CommSimulate parameter activates the Communications Simulator, and the CommEnable parameter activates the communications toolbar.

For more information about setting server component parameter values, see *Siebel System Administration Guide*.

To enable or disable communications simulation for Siebel Web Client users

- On each Application Object Manager that is to support communications simulation for users, set the server component parameter CommSimulate (Simulate Communication) to TRUE. Alternatively, to disable simulation, set it to FALSE. To set this parameter, do *one* of the following:
  - Navigate to Administration - Server Configuration > Servers, specify the Siebel Server, and select Call Center Object Manager (for example) in the Components list. Set a value for the Simulate Communication parameter in the Parameters list (for this component).
  - On the command line, run Siebel Server Manager, specifying your Siebel Gateway Name Server, Siebel Enterprise Server, and Siebel Server, as appropriate. In Server Manager, enter (on one line) a command like the following, as appropriate for your Application Object Manager component:

    ```
    change parameter CommSimulate=true for component SCCObjMgr_enu
    ```

After you set the CommSimulate parameter value, the change will take effect for each subsequent user login. Optionally, you may request existing users to log out and log in again.

For more information about using Siebel Server Manager (GUI or command line), see *Siebel System Administration Guide*.

Administering Communications Session Manager

This section describes how to administer the server component Communications Session Manager. The short name for this component is CommSessionMgr.

This server component uses generic configuration parameters and does not need to be configured. However, you can configure logging parameters. For details, see "Parameters for Communications Session Manager" on page 250.

Overview of Communications Session Manager

The Communications Session Manager supports multichannel user-interactive sessions for agents using the communications toolbar for voice, email, or other types of work items. It manages agent sessions, for which applicable interactive communications drivers are loaded into memory.

For most Siebel client deployment choices, the Communications Session Manager must be available for the instances of the Application Object Manager that connect to it for each agent's communications session.

In each case, the component that is employed is identified through the applicable data source for the Siebel application session. For each data source, a specific Siebel Gateway Name Server, Siebel Enterprise Server, and Siebel Server are associated. The Communications Session Manager component must be enabled and running on this Siebel Server.

Alternatively, you can specify the following communications configuration parameters, to identify the Communications Session Manager to use:

- **GatewayAddress.** Specify the Siebel Gateway Name Server. For example, specify a value like *gateway-host*.
- **EnterpriseServer.** Specify the Siebel Enterprise Server. For example, specify a value like *siebel*.
- **RequestServer.** Specify the Siebel Server. For example, specify a value like *server-host*.
- **CommSessionMgr.** Specify the Communications Session Manager. For example, specify a value like *CommSessionMgr*.

If an agent logs in using a configuration containing these parameters, the agent will be connected to the component identified by these values. For more information about these parameters, see "Specifying Parameters for Communications Configurations" on page 52.

For information about how the CommLocalDriver parameter relates to the Communications Session Manager component, see "Parameters for Application Object Manager and Siebel Developer Web Client" on page 242.

When Communications Session Manager Is Unavailable

If the Communications Session Manager stops running or becomes unavailable for some reason, all connected users (agents) will receive the following message in an alert-type dialog box in their browser: "Connection to Communications Server is down. Toolbar is reset!"

For each such agent, the communications toolbar displays as if no communications driver is loaded. That is, the toolbar is displayed but most of the toolbar buttons are unavailable.

Running Communications Session Manager

When the Communications Management component group is enabled, the Communications Session Manager component is started automatically. For any machine on which you do not want to run Communications Session Manager, configure the Siebel Server not to start it.

Communications Session Manager is a batch-mode component. It relies on the services of the Server Request Broker and Server Request Processor server components. These components must be running on the Siebel Server for communications to be processed successfully.

NOTE: If your CTI middleware server is restarted, then, depending on the third-party vendor requirements, you may also need to restart the Communications Session Manager server component that connects to it. Restarting the Communications Session Manager is recommended for deployments of Siebel CTI Connect.

For more information about configuring, starting, and stopping Siebel Server components, see *Siebel System Administration Guide*.

Administering Communications Configuration Manager

This section describes how to administer the server component Communications Configuration Manager. The short name for this component is CommConfigMgr.

This server component uses generic configuration parameters and does not need to be configured.

The Communications Configuration Manager enhances application performance when agent communications sessions are initiated. It does this by reducing or eliminating the downloading of communications configuration data for each session. After the first agent login that uses a particular configuration, for all subsequent logins with the same configuration, the configuration is loaded from a cache rather than loaded from the database.

For information about enabling the use of this component using the CommConfigManager parameter, see "Parameters for Application Object Manager and Siebel Developer Web Client" on page 242.

The Communications Configuration Manager must be available for the instances of the Application Object Manager that connect to it for each agent's communications session.

In each case, the component that is employed is identified through the applicable data source for the Siebel application session. For each data source, a specific Siebel Gateway Name Server, Siebel Enterprise Server, and Siebel Server are associated. The Communications Configuration Manager component must be enabled and running on this Siebel Server.

NOTE: Enable the Communications Configuration Manager only after the communications configuration has been thoroughly tested and you are ready to deploy it to your agents. If any changes are made to the communications configuration, then you must restart the Communications Configuration Manager component in order to refresh the cached configuration data.

Running Communications Configuration Manager

When the Communications Management component group is enabled, the Communications Configuration Manager component is started automatically. For any machine on which you do not want to run Communications Configuration Manager, configure the Siebel Server not to start it.

For more information about configuring, starting, and stopping Siebel Server components, see *Siebel System Administration Guide*.

Administering Communications Inbound Receiver

The server component Communications Inbound Receiver receives and processes inbound work items, such as email messages. Depending on your deployment, it may queue them for further processing by the Communications Inbound Processor component. The short name for this component is CommInboundRcvr.

For more information about administering and running Communications Inbound Receiver, see *Siebel Email Response Administration Guide*.

About Real-Time and Nonreal-Time Processing Modes

You can process inbound communications events in one of two modes, real-time and nonreal-time.

Real-time mode is the default setting for each response group. To specify nonreal-time mode for a response group, you add the input argument RealTime, with a value of FALSE. See the *Siebel Email Response Administration Guide* for information on how to set this input argument.

Each mode has implications for how you deploy Siebel Server components, as follows:

- **Real-time mode.** In this mode, Communications Inbound Receiver both receives and processes inbound messages. Communications Inbound Processor is not used. Real-time mode may be deployed in the following cases:
  - Real-time mode is *required* for handling voice work items (using Siebel CTI with Siebel Universal Queuing, in order to route calls).

- Real-time mode may be appropriate if you have a relatively low volume of messages. In such a case, you can use real-time mode in order to perform all inbound processing through a single component (Communications Inbound Receiver).

- **Nonreal-time mode.** In this mode, Communications Inbound Receiver receives inbound messages and queues them for further processing by Communications Inbound Processor. You can determine how many instances of each component you require.

Inbound processing for Siebel Email Response can use either real-time or nonreal-time mode, according to your business requirements.

NOTE: Nonreal-time mode may be most appropriate for Siebel Email Response deployments with a high volume of messages, because of the greater degree of scalability and availability provided.

Event Processing for Real-Time and Nonreal-Time Modes

In each mode, inbound email is saved as event data to the local disk, in a file with extension .evt. (Message content and attachments are also saved to the Siebel File System, as described in "Activity Attachments Stored for Inbound Email" on page 259.) How the inbound event is then processed depends on whether you are using real-time mode or nonreal-time mode:

- **Real-time mode.** In real-time mode, where all event processing is done by Communications Inbound Receiver, additional files are created on the local disk that represent different states of processing or represent error conditions. Such file extensions include:

  - .xevt – Events ready for processing by the worfklow process.
  - .error – Events that generated known errors and that can be retried.
  - .crash – Events that failed for unknown reasons and should not be retried.

  For more information about the processing of these files, see *Siebel Email Response Administration Guide*.

- **Nonreal-time mode.** In nonreal-time mode, Communications Inbound Receiver writes event data to the Siebel Database (in the S_CM_INBND_EVT table) and to the Siebel File System, and submits a request to Server Request Processor and Server Request Broker. Communications Inbound Processor picks up this request and processes the event.

  Nonreal-time events that have been submitted to Communications Inbound Processor can be tracked using the Communications Inbound Events view in the Administration - Communications screen (unlike events in real-time mode). Events that could not be fully processed due to transient errors can be resubmitted manually from this view.

  Valid status for events in the Communications Inbound Events view include:

  - **Queued.** Communications Inbound Receiver has submitted the event for processing by Communications Inbound Processor.
  - **CIP Processing.** Communications Inbound Processor has received the event and is currently processing the event.

- **Error.** Event processing has failed due to a known error. You can resubmit the event for processing using the Submit Request command in the applet menu.
- **Fatal.** Event processing has failed due to an unknown error and should not be resubmitted.

For more information about using the Communications Inbound Events view, see *Siebel Email Response Administration Guide*.

Running Communications Inbound Receiver

When the Communications Management component group is enabled, the Communications Inbound Receiver component is started automatically. For any machine on which you do not want to run Communications Inbound Receiver, configure the Siebel Server not to start it. Communications Inbound Receiver restarts automatically if the Siebel Server goes down and is brought back up.

Communications Inbound Receiver is a batch-mode server component, although it also has characteristics of other component types. It relies on the services of the Server Request Broker and Server Request Processor server components. These components must be running on the Siebel Server for communications to be processed successfully.

As a batch-mode component, Communications Inbound Receiver is subject to the requirements described in "Synchronizing Batch-Mode Server Components" on page 239.

For more information about configuring, starting, and stopping Siebel Server components, see *Siebel System Administration Guide*.

To provide greater availability and reliability for your deployment of the Communications Inbound Receiver server component, active response groups and individual profiles loaded within active response groups can be updated without your having to stop and restart the server component:

- Profiles can be updated by using the Submit Profile Changes command for the profile.
- Response groups can be updated (all profiles within the current response group) by using the Submit Response Group Changes command for the response group.

For more information, see "Specifying Parameter Override Values for Profiles" on page 47. See also *Siebel Email Response Administration Guide*.

Configuring Parameters for Communications Inbound Receiver

You can specify values for the following parameters for the Communications Inbound Receiver server component:

- **Administrator Email Address (alias AdminEmailAddress).** Specify a value for the Administrator Email Address parameter, to provide an administrator email address to send messages to in case of inbound communications processing errors, when no such address is defined for the response group.

- **Default Administrator Address (alias DefaultAdminAddress).** Specify a value for the Default Administrator Address parameter, to provide an administrator email address to send messages to in case of server errors, such as when the database connection has been broken.
- **Event Queue Directory (alias EventQueueDirectory).** Specify a value for the Event Queue Directory parameter, to provide the name of the directory in which to store queued events. Use this parameter to specify a value to override the default location in the "bin/queued" directory.
- **Maximum Tasks (alias MaxTasks).** Specify a value for the Maximum Tasks parameter, to configure the maximum number of tasks for this component. (This parameter is part of the Process Management subsystem.)

  NOTE: Maximum Tasks should always be set to a value greater than the value of Max Threads.
- **Max Threads (alias MaxThreads).** Specify a value for the Max Threads parameter, to configure the maximum number of threads for this component. The value that you specify determines the number of threads that the Communications Inbound Receiver can create to process email at the same time. The value that you specify for MaxThreads should take account of the CPU capacity available on the machine that hosts the Communications Inbound Receiver.
- **SMTP Server Name (alias SMTPServer).** Specify a value for the SMTP Server Name parameter, to provide the name of the SMTP server that will be used to send email to the administrator.
- **SMTP Server Port (alias SMTPServerPort).** Specify a value for the SMTP Server Port parameter, to provide the port for the SMTP server that will be used to send email to the administrator.

Activity Attachments Stored for Inbound Email

For each inbound email message that has been processed by Communications Inbound Receiver or Communications Inbound Processor, an activity record is created, using activity type Email - Inbound. The original email content is saved in the form of one or more attachments to this activity record.

The entire original email message content (the entire MIME-encoded message as received by the POP3 server) is saved in an attachment file named Original_Msg.txt.

If the original message is more than 16,000 characters long (including any HTML markup), then the full message is saved as another attachment, named SiebelLongEmailBody.txt (for plain-text messages) or SiebelLongEmailBody.htm (for HTML messages).

Additional files may be created and saved as attachments to the activity, depending on the existence of embedded message content and on the setting of the Parse Embedded Messages parameter for the Internet SMTP/POP3 Server driver. For more information about this parameter, see "Driver Parameters for Internet SMTP/POP3 Server Driver" on page 360.

- When Parse Embedded Messages is FALSE, the following behavior applies:

- For each embedded message, a single attachment file is created that contains the entire MIME embedded message. These files are named EmbeddedMsg*partspecifier*.eml, where *partspecifier* represents the file's placement in the original message's structure—for example, EmbeddedMsg01.eml, EmbeddedMsg3.4.eml, and so on. These files can be opened using any application that can read an EML file, such as Microsoft Outlook Express.

- When Parse Embedded Messages is TRUE (the default), the following behavior applies:
  - For a plain-text email message, all text components are saved in one or more attachment files. These files are named textplain*partspecifier*.txt, where *partspecifier* represents the file's placement in the original message's structure—for example, textplain01.txt, textplain3.4.txt, and so on.
  - For an HTML email message, all HTML components are saved in one or more attachment files. These files are named texthtml*partspecifier*.htm, where *partspecifier* represents the file's placement in the original message's structure—for example, texthtml01.htm, texthtml3.4.htm, and so on.
  - Any attachments to the original email message are also saved as attachments to the activity record, using their original file names.

Administering Communications Inbound Processor

The server component Communications Inbound Processor processes inbound work items, such as email messages, that were previously received and queued by the Communications Inbound Receiver component. The short name for this component is CommInboundProcessor.

NOTE: Communications Inbound Processor is used for nonreal-time event processing, but not for real-time event processing. For more information, see "About Real-Time and Nonreal-Time Processing Modes" on page 256.

Running Communications Inbound Processor

When the Communications Management component group is enabled, the Communications Inbound Processor component is started automatically. For any machine on which you do not want to run Communications Inbound Processor, configure the Siebel Server not to start it. Communications Inbound Processor restarts automatically if the Siebel Server goes down and is brought back up.

Communications Inbound Processor is a batch-mode server component. It relies on the services of the Server Request Broker and Server Request Processor server components. These components must be running on the Siebel Server for communications to be processed successfully.

As a batch-mode component, Communications Inbound Processor is subject to the requirements described in "Synchronizing Batch-Mode Server Components" on page 239.

For more information about configuring, starting, and stopping Siebel Server components, see *Siebel System Administration Guide*.

For more information about administering and running Communications Inbound Processor, see *Siebel Email Response Administration Guide*.

Administering Communications Outbound Manager

This section describes how to administer the server component Communications Outbound Manager. The short name for this component is CommOutboundMgr.

The Communications Outbound Manager processes outbound communications, for email, fax, or page channels. It supports communication requests, whether they are created and submitted directly or by Siebel Workflow. It also supports outbound capabilities for Siebel Email Response and for the Send Email, and Send Fax commands.

For this server component, you can configure a parameter that specifies how Siebel bookmarks are generated and configure a logging parameter. Otherwise, this component uses generic parameters and does not need to be configured.

For details, see "Configuring Communications Outbound Manager" on page 262 and "Configuring Shared or Separate Logging" on page 262.

See also Chapter 10, "Defining Outbound Communication Requests," and other sections describing features that use outbound communications.

Running Communications Outbound Manager

When the Communications Management component group is enabled, the Communications Outbound Manager component is started automatically. For any machine on which you do not want to run Communications Outbound Manager, configure the Siebel Server not to start it.

Communications Outbound Manager is a batch-mode server component. It relies on the services of the Server Request Broker and Server Request Processor server components. These components must be running on the Siebel Server for communication requests to be dispatched successfully.

NOTE: If a messaging system server, such as an email server, is restarted, then the Communications Outbound Manager server component that connects to it must also be restarted.

As a batch-mode component, Communications Outbound Manager is subject to the requirements described in "Synchronizing Batch-Mode Server Components" on page 239. If Communications Outbound Manager has not been synchronized, as appropriate, then users who submit outbound communication requests may receive this error message:

```
Unable to find definition for component CommOutboundMgr.
```

For more information about monitoring communication requests and server requests, see "Monitoring Outbound Communication Request Status" on page 300 and "Monitoring Siebel Server Request Status" on page 301.

For more information about configuring, starting, and stopping Siebel Server components, see *Siebel System Administration Guide*.

Configuring Communications Outbound Manager

For Communications Outbound Manager, you can configure a parameter that specifies how Siebel bookmarks are generated, configure a parameter that specifies whether logging should use shared or separate files, and configure logging levels.

For more information about configuring logging for server components, see *Siebel System Administration Guide*.

Configuring Siebel Bookmarks

In order to support the Attach Bookmark setting for advanced communications templates, the Siebel administrator must specify a value for the WebServer server component parameter for the Communications Outbound Manager.

This parameter specifies a string identifying the Web server and Application Object Manager to include in the URL, in the following form:

`http://web_server/application_object_manager`

In order to access the bookmarked record, the recipient users must have access to the specified Web server and Application Object Manager.

For more information about using the Attach Bookmark setting for advanced templates, see "Fields for Templates" on page 280.

Configuring Shared or Separate Logging

An administrator can review log files for Communications Outbound Manager to monitor its performance and to monitor user activities that invoke this component.

Depending on how you have configured the Communications Outbound Manager server component, a single log file may be generated for all requests, or a separate log file may be generated for each request (the default). The parameter that modifies this setting is called LogUseSharedFile.

Log files are written to the log subdirectory of the Siebel Server installation directory.

Set LogUseSharedFile according to how the server component is generally used:

- For miscellaneous uses, including supporting the Send commands, sending auto-acknowledgement messages and replies for Siebel Email Response, or sending several outbound requests to a small number of recipients each, it may be best to set LogUseSharedFile to TRUE, to reduce clutter in your log directory.
- For high-volume outbound communication requests, however, it may be best to leave this parameter set to FALSE, to generate a single log file for each request that you can analyze. LogUseSharedFile is FALSE by default.

The log file names vary according to how you set LogUseSharedFile:

- When a single log file is generated for all requests, the file name is in the form CommOutboundMgr_*xxx*.log, where *xxx* is the ID number for the main Communications Outbound Manager task.

- When a separate log file is generated for each request, the file names are in the form CommOutboundMgr_*xxx*.log, where *xxx* is the ID number for the Communications Outbound Manager task applicable to the specific communication request.

The business service Outbound Communications Manager, run on the Siebel Server, also generates log files in the same location, using the same naming convention.

Configuring Log Levels for Communications Outbound Manager

An administrator can set logging levels for Communications Outbound Manager, to specify the degree of detail captured in the logs. Logging levels are:

- **CommSrvrError (level 1).** The lowest level of logging. The most severe errors are logged. Level 1 is the default.
- **CommSrvrWarning (2).** Moderate level of logging. Logs more detail than level 1. Generally recommended for production use.
- **CommSrvrTrace (3).** Moderately high level of logging. Logs more detail than level 2.
- **CommSrvrDebug (4).** The highest level of logging. All errors and warnings and other events are logged. Use level 4 for testing purposes.

The procedures below are for the Siebel Server Manager—both the Siebel client version (GUI) and the command-line version.

To configure log levels for Communications Outbound Manager (GUI)

1. Navigate to Administration - Server Configuration > Servers.
2. Specify the Siebel Server on which Communications Outbound Manager is running.
3. In the Components list, select the record for the Communications Outbound Manager component.
4. Click the Events tab.
5. Select the record for the event type CommServer.
6. Specify one of the values described above, from 1 to 4.

To configure log levels for Communications Outbound Manager (command line)

1. Start the command-line version of the Siebel Server Manager.
2. Type the following at the command line:

   `change evtloglvl CommSrvr=`*n* `for comp CommOutboundMgr`

   where *n* is the log level you want to specify, from 1 to 4.

Specifying the Siebel Server for Communications Outbound Manager

If you have more than one Siebel Server, and you want all outbound communications that are processed using a particular communications driver or profile to use a specific Siebel Server, you can configure a parameter to specify its name.

To do this, you set the value of the Siebel Server driver parameter (generally, using a profile parameter override) to the name of the Siebel Server that is to handle the delivery of the outbound communications.

This parameter is supported for outbound communications sent (using the Communications Outbound Manager server component) using communications drivers that support outbound communications.

NOTE: The value of the Siebel Server parameter must exactly match the actual name of the Siebel Server.

As with other driver parameters, you can either specify a default value or provide an override value for a particular communications profile.

NOTE: If an outbound communication request includes multiple templates, and a value is provided for this parameter for the delivery profile associated with any of these templates, then a different value cannot be specified for this parameter for profiles for any of the other templates specified in the same request.

For more information, see the driver parameter sections in Chapter 13, "Using Email, Fax, and Other Systems."

Specifying Component Name for Outbound Communication Requests

When you create an outbound communication request, you can specify the name of the server component that is to process the request.

The default component name is CommOutboundMgr (for Communications Outbound Manager). If you have configured new server components based on CommOutboundMgr, you can give them other names. Then, when you create requests, you can provide the applicable component name in the Component Name field for the request.

When Siebel Server load-balancing is in effect, you can create multiple components using the same name, such as on a subset of your total set of Siebel Servers.

If different Siebel Servers may use different Siebel repository (.srf) files, then you may want to use this mechanism to make sure that certain types of requests are processed by a specified set of Siebel Servers.

For more information, see the description for the Component Name field, in the section "Fields for Outbound Communication Requests" on page 297.

Outbound Communications for Siebel Mobile Web Client

Outbound communications sent from Siebel Web Client are processed immediately. Communications sent from Siebel Mobile Web Client, while you are connected to the local database and disconnected from the enterprise database, are saved until you synchronize. They are then processed for delivery by the Communications Outbound Manager server component.

9 Configuring Communications Templates

This chapter provides information about configuring and managing communications templates. It includes the following topics:

About Communications Templates

Communications templates may be sent to outbound communications recipients in the following scenarios:

- Using the Send Email, Send Fax, or Send Page commands
- Replying to an inbound message in Siebel Email Response
- Creating and submitting an outbound communication request (whether this is done manually or done using a module like Siebel Workflow that can invoke business service methods)

Templates provide both structure and content for outbound messages. Content may include both explicitly specified text and Siebel field values substituted into the outbound message. Field data may be substituted for all supported Siebel data types. In an end-user context of using the Send commands or replying to inbound messages, additional content can also be specified by the end user.

For outbound communication requests, which can include multiple templates per request, each template also includes a specification of the communications profile that determines how the template's message will be delivered. For more information about outbound communication requests, see Chapter 10, "Defining Outbound Communication Requests."

A set of templates supporting different communication channels are provided by Siebel Systems. These templates may be used or adapted for use with the Send Email command, Siebel Email Response, outbound communication requests, Siebel Workflow, and so on.

For additional guidelines on creating templates for use by Siebel products documented in other books, such as Siebel Email Response, see the applicable documentation, such as *Siebel Email Response Administration Guide*.

NOTE: The procedures described in this chapter can be performed either by administrators or by advanced end users.

If you are implementing a custom channel type, see also "Configuring Communications List of Values Types" on page 163.

Template Content and Formatting

Templates can include any of the following content elements:

- A single block of template text specified in the Text field.
- One or more template items providing component parts for the template. A template item can be based on a Siebel literature item or a text file.
- A combination of template text and template items.

NOTE: For template users to be able to work with literature items or files, the Siebel File System must be properly configured.

For email or fax messages, the content of a template item can be specified to appear in the body of a message or to appear as an attachment. Both plain-text and HTML content are supported for email or fax templates, with some limitations for fax. Standard HTML tagging options can support text formatting, images, hyperlinks, tables, and so on. For more information, see "HTML Formatting."

HTML Formatting

For email or fax templates, HTML can be defined in the template text or, for advanced templates, in template items. (Template items are supported only for advanced templates, which are sent using outbound communication requests.)

Template text can be formatted as HTML using controls in the Text field. When a template is saved, formatting of text in this field is preserved only if the HTML Template check box is checked for the template.

For advanced templates, checking the HTML Template check box also specifies that HTML text from template items specified for the message body is sent as HTML. If HTML Template is not checked, this text is converted to plain text before sending.

If the default message format (specified in the Send Email group in the Outbound Communications options of the User Preferences screen) is set to HTML, then HTML formatting is enabled for the Send Email and Send Fax windows. HTML templates only are listed in the Body drop-down list.

Similarly, if the default message format setting for Siebel Email Response is set to HTML (also in the Outbound Communications options of the User Preferences screen), then HTML formatting is enabled for email replies in the Outbound Messages form of the Communications screen. HTML templates only are listed in the Body drop-down list.

For more information, see:

- "Template Visibility and Access" on page 270
- "Preference Settings for Outbound Communications" on page 306
- "Editing and Formatting Controls for Template Text" on page 275
- "Editing and Formatting Controls for Send Email and Send Fax" on page 332
- Chapter 10, "Defining Outbound Communication Requests"
- "Using HTML Email" on page 356

For more information about HTML formatting for Siebel Email Response, see *Siebel Email Response Administration Guide* and *Applications Administration Guide*.

HTML Wrapper Templates for Email

You can create a special template to serve as a wrapper around each outbound email message, in order to format messages with HTML. Such a wrapper template can contain links to your Web site, marketing messages, and so on.

To create a wrapper template, you can use any HTML editor. To use this feature with outgoing email, create communications templates as described in this section, and modify the appropriate workflows in Siebel Workflow. By default, the applicable workflow is Email Response - Client Send Email.

Users send messages as they normally do. When a user clicks Send, the email is embedded in your wrapper template and sent to the message recipient.

For more information about using HTML wrapper templates with Siebel Email Response, see *Siebel Email Response Administration Guide*.

Substitution Fields in Templates

Field values from Siebel Database records can be substituted into the structure of the template for each recipient of a communication for which the template is used. Field substitution can apply to text specified directly for the template, and to the subject line, as well as to literature items and files.

Available fields for substitution are those that correspond to the recipient group or business object specified for the template:

- For simple templates, which are used with Send commands or Siebel Email Response replies, you can specify a business object that has a primary business component—for example, Account.
- For advanced templates, which are used with outbound communication requests, you can specify a predefined recipient group that corresponds to one of the recipient groups you can specify for requests—for example, Account Team Members.

Fields from the Account business component can be inserted into template text (for simple or advanced templates) or inserted into template items (for advanced templates only).

For a template item, you can also use advanced field substitution, where you specify an iteration child business component from which to substitute field values for a given template item. Each template item repeats, in iterative fashion, for every applicable child record.

For example, assume the recipient group is Account Team Members, and Opportunity is specified in the Iteration Child Business Component field for the template item. In the outgoing message, the template item's content will be repeated once for each opportunity (child business component) that is associated with the account (parent business component).

Field substitution can be employed within plain-text or HTML templates or template items. (Template items specifying files of other formats, such as RTF or Microsoft Word, cannot support field substitution.)

In an HTML template or template item, for example, a row containing substitution fields can, for example, be specified in a single-row HTML table in order to align field data. If an iteration child business component is specified, the tables are repeated, thus allowing you to align field data from multiple records.

In order for field substitution to function correctly for an outbound communication request, the request must specify the same recipient group as was specified for the template. If a different recipient group is used, the field names (with enclosing brackets) may appear in each message as literal character strings without substitution. Substitution still occurs for fields that are applicable to the request's recipient group.

Siebel Repository Requirement for Using Substitution Fields

The Siebel repository applicable to where templates are created must be consistent with the repository applicable to where templates will be used. Otherwise, field substitution behavior when the templates are used may not occur as desired.

For example, if you use the Siebel Mobile Web Client (connected to a local database), then you create templates only if the applicable business objects and business components (and substitution fields) in that environment are consistent with those in the enterprise database.

Template Visibility and Access

Templates can be created solely for personal use, or can be made publicly available to all users of the applicable communications features.

NOTE: In general, templates that are created for group use should be created, modified, or reviewed by advanced users who have administrator privilege. Such users work with templates in the All Templates view in the Administration - Communications screen.

Depending on the context of use, which templates are displayed to a user, such as in the Body drop-down list, depends on the following factors:

- The setting of the Public check box for the template. Templates listed for a user will not include those for which Public was not checked. Templates the user created (such as in the My Templates view in the Communications screen) are not subject to this limitation.
- The specified channel (such as email or fax) for the template. Templates listed are only those applicable to the feature. For example, only email templates are listed for Send Email and other email features.

- The specified template type, such as Greeting, Body, or Closing. For the Send commands, the template type must be Body in order to appear in the Body drop-down list. For email replies, templates of each type are specified using a different drop-down list. (Template type is ignored for communication requests.)
- The specified language and locale for the template. For the Send Email and Send Fax commands, templates listed are only those for which the language and locale correspond to the user's outbound communications preferences. (If these user preferences are not set, these language and locale for the Application Object Manager are used—as applies for all Siebel client types.)

  For the Send Email command and for email replies, the user can specify overrides for the language and locale settings, in order to view a different set of templates to choose from.
- For simple templates used with the Send commands: the specified object (such as Account) for the template. Templates listed are only those for which the specified business object corresponds to the business component (such as Account) for the current applet when the user chose the Send command. Templates listed may also include those for which no object was specified.
- For templates used with the Send Email or Send Fax commands, or with email replies: the setting of the HTML Template check box for the template. In the Outbound Communications options of the User Preferences screen, users can specify a default message format of either Plain Text or HTML. This user preference setting is specified separately for Send Email (and Send Fax) and for Email Response. In each context of use:
  - If the preference setting is HTML, templates listed are only those for which HTML Template is checked.
  - If the preference setting is Plain Text, templates listed are only those for which HTML Template is not checked.

Siebel Views for Working with Templates

You create templates in either of the following views:

- My Templates view (in the Communications screen, for end users)
- All Templates view (in the Administration - Communications screen, for administrators or other authorized advanced users)

In these views, respectively, end users see only the templates they have created (My Templates), and administrators see all templates (All Templates).

NOTE: Procedures in this section generally assume a non administrator user, using the My Templates view in the Communications screen.

Creating Simple Templates

This section describes how to create a simple template. Simple templates are used for the Send Email, Send Fax, or Send Page commands. These commands may be accessed from the File application-level menu, from the communications toolbar, or using keyboard shortcuts. Simple templates also apply to replies in Siebel Email Response.

For information about the fields for the Templates list and the Simple tab, see "Fields for Templates" on page 280.

Outbound communication requests use advanced templates. Template items are supported for advanced templates only. For more information, see "Creating Advanced Templates" on page 273 and "Specifying Template Items for an Advanced Template" on page 277.

To create a simple template

1. Navigate to Communications > My Templates.
2. In the Templates list:
   a. Add a new record. (Alternatively, you can add the record in the Simple form below.)
   b. Specify the name of the template for which you added a new record.
   c. Specify the channel type, such as Email, Fax, or Pager.
   d. Specify the language and locale that is applicable to this template.
   e. Optionally, provide a short description for the template.
3. In the Simple form:
   a. Specify the name of the template, if you did not already do so in the Templates list.
   b. Specify the channel type for the template, if you did not already do so in the Templates list.
   c. Specify the template type, such as Greeting, Body, or Closing:
      - When using the Send Email, Send Fax, or Send Page command, users can choose specific templates of type Body to insert into the message.
      - When replying to an inbound email message using Siebel Email Response, users can choose specific templates of each type to insert into the reply.
   d. Specify the language and locale for this template, if you did not already do so in the Templates list.
   e. For an email or fax template, specify whether this is an HTML template or a plain-text template.

      For a simple template, this setting determines whether the template is listed in the Body drop-down in usage context. Templates are listed only when they correspond to the user preference setting for Default Message Format. For more information, see "Template Visibility and Access" on page 270.
   f. For a template that is intended for public use, check the Public box when the template is ready to be publicly available.
   g. Optionally, specify for Pick Available Substitutions the appropriate business object (Object) as the template's recipient group:
      - The primary business component of the business object that you specified determines which substitution fields are available for the template text. The business components listed are those that serve as the primary business component for a business object. For more information, see "Substitution Fields in Templates" on page 269.

- For the Send commands, the recipient group also helps determine when users can specify the template. For more information, see "Template Visibility and Access" on page 270.

The actual recipients are determined in the context in which the template is used.

h. Specify a subject line for the template, as applicable for the channel type.

The subject line is overridden by any subject line specified by the user in the context in which the template is used—for example, in an email message in the Send Email window, or in a reply in Siebel Email Response.

i. Enter the template text. As appropriate, copy and paste fields from the list of available field substitutions. Include the brackets around the field names.

The Text field allows you to use editing and formatting controls, as described in "Editing and Formatting Controls for Template Text" on page 275. For an HTML email or fax template, formatting you apply is preserved. For a plain-text template, formatting is stripped when the template is saved.

The Substitutions list is populated with field names as described in "Fields for Templates" on page 280 (for the Simple tab).

4. Test the template to verify that it functions as required and that field substitution behavior works correctly. Do this before you make a template publicly available.

To test a template, send it in the intended context: that is, by using the applicable Send command, or by specifying a test reply for Siebel Email Response.

Creating Advanced Templates

This section describes how to create an advanced template. Advanced templates are specified for outbound communication requests. Differences from simple templates include the following:

- Delivery profile can be associated for an advanced template.
- A recipient group is specified for an advanced template (a business object is specified for a simple template).
- You can specify to create activities for requests that use the template.
- You can specify to include a bookmark URL for the recipient source record.
- You can create template items for advanced template. For more information, see "Specifying Template Items for an Advanced Template" on page 277.

For information about the fields for the Templates list and the Advanced tab, see "Fields for Templates" on page 280. For information about the fields for template items, see "Fields for Template Items" on page 287.

For information about creating a simple template, see "Creating Simple Templates" on page 271.

NOTE: For an advanced template, a communications profile you will specify as the delivery profile must already exist before you can specify it for the template. For more information about creating profiles, see "Configuring Communications Drivers and Profiles" on page 44.

To create an advanced template

1. Navigate to Communications > My Templates.
2. In the Templates list:
   a. Add a new record. (Alternatively, you can add the record in the Advanced form below.)
   b. Specify the name of the template for which you added a new record.
   c. Specify the channel type, such as Email, Fax, Pager.
   d. Specify the language and locale that is applicable to this template.
   e. Optionally, provide a short description for the template.
3. Click the Advanced view tab.
4. In the Advanced form:
   a. Specify the name of the template, if you did not already do so in the Templates list.
   b. Specify the channel type for the template, if you did not already do so in the Templates list.
   c. Specify the delivery profile for the template.
   d. Specify the language and locale for this template, if you did not already do so in the Templates list.
   e. For an email or fax template, specify whether this is an HTML template or a plain-text template.

      NOTE: The HTML Template setting affects the text you enter in the Text field. It does not affect template items.

   f. For a template that is intended for public use, check the Public box when the template is ready to be publicly available.
   g. For an email template, specify whether to include a Siebel bookmark with the message.
   h. Check the Create Activity box, if you want to set activity logging for any outbound communication request for which this template is specified.
   i. Optionally, specify the recipient group, which determines what kind of recipients the template will be sent to.

      The recipient group you specify here only determines which substitution fields are available for the template text. The actual recipients are determined in the context in which the template is used (for a communication request).

   j. Specify a subject line for the template, as applicable for the channel type.

k Optionally, enter the template text. As appropriate, copy and paste fields from the list of available substitution fields. Include the brackets around the field names.

The Text field allows you to use editing and formatting controls, as described in "Editing and Formatting Controls for Template Text" on page 275. For an HTML email or fax template, formatting you apply is preserved. For a plain-text template, formatting is stripped when the template is saved.

The Available Fields list is populated with field names as described in "Fields for Templates" on page 280 (for the Advanced tab).

A template that includes template items does not require that you also include template text.

5 Optionally, add template items to the template, as described in "Specifying Template Items for an Advanced Template" on page 277.

A template that does not include template text should generally include one or more template items in order to provide content. You can combine template text and template items in the same template. The template text appears in the message body, before the content specified with the template items.

6 Test the template to verify that it functions as required and that field substitution behavior works correctly. Do this before you make a template publicly available.

To test a template, send it in the intended context: that is, by creating and submitting an outbound communication request.

Editing and Formatting Controls for Template Text

For email or fax templates, you can edit and format your template text, using controls in an editing bar that appears when you click in the Text field in the Simple tab or Advanced tab.

NOTE: HTML formatting does not apply to templates for channels other than email or fax, even if the HTML Template check box is checked. Any formatting is stripped out when such a template is used.

You can use the editing controls, such as those for clipboard operations, without applying HTML formatting.

For an HTML email or fax template, formatting you apply in the Text field is preserved when the template is saved. For a plain-text template, such formatting is stripped out when the template is saved.

NOTE: If you change the setting of the HTML Template check box, the change is not apparent until you step off the record to display another template record, then re-select this template record. In particular, note that if you uncheck this box, the HTML formatting controls will still be available, and formatted text is not immediately converted to plain text. However, when the template is saved, any formatting you applied is removed.

Managing Line Breaks for HTML and Plain-Text Templates

The following behavior regarding line breaks applies to *both* plain-text and HTML templates:

- Pressing ENTER creates a new paragraph.
- Pressing SHIFT+ENTER creates a new line.

Managing Links and HTML Elements from Other Sources

For email messages, you can enter URL or mailto links directly and they will be automatically converted to links if the template is saved as HTML. (This behavior may not apply to email sent using third-party email clients.) For example:

- www.siebel.com
- http://www.siebel.com
- ftp://ftp.topsecretclassified.gov
- mailto:user@siebel.com

Graphics or other tag-based elements displayed on Web pages, such as horizontal rules or tables, may be selectable by dragging the mouse pointer. In some cases, such elements can be copied and pasted into the template text.

NOTE: Content other than text or HTML tag elements cannot be added directly to the template text. For example, you cannot copy and paste graphics or other files into the template text. (For advanced templates, graphics can be specified using template items.)

Editing and Formatting Options

The editing and formatting options for your template text include, from left to right:

- **Find/Replace.** Click the arrow to display the Find controls above the editing bar, or to hide these controls. When the Find controls appear above the editing bar, click the arrow on the left to toggle between the Find controls and the Find and Replace controls. Enter text to find, or enter replacement text, then click Go. Find operations are not case-sensitive.
- **Cut.** Click to cut selected text to the clipboard.
- **Copy.** Click to copy selected text to the clipboard.
- **Paste.** Click to paste text from the clipboard into the template text. Depending on the source, text you paste may include HTML formatting.
- **Font.** Choose a font from a drop-down list to apply to selected text. Available fonts include Arial (the default), Verdana, Times New Roman, and Courier.
- **Size.** Choose a size from a drop-down list to apply to selected text. Point sizes include 8 (the default), 10, 12, 14, 18, 24, 36.
- **Font Color.** Click the arrow to display font colors above the editing bar, then click to choose a color to apply to selected text.
- **Bold.** Click to apply bold formatting to selected text, or to remove bold.
- **Italic.** Click to apply italic formatting to selected text, or to remove italics.

- **Underline.** Click to apply underlining to selected text, or to remove underlining.
- **Ordered List.** Click to apply numbering to selected text (make it an ordered list), or to remove numbering.
- **Unordered List.** Click to apply bullets to selected text (make it an unordered list), or to remove bullets. Bullets appear differently at different levels of indenting.
- **Indent.** Click to increase indenting for selected paragraphs.
- **Outdent.** Click to decrease indenting for selected paragraphs (outdent).
- **Left Align.** Click to left-align selected paragraphs.
- **Center Align.** Click to center selected paragraphs.
- **Right Align.** Click to right-align selected paragraphs.

Specifying Template Items for an Advanced Template

This section describes how to add template items for an advanced template.

Specifying template items allows you to construct templates in modular fashion, based on literature items or files. It also allows you to use advanced field substitution (iteration child business component), to specify content both for the message body and for attachments, and to specify alternate versions of the message body (for email only): plain text and HTML.

You can combine template text and template items in the same advanced template. A template that does not include template text should include at least one template item. If you include template text, this text appears in the message body, before any content specified with template items.

You must select either a literature item *or* a file to include for the template item.

Checking the HTML Template check box for a template specifies that HTML text from template items specified for the message body is sent as HTML. If HTML Template is not checked, this text is converted to plain text before sending.

For more information about template item fields mentioned here, see "Fields for Template Items" on page 287.

To add template items for a template

1. Navigate to Communications > My Templates.
2. In the Templates list, select the advanced template for which you are adding template items.
3. For an HTML template, verify that the HTML Template check box is checked.
4. Click the Template Items view tab.
5. In the Template Items list, add a new record for each template item.
   - To add a file, do *one* of the following:

- Click New File *(recommended)*. Choose a file from the Choose File dialog box. Go to Step 6, then go to Step 9.
- Click New (or choose New Record from the menu). Go to Step 6, then go to Step 7.

■ To add a literature item, click New (or choose New Record from the menu). Go to Step 6, then go to Step 8.

6 Specify a sequence integer number for the template item. (The default value is the lowest positive integer not in use by an existing template item.)

7 To finish adding the file (begun in Step 5):

a Click the select button for the Attachment Name field.

b In the Add Attachment dialog box, click Browse, specify a file from the operating system, then click Add. Go to Step 9.

8 To finish adding the literature item (begun in Step 5):

a Click the select button for the Literature Name field.

b In the Pick Literature dialog box, specify a literature item, then click OK.

9 For a template item that is to be an attachment to an email or fax message, specify a label in the Attachment Label field.

10 In the Template Item form:

a Optionally, specify an iteration child business component from which you can specify substitution fields. For more information, see "Fields for Template Items" on page 287.

b Check the Substitute Values box, if you want to use field substitution for the file or literature item specified for the template item in previous steps.

The Available Fields list is populated with field names.

c Check the Message Body box, as appropriate:

- For an email or fax template, check the Message Body box to include the template item content in the message body. Otherwise, the template item content will be an attachment to the email or fax message.
- For a page template, you must check the Message Body box to include the template item. For a page template, the message will fail if any template item is not specified for the message body. (Template items must be plain-text files or literature items.)

Modifying a Template Item That Specifies a File

If you need to modify a file you added to a template item in the previous procedure, you can do this in either of two ways.

To modify a template item file if the original operating system file still exists

1 Open the file in the operating system environment.

2 Modify it, and save it to the operating system.

3 Delete the template item record to which you added the file.

4 Create a new template item record, adding the modified version of the file.

To modify a template item file if the original operating system file does not exist

1 Open the file from the template item record, by clicking the hyperlink in the Attachment Name field.

2 Modify that file, and save it to the operating system.

3 Delete the template item record to which you added the file.

4 Create a new template item record, adding the modified version of the file you just saved.

Copying or Deleting Templates and Template Items

You can copy or delete a template, copy a template item within a template, or delete a template item from a template.

Copying Templates and Template Items

When you copy a template, you must specify a new name for the new template. As appropriate, you can modify existing settings such as the delivery profile, specify different template text, or add template items. No template items are automatically associated with the new template.

When you copy a template item within a template, the copy is identical to the template item that you copied, except that its sequence number is one higher than the highest sequence number for all the other template items. As appropriate, you can modify existing settings, such as to specify a different file or literature item, change whether the template item's content is to appear in the message body, or change the value-substitution settings.

NOTE: Copying a communications template created for use with one channel type and modifying it for use with another channel type may not always work as intended.

Deleting Templates and Template Items

When you delete a template, its template items are also deleted. However, the files or literature items used by the template items are not deleted from the Siebel File System. When you delete a template item, the file or literature item used by the template item is not deleted from the Siebel File System.

Each end user can delete any template the user has created, or delete template items from such a template. A Siebel administrator can delete any template or template item.

Fields for Templates

This section contains descriptions for the fields in the My Templates or All Templates view that you specify when you create templates. The options are organized according to where they are specified.

For descriptions of the fields for template items, see "Fields for Template Items" on page 287.

Templates List

The following options are specified in the Templates list:

- **Name (required).** Specify a unique name for the template.
- **Channel Type (required).** Select from a picklist the type of communication channel for the template. Predefined options include Email, Fax, and Pager.

  For templates to be used with outbound communication requests, the channel type can help determine how to deliver a template, such as for fax messages.

  The Default Preference setting you can specify when creating requests (for use with the Only Send Preference check box) generally corresponds to the Channel Type setting for the included templates.

  For the Send commands and for email replies in Siebel Email Response, the channel type also determines whether the template will appear in the Body drop-down list in the context of use. For more information, see "Template Visibility and Access" on page 270.
- **Language (required).** Specify the language for the template. Templates listed for use are filtered based on this setting. The field value defaults to the current language in effect for the user's session.

  For more information, see "Template Visibility and Access" on page 270 and "Preference Settings for Outbound Communications" on page 306.
- **Locale (required).** Specify the locale for the template. Templates listed for use are filtered based on this setting. The field value defaults to the current locale in effect for the user's session.

  For more information, see "Template Visibility and Access" on page 270 and "Preference Settings for Outbound Communications" on page 306.
- **Description.** Enter a brief description of the template.

Simple Form

This section describes the template options you specify in the Simple form.

Template Properties

The following fields are grouped under Template Properties:

- **Name (required).** See "Templates List" on page 280.

- **Channel Type (required).** See "Templates List" on page 280.
- **Template Type.** Specify the type of template. This option helps users choose templates with the appropriate function. Default values include Greeting, Body, and Closing. Consider the following points when you specify a template type:
  - A reply message for Siebel Email Response can use all of these template types.
  - The Send commands use templates of type Body.
- **Language (required).** See "Templates List" on page 280.
- **Locale (required).** See "Templates List" on page 280.
- **HTML Template.** Check this option to specify that an email or fax template is an HTML template. Leave this box unchecked if the template is plain text. If this box is not checked, then any formatting you apply in the Text field is stripped when you save the template.

  For an advanced template, checking the HTML Template check box also specifies that HTML text from template items specified for the message body is sent as HTML. If HTML Template is not checked, this text is converted to plain text before sending.

  For information about HTML formatting in the Text field, see "Editing and Formatting Controls for Template Text" on page 275.

  This setting also determines whether the template may be listed in the context of use, based on the applicable outbound communications user preference setting for default message format (for Send Email or Send Fax, or for email replies for Siebel Email Response).

  For more information, see "Template Visibility and Access" on page 270 and "Preference Settings for Outbound Communications" on page 306.
- **Public.** Check this option to make this template available for all users to specify in the context in which the templates will be used, such as in using the Send commands or sending email replies. Leave this box unchecked if the template is available only to the current user and to administrators.

Pick Available Substitutions

The following fields are grouped under Pick Available Substitutions:

- **Object.** Select from a picklist the type of recipient group (object) for the template. The recipient groups you can choose correspond to business objects for which a primary business component is specified—such as Account, Opportunity, Contact, and so on.

  The object specified for the template is optional and is used to obtain values for the Substitutions list.

  The recipient group you specify here only determines which substitution fields are available for the template text. The actual recipients are determined in the context in which the template is used—such as in using the Send commands or sending email replies.

  NOTE: Compare the Object field on the Simple tab functions to the Recipient Group field on the Advanced tab.

- **Substitutions.** Displays the business component fields that are valid for substitution. You can copy and paste such fields into the Text field for the template.

  The fields available for substitution are from the primary business component of the business object specified by the Object field.

  All substitution fields must be enclosed within brackets ([ ]) in the template text.

  For example, available substitution fields for Contact include [Contact.Last Name], while fields for Account include [Account.Name].

  NOTE: For an HTML template, make sure that the enclosing brackets and text of any substitution field text do not include any tagging changes. For example, do not change from bold to regular text within the enclosing brackets of a substitution field. Doing so may prevent proper data substitution.

  If your template must contain a literal bracket character ([or]), insert a backslash character (\) before the bracket to "escape" the bracket character, so it will be correctly interpreted.

  NOTE: If a substitution field is not enclosed in brackets, or if some other notation is used, such as enclosing the field name within percentage characters (%), then Communications Outbound Manager cannot perform substitution for the field.

Compose Template

The following fields are grouped under Compose Template:

- **Subject.** Enter text for a subject line for the template.

  Text in the Subject field can also include substitutions from the Substitutions list, as determined by the specified object (Object field):

  - For a new email message (as opposed to a reply), the text from this field is used in the Subject line of the message.
  - For a reply message in Siebel Email Response, the subject-line text is derived from the subject line of the original inbound email message.
  - For a fax, the text from this field appears on the cover page of the faxed message.

    NOTE: If you are using Captaris RightFax with the Unicode patch, then the subject text will appear on the fax cover page only for ENU (U.S. English) software. For all other languages, the subject text will appear on a second page of the faxed message. For more information, see "Integrating with Fax Systems" on page 357.
  - For a page message the text from the Subject field is not used.

- **Text.** Specify text for the template. As appropriate, you can copy and paste fields from the available substitutions.

  All template text appears at the start of the message body.

  For information about editing and formatting within this field, see "Editing and Formatting Controls for Template Text" on page 275.

Advanced Form

This section describes the template options you specify in the Advanced form.

Template Properties

The following fields are grouped under Template Properties:

- **Name (required).** See "Templates List" on page 280.
- **Channel Type (required).** See "Templates List" on page 280.
- **Delivery Profile.** Select the communications profile to use in order to deliver the template to recipients. The profile must support the specified channel type.

  The delivery profile *must* be specified for any template that will be sent using an outbound communication request. This field is not present on the Simple tab because it is used for outbound communication requests only.

  For more information, see "Configuring Communications Drivers and Profiles" on page 44.
- **Language (required).** See "Templates List" on page 280.
- **Locale (required).** See "Templates List" on page 280.
- **HTML Template.** Check this option to specify that an email or fax template is an HTML template. Leave this box unchecked if the template is plain text. If this box is not checked, then any formatting you apply in the Text field is stripped when you save the template.

  For information about HTML formatting in the Text field, see "Editing and Formatting Controls for Template Text" on page 275.
- **Public.** Check this option to make this template available for all users to specify in the context in which the templates will be used, such as in sending outbound communication requests. Leave this box unchecked if the template is available only to the current user and to administrators.

  NOTE: The Public field is ignored for any template that is specified to be sent using an outbound communication request initiated through Siebel Workflow. Any defined template can be specified.

- **Attach Bookmark.** Includes a Siebel bookmark in communication requests sent using this template.

  A Siebel bookmark is a URL that links to a specific record in the Siebel application. A bookmark included in an email message allows a user to click to navigate directly to this record. If you specify this option, a bookmark is included in outbound messages sent using this template. The bookmark URL is appended to the message body. The bookmark URL functions as a link whether the template is HTML or plain text.

  For example, if you are sending email to account team members, a bookmark URL may be sent providing access to the account record that was specified as the recipient source for the communication request.

  Bookmarks can be included in messages sent using templates of any applicable channel. However, only bookmarks in email messages allow the recipient to click to navigate to the target record in the Siebel application.

  If the recipient user is running the Siebel application, the bookmark record can open in this window or can open in another browser window. If the user is not logged in, the user is prompted to log in before the record is accessed.

  NOTE: A bookmark can be manually inserted in an email message or document by choosing the command Create Bookmark from the applet-level menu.

  Before you can specify to include bookmarks, note the following requirements:

  - The Siebel administrator must specify a value for the WebServer server component parameter for the Communications Outbound Manager. In order to access the bookmarked record, the recipient users must have access to the Web server and Application Object Manager specified using this parameter.

    For more information, see "Administering Communications Outbound Manager" on page 261.

  - For each business component representing a type of record for which bookmarks are to be supported, the Siebel configurator must verify that the user property Default Bookmark View has been appropriately defined.

    This property specifies the Siebel application view in which the bookmark record should open for the end user, who is in this case the email recipient. The specified view should be an end user view that is available to the intended recipients. This user property is already defined for many key business components. For example, for the Account business component, the default bookmark view is Account List View.

    For more information about configuring user properties for business components, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

- **Create Activity.** Check this option to set activity logging for outbound communication requests that include this template.

  If Create Activity is checked for the template, one activity is created for each recipient, after the message is sent to that recipient. For each activity record, the text from the template's Subject field is written to the Description field, and the text from the template's message body is written to the Comments field.

  Sending a message to a large number of recipients generates an equally large number of activity records.

  Each activity that is created will be owned by the user who started the Siebel Server running Communications Outbound Manager. Typically, this is SADMIN.

  NOTE: If you create a new recipient group using a custom business component, and will set the Create Activity flag for an advanced template that uses this recipient group, then you must create a System Activity object on the custom business component in order to specify fields that can be logged with activity logging. See also "Configuring Recipient Groups for Requests and Advanced Templates" on page 167.

  For more information about activity logging, see "Viewing Activity Records for Communication Requests" on page 302.

Pick Available Substitutions

The following fields are grouped under Pick Available Substitutions:

- **Recipient Group.** Select the type of recipient group for the template. The recipients might be, for example, all contacts associated with an opportunity—this particular group is identified as Opportunity Contacts.

  The recipient group specified for the template is optional and is used to obtain values for the Available Fields list.

  The recipient group you specify here only determines which field substitutions are available for the template text. The actual recipient group, and actual recipients, are determined for the communication request where the template is used.

  NOTE: Compare the Recipient Group field on the Advanced tab with the Object field on the Simple tab.

  The Recipient Group picklist displays recipient groups for many major business objects and business components. For predefined values, see "Predefined Recipient Groups" on page 169.

  You can customize the list of values for the Recipient Group picklist in order to add or remove recipient groups. For more information, see "Configuring Recipient Groups for Requests and Advanced Templates" on page 167.

- **Available Fields.** Displays the business component fields that are valid for substitution. You can copy and paste such fields into the Text field for the template.

  The fields available for substitution are from the recipient business component specified by the Recipient Group (such as Contact, for Account Contacts), or from the primary business component in the recipient source business object (such as Account, for Account Contacts). Fields for the primary business component are prefaced by the business component name, followed by a period.

  All substitution fields must be enclosed within brackets ([ ]) in the template text.

  For example, available substitution fields for Account Contacts include [Last Name], from the Contact business component, and [Account.Name], from the Account business component.

  NOTE: For an HTML template (HTML text in the Text field) or an HTML template item, make sure that the enclosing brackets and text of any substitution field text do not include any tagging changes. For example, do not change from bold to regular text within the enclosing brackets of a substitution field. Doing so may prevent proper data substitution.

  If your template must contain a literal bracket character ([or]), insert a backslash character (\) before the bracket to make sure that it will be correctly interpreted.

  NOTE: If a substitution field is not enclosed in brackets, or if some other notation is used, such as enclosing the field name within percentage characters (%), then Communications Outbound Manager cannot perform substitution for the field.

Compose Template

The following fields are grouped under Compose Template:

- **Subject.** Enter text for a subject line for the template.

  Text in the Subject field can also include substitutions from the Available Fields list, as determined by the specified recipient group:

  - For an email message sent using a communication request, the text from this field is used in the Subject line of the message.
  - For a fax sent using a communication request, the text from this field appears on the cover page of the faxed message.

    NOTE: If you are using Captaris RightFax with the Unicode patch, then the subject text will appear on the fax cover page only for ENU (U.S. English) software. For all other languages, the subject text will appear on a second page of the faxed message. For more information, see "Integrating with Fax Systems" on page 357.
  - For a page message, text from this field is not used.

- **Text.** Specify text for the template. As appropriate, you can copy and paste fields from the available substitutions.

  All template text appears at the start of the message body.

  For information about editing and formatting within this field, see "Editing and Formatting Controls for Template Text" on page 275.

Fields for Template Items

This section describes the information you can specify for a template item. The options are organized according to where they are specified.

For descriptions of the fields for templates, see "Fields for Templates" on page 280.

Template Items List

The following options are specified in the Template Items list:

- **Sequence (required).** A sequence number is provided for the template item, using the lowest available number (starting with 1). The sequence number is used in assembling the template items for the communication.

  The default sequence value for a new template item is the lowest number not used by an existing template item for the same template. For template items specified for the message body, messages to be delivered to recipients are constructed as follows:

  - All plain-text template items are concatenated, in sequence, to form the plain-text message body for the email message. (Applies to email, fax, and page templates.)
  - All HTML template items are concatenated, in sequence, to form the HTML message body for the email message. If HTML template text is defined, then this text is appended by the HTML template items for the message body. (Applies to email templates only.)

  Only template items that have the Message Body flag checked form the plain-text message body *or* (for email only) HTML message body.

  For an email message, the recipient's email client will display the HTML version of the message body if it can do so; otherwise, the email client will display the plain-text version of the message body. Sending separate plain-text and HTML versions of the same email message body is called MIME multipart/alternative support. (MIME stands for Multipurpose Internet Mail Extension.)

  NOTE: For a fax template, specify HTML template items as attachments *only*. Do not specify HTML template items for the message body. (The text in the Text field may be either HTML or plain text.)

  You can change the sequence number for a template item, but you cannot specify a sequence number that is in use by another template item. Instead, you must shift the numbers to make space for the one you want to modify. For example, if you have sequences 1 and 2, to swap them you could change 1 to 3, change 2 to 1, then change 3 to 2.

- **Attachment Label.** Specify a label for a template item that is to be an attachment to an email or fax message. In this context, what is meant by an attachment is any template item for which Message Body is not checked.

  For a template item that you are adding as an attachment to an email or fax message, the Attachment Label field provides an optional string to name the attachment. If an attachment label is not specified, the file name and extension are used as the attachment label.

  For an HTML template item that you are adding as an attachment to an email or fax message:

- The Attachment Label field specifies a string that is used for embedding the template item into the HTML. The string must exactly match a reference in the HTML markup, in order for the HTML content to provide recipients access to the template item content, such as an embedded picture.

  The email client for an email recipient, or the email server for your fax gateway, matches these strings in order to correctly locate and display or process each picture, or other embedded attachment.

  For example, for an attachment label "my_picture.jpg" (without the enclosing quotes), the matching HTML reference for this embedded picture file is as follows:

  ```
  <IMG SRC="my_picture.jpg">
  ```

- If an attachment label is not specified, the attachment is a true attachment to the HTML email or fax message, and is not embedded in the HTML message.

NOTE: If an attachment label is specified that does not exactly match a reference in the HTML markup, then email clients, or the email server for your fax gateway, may not be able to access the template item. The template item is an embedded attachment for which the HTML markup provides no access.

For more information, see the description for the Attachment Name field, later in this section.

Embedding pictures or other referenced files in HTML makes an email message larger and makes it take longer to send, or makes a fax take longer to be processed by your fax gateway. For an email message, you can also make a file available on the Internet and include its URL in the HTML. If you do this, do not also add the files as template items. Recipients can view the images only if they are connected to the Internet when they read the email message.

- **Literature Name.** Specify a literature item from the Siebel File System that you want to use for the template item.

  For instructions for adding a literature item for a template item, see "Specifying Template Items for an Advanced Template" on page 277.

  For a literature item you specify, values can be substituted using the business component fields listed in the Available Fields list, if Substitute Values is checked.

  A template item containing a literature item can be specified for the message body for the template, or specified to be an attachment.

  You can drill down on the literature item to navigate to another view where you can open the item.

  For information about adding literature items to the Siebel File System, see *Applications Administration Guide*.

- **Attachment Name.** Specify the name of an operating system file you want to use for the template item. The file can be a text file, an HTML file, a graphics file, or another type of file.

  For instructions for adding a file for a template item, see "Specifying Template Items for an Advanced Template" on page 277.

  Once you have specified a file, its name (without the file extension) is displayed in the Attachment Name field. (The file extension is displayed in the File Type field.)

  Any graphics (picture) file for an HTML message must be added as a file and specified as an attachment to the message—that is, it cannot be specified for the message body. An attachment label must be specified for the file, as described in this section for the Attachment Label field.

  For a file you specify, values can be substituted using the business component fields listed in the Available Fields list, if Substitute Values is checked.

  A template item containing a file can be specified for the message body for the template, or specified to be an attachment.

  NOTE: When you specify a file for a template item, the file is automatically added to the Siebel File System. The operating system file is never referenced after this. When you send the file, you send the version in the Siebel File System.

- **File Type.** (read only) Displays the extension of a file you added. The file name, without the extension, is displayed in the Attachment Name field.
- **File Size.** (read only) Displays the size, in kilobytes, of a file you added.

Template Item Form

The following options are specified in the Template Item form.

Template Item Commands

The following fields are grouped under Template Item Commands:

- **Substitute Values.** Check this option to enable value substitution for the template item. For more information, see the descriptions for the Available Fields list and for the Iteration Child Business Component field.

- **Message Body.** Check this option to specify that the template item's content is used in the message body of the template's message.

  Multiple plain-text or HTML template items marked for the message body will be concatenated into a single message body for each of these types, in the sequence determined by their sequence numbers. See also the description for the Sequence field, earlier in this section.

  NOTE: For a fax template, specify HTML template items as attachments only. Do not specify HTML template items for the message body.

  If a template item is used as part of the message body, it must be either a plain-text or HTML file. Communications Outbound Manager uses the file extension to distinguish whether a template item's file is in HTML format. Any file with an extension containing "htm" (such as .htm, .html, or .shtml) is treated as an HTML file.

  The setting of the HTML Template check box also determines how HTML template items are handled. For details, see "Fields for Templates" on page 280.

  A file or literature item, for which values will be substituted using business component fields, can be specified for the message body or specified as an attachment. (When specified for the message body, files or literature items can be plain-text or HTML files only; other file types are not supported and may produce unacceptable results.)

  If Message Body is not checked, the current template item will be an attachment to the message, using the label you specify in the Attachment Label field.

  For templates of channel type Fax, any template items not specified for the message body are subject to what is supported by the sending fax server. Some fax servers support certain types of attachments only; some may not support any attachments.

  For templates of channel type Pager, any template that has one or more template items that are not specified for the message body will not be sent.

Pick Available Substitutions

The following fields are grouped under Pick Available Substitutions:

- **Iteration Child Business Component.** Select the name of a business component from which you want to include record data.

  The available business components you can specify are direct children of the recipient source business object that was specified by the Recipient Group.

  NOTE: Within the source business object, any business component that you want to be able to specify in the Iteration Child Business Component field must have a link in which the parent business component is the primary business component on the business object.

  If Substitute Values is checked and you specify a child business component, the template item is appended to the message body or attachment once for each record in the child business component, in iterative fashion. Business component field values are substituted for each record.

  A template item can include fields from the child business component for which values will be substituted. The Template Item form displays the available substitutions.

  For example, if you are targeting a template to Account Team Members, and you want to include a list of opportunities for the account, you can choose Opportunity as the iteration child business component for a template item.

  If the file or literature item for this template item includes any of the field names shown in the Available Fields list, then values for these fields, for all opportunities that are children of the account, are included in the message body or attachment. In this example, values for opportunity name and expected revenue figures may be substituted.

- **Available Fields.** Displays the business component fields that are valid for substitution. You can use the Available Fields list for reference, or copy and paste such fields into a template file or literature item that you use for the template item.

  The fields in the Available Fields list are from *one* of the following:

  - The recipient business component specified by the Recipient Group (such as Contact, for Account Contacts), or the primary business component in the recipient source business object (such as Account, for Account Contacts). Fields for the primary business component are prefaced by the business component name, followed by a period. These business component fields are displayed in the Available Fields list if the Iteration Child Business Component field is empty.
  - The business component specified in the Iteration Child Business Component field representing a child (such as Account Note) of the recipient source business object (such as Account). Fields for the iteration child business component are prefaced by the business component name, followed by a period.

  All substitution fields must be enclosed within brackets ([ ]) in the template item's file or literature item.

  For example:

  - Available substitution fields for the Account recipient group (for a simple template) include [Account.Name], [Account.Last Name], [Account.Location], and so on, from the Account business component, which is the primary business component for the Account business object.
  - Available substitution fields for the Account Contacts recipient group (advanced template) include:

  - [Last Name], [City] and so on, from the Contact business component
  - [Account.Name], [Account.Last Name], [Account.Location], and so on, from the Account business component
- Available substitution fields for the Account Note iteration child business component include [Note], [Note Type], and so on, from the Note business component.

NOTE: For an HTML template (HTML text in the Text field) or an HTML template item, make sure that the enclosing brackets and text of any substitution field text do not include any tagging changes. For example, do not change from bold to regular text within the enclosing brackets of a substitution field. Doing so may prevent proper data substitution.

If your template must contain a literal bracket character ([or]), insert a backslash character (\) before the bracket to "escape" the bracket character, so it will be correctly interpreted.

NOTE: If a substitution field is not enclosed in brackets, or if some other notation is used, such as enclosing the field name within percentage characters (%), then Communications Outbound Manager cannot perform substitution for the field.

10 Defining Outbound Communication Requests

This chapter provides information about creating and submitting outbound communication requests, and monitoring the status of outbound communication requests. It includes the following topics:

About Outbound Communication Requests

An outbound communication request is a defined request to send content from communications templates (optionally with field substitution) to recipients associated with business object instances—such as to contacts associated with an account. Requests are sent using email, fax, pager, or other channels, as defined in the specified templates and as supported for your communications infrastructure.

Requests can be created and submitted directly, as described in this section, or created and submitted programmatically using Siebel Workflow and business services.

NOTE: The procedures described in this chapter can be performed either by administrators or by advanced end users.

Prerequisite Recipient Addressing Information

For an outbound communication request to succeed, the business component records, or other associated records, for the recipients of the communication must have the appropriate contact information defined for them. Email address, fax number, and pager fields for contacts, employees, partners, prospects, or other potential recipients must contain valid data.

For information about specifying which business components fields contain addressing information, see "Configuring Recipient Sources Applets" on page 174.

Siebel Views for Working with Outbound Communication Requests

End users can create and submit communication requests in these views in the Communications screen:

- My Outbound Requests
- My Outbound Request Overview

Administrators can create and submit communication requests in these views in the Administration - Communications screen:

- All Outbound Requests
- Outbound Request Overview

End users see only the requests they have created. Administrators see all requests.

NOTE: Procedures in this section assume a non administrator user, using the My Outbound Requests and My Outbound Request Overview views in the Communications screen.

Creating and Submitting Outbound Requests

This section describes how to create and submit an outbound communication request. The procedures are the same for end users and administrators, though the administrators generally use different views for this activity.

Communication requests submitted from Siebel Web Client are processed immediately. Requests submitted from Siebel Mobile Web Client, while you are connected to the local database and disconnected from the enterprise database, are saved until you synchronize. They are then processed for delivery by the Communications Outbound Manager server component.

For information about dividing high-volume requests into subrequests for efficient processing, see "Configuring Subrequests" on page 296.

NOTE: If you are implementing a custom channel type, see also "Configuring Communications List of Values Types" on page 163.

Creating an Outbound Communication Request

This section describes how to create an outbound communication request.

To create an outbound communication request

1. Navigate to Communications > My Outbound Request Overview.

   The My Outbound Request Overview view appears.

2. Add a new record in the Communication Request form.

3 Provide information for the new communication request, as described in "Fields for Outbound Communication Requests" on page 297.

   a Specify a type of recipient group, such as Opportunity Contacts.

   b Specify at least one instance of the recipient source (such as an opportunity) in order to determine the actual recipient group for the request.

4 Add templates for the request, as follows:

   a Click the Request Templates view tab.

   b In the Request Templates list, add a new record.

   c Choose one or more existing templates from the dialog box that appears.

      For an end user, the templates displayed in the dialog box are those for which the primary business component of the associated recipient group matches that of the request's recipient group. The templates listed were either created by that user or marked for public access. For an administrator, all templates are displayed.

      In the requests views, records for templates are read-only. You can modify the template in the end-user or administrator views for templates. For more information, see Chapter 9, "Configuring Communications Templates."

5 Optionally, specify how many subrequests the request is broken into, as described in "Configuring Subrequests" on page 296:

   a Click the Subrequests button on the Communication Request form or the Communication Requests list.

   b In the dialog box that appears, specify a number between 2 and 26, inclusive, to indicate how many subrequests should process the parent request.

6 Submit the request, as described in the following section.

Submitting an Outbound Communication Request

This section describes how to submit an outbound communication request you previously created.

NOTE: If the Communications Outbound Manager component aborts, a communication request that has already been submitted can be resumed automatically upon restarting the Siebel Server. Alternatively, the request can be resubmitted in order to process it from start to finish.

For more information, see "Restarting an Aborted Outbound Communications Manager Component" on page 299, "Monitoring Outbound Communication Request Status" on page 300, and "Monitoring Siebel Server Request Status" on page 301.

To submit an outbound communication request

1 Navigate to Communications > My Outbound Requests.

2 In the Communication Requests list, select the record for a communication request that is ready to be submitted.

   For an end user, the requests displayed are those that were created by that user. For an administrator, all requests are displayed.

3 From the Show drop-down list, choose My Outbound Request Overview.

   The My Outbound Request Overview view appears.

4 Verify that the request has been correctly configured and includes the intended templates. Modify the request, as appropriate.

5 Click Submit in the Communication Request form.

   If the communication request is properly processed by the Server Request Processor, Server Request Broker, and Communications Outbound Manager server components, the request will be passed to the appropriate messaging systems for delivery to the intended recipients.

6 Monitor the request in progress, as appropriate.

   For information about monitoring requests, see "Monitoring Outbound Communication Request Status" on page 300 and "Monitoring Siebel Server Request Status" on page 301.

Configuring Subrequests

An outbound communication request can be broken into subrequests, which can be processed by more than one instance of the Communications Outbound Manager server component in order to achieve greater scalability for requests with a large number of recipients or large processing demands.

You can specify from 2 to 26 subrequests for a given request. Subrequests are divided based on the last name of each recipient.

For example, if 2 subrequests are specified, then all recipients with last names A–M are handled by one subrequest, and those with last names N–Z are handled by the second subrequest. If 3 subrequests are specified, recipients are allocated to three alphabetical groupings (A–I, J–R, S–Z). If 4 subrequests are specified, recipients are allocated to four alphabetical groupings (A–F, G–M, N–S, T–Z), and so on.

Subrequests are distributed to multiple instances of the server component whose name is specified for the request. This approach load-balances your communication requests to achieve scalability objectives—particularly when subrequests are divided among multiple Siebel Server machines. Requests are distributed by Server Request Processor and Server Request Broker.

If you click the Subrequests button after subrequests have already been specified, the number of subrequests is displayed. After a request has been processed, the Communication Subrequests view tab displays a list of the subrequests for the request. The form below the Communication Subrequests list includes the Recipient Search Specification field, which may display information about how the set of recipients for the subrequest was defined.

To configure your system for optimal processing of high-volume outbound communication requests using subrequests, you can designate Siebel Server instances running on multiple server machines to run Communications Outbound Manager. For more information about Communications Outbound Manager, see "Administering Communications Outbound Manager" on page 261.

NOTE: Subrequests cannot be specified for any outbound communication request for which more than one recipient source object has been specified. For example, for a request based on the recipient group Campaign Contacts, one and only one campaign record can be specified if you are to specify subrequests for this request.

Subrequest SQL File Option

Depending on the requirements for your deployment, your company may choose to implement alternative SQL for processing subrequests. You can do this by creating a SQL file to be read in place of the default Siebel method of processing subrequests.

This file must be named commoutboundmgr.cfg, and it must be located in the subdirectory CMS of the Siebel File System directory. For example, depending on your installation, the full path might be as follows:

```
C:\sea78\FS\CMS\commoutboundmgr.cfg
```

If no file exists of this name exists in the designated location, then the default hardcoded SQL from Siebel Systems is used for processing subrequests.

For more information on this issue, refer to Siebel SupportWeb.

Deleting Outbound Communication Requests

You can delete a record for an outbound communication request. An end user can delete any request the user has created. A Siebel administrator can delete any request.

Any request that is being processed (that is, the status has changed from Submitted to Processing) will continue to be processed until completion, even if the request is deleted.

Fields for Outbound Communication Requests

This section describes the fields for outbound communication requests. Several fields are read only, and can be referenced for tracking purposes:

NOTE: Not all fields appear on all of the outbound request views. Check the application user interface for the specific views on which a particular field appears.

- **Description.** Enter a description for the request.
- **Request #.** (read only) A unique ID for each request, which is generated when the request is created.

- **Recipient Group.** Select from a picklist the group of intended recipients for the communication. The recipients might be, for example, all contacts associated with an opportunity—this group is identified as Opportunity Contacts.

  After you specify the Recipient Group here, the Recipient Sources list in My Outbound Request Overview changes to reflect the business object representing the source from which the recipients are derived, such as Opportunities. The recipient source business object is identified using a label displayed on the right side of the applet header area.

  You add records to the Recipient Sources list, which is renamed to reflect the specified business object, to specify one or more instances of the primary business component for this business object—such as particular opportunities.

  The read-only Recipients list, which appears below the Recipient Sources list, shows every individual recipient to whom a separate instance of the communication will be sent—such as the Opportunity Contacts for each opportunity.

  NOTE: The Recipient Group picklist displays recipient group types for many major business objects and business components. For a list of the predefined values and the business object and business component associated with the recipient group, see "Predefined Recipient Groups" on page 169.

  You can customize which values appear in the Recipient Group picklist. For more information, see "Configuring Recipient Groups for Requests and Advanced Templates" on page 167.

- **Default Preference.** Specify a default delivery type to use for communication requests to each individual recipient if Only Send Preference is checked for the request, but no communications preference is specified for the recipient.

  For more information, see the description for Only Send Preference.

- **Only Send Preference.** Check this option to send to each individual recipient only the templates that match the recipient's communications preference, if it is specified, or that match the channel preference specified in the Default Preference field.

  If you construct a communication request that contains templates of more than one channel type, and this option is checked, then an individual recipient can receive communications only for a single channel type.

  The communications preference field affected by the Only Send Preference check box generally corresponds to the setting of the Recipient Preferred Medium Field user property for the business component.

  For example, for both employees and contacts, the Preferred Communications field is the default field containing the channel preference. (For contacts, the caption for this field is Contact Method.) If this user property is not defined, then the preference is retrieved from the Preferred Communications field, if the business component includes such a field. For more information, see *Applications Administration Guide*.

  If the communications preference is not specified for a recipient, and the Only Send Preference option is checked, then the preference specified in the Default Preference field is used for that recipient.

- **Start Time.** (read only) The time that the Communications Outbound Manager server component started processing the request.

- **End Time.** (read only) The time that the request was fulfilled.
- **Created.** (read only) The time that the request was created.
- **Created By.** (read only) The user who created the request.
- **Status.** (read only) The current status of the request. For more information, see "Monitoring Outbound Communication Request Status" on page 300.
- **Status Message.** (read only) Status message text generated by the Communications Outbound Manager server component for the request, if such text was generated. For more information, see "Monitoring Outbound Communication Request Status" on page 300.
- **Comments.** Enter comments about the request.
- **Component Name.** Displays the name of the server component that is to process the outbound communication request.

  The default component name is CommOutboundMgr (for Communications Outbound Manager). If you have configured any new server components based on CommOutboundMgr, you can specify another component name here.

  For example, if you have created and configured the server component CommOutboundMgr2, and want the current request to be processed by an instance of this component, specify this new component name in this field.

  NOTE: If your Siebel application deployment involves different .srf files (Siebel repository files), then run differently named server components, based on CommOutboundMgr, for each different repository file used by one or more Siebel Server. Otherwise, field data may not be correctly substituted in your communications templates.

  For more information, see "Administering Communications Outbound Manager" on page 261. Also see *Global Deployment Guide*.

  NOTE: Subrequests specified for a request may be processed by more than one server task for the server component designated to handle the request, according to how your environment is configured. For more information about subrequests, see "Configuring Subrequests" on page 296.

Restarting an Aborted Outbound Communications Manager Component

If the Siebel Server aborts (or the host machine stops) while the Communications Outbound Manager component is processing a communication request, Communications Outbound Manager creates a file in the Siebel File System containing information about the last message sent.

The file is named CommOutboundMgr*xxx*.crf, where *xxx* is the ID number for the request.

If, at a later time, a request with the same request ID is submitted, Communications Outbound Manager will detect the corresponding file .crf file and resume the request where it left off.

If the request or the template data was modified since the time the request was originally submitted, then the total message output of the resumed request plus the aborted request may not exactly match the output that the original request would have generated had it not been aborted.

You can delete the .crf file if you do not need it. If you delete the .crf file and resubmit the request, then the request will be processed completely from start to finish.

The restart capability described in this section applies only to when the Communications Outbound Manager component aborts while processing a valid communication request. It does not apply to when the component stops processing a request due to error, such as incorrectly specified recipients. In such a case, Communications Outbound Manager does not abort and can continue to receive and process requests.

Monitoring Outbound Communication Request Status

Outbound communication requests initiated by end users or administrators can be monitored in the views for working with outbound communication requests. These views display the status of each communication request that has been run by the Communications Outbound Manager component.

NOTE: It is recommended that you specify text for the Comments field when you create requests. Such comment text can help you to identify or differentiate requests or subrequests you are monitoring.

To monitor outbound communication requests

1. Navigate to one of the following views:
   - Administration - Communications > All Outbound Requests
   - Communications > My Outbound Requests
2. In the Communication Requests list, query the listed communication requests for all records matching certain criteria. Then check the status for these requests.

   For example, you might want to find all email requests with a particular status, all requests created or submitted by a particular user, or all requests submitted to a specific server component.
3. As applicable, in the Communications Subrequests list, browse the listed subrequests for a request that has been submitted. Then check the status for each subrequest.

Status Settings for Outbound Communication Requests

For each outbound communication request record, or subrequest record, detailed status information may be shown in the Status Message field. In addition, the Status field displays one of the following values:

- **In Progress.** The request has been created but has not yet been submitted for processing by the Communications Outbound Manager.
- **Submitted.** The request has been submitted to the Communications Outbound Manager server component. The request will have this status even if the Communications Outbound Manager has not yet received it.

- **Processing.** The Communications Outbound Manager has received and is processing the request.
- **Done.** The Communications Outbound Manager has successfully completed processing the request.
- **Incomplete.** The request could not be fully processed because it did not include all necessary information—for example, one or more recipients could not be resolved, no templates were specified for the request, or a template was invalid.

  A request with subrequests may have the status incomplete if only some of the subrequests were correctly processed. Each subrequest has its own status. Resubmit any subrequest that was not processed.

  NOTE: If address information, such as the email address, is missing for any of the recipients of a communication request, the request status after processing will be Incomplete. This status does not, however, mean that Communications Outbound Manager failed to deliver to recipients for which addresses were available.

Monitoring Siebel Server Request Status

Outbound communication requests initiated by end users or administrators can be monitored in the Administration - Server Management screen, under the Jobs link. This view displays the status of a request that is being sent as a server request to the Communications Outbound Manager component for processing as a communication request.

When a user or an administrator clicks the Submit button to submit an outbound communication request to Siebel Communications Server, a Siebel Server request is created.

The Server Request Processor and Server Request Broker server components initially process the request, then pass it to the Communications Outbound Manager server component for further processing.

NOTE: Server Request Processor and Server Request Broker must be running for the Communications Outbound Manager to operate. The processing of a request is subject to the availability of these components. Delays in request processing may be due to intermittent availability of these components.

An administrator can monitor outbound communication requests as server requests, using the Jobs view.

In addition, the administrator can perform several actions on a request:

- **Hold.** Hold back a request from the queue.
- **Resume.** Place a request back in the queue.
- **Cancel.** Cancel a request that is currently in the queue.

For more information about server requests, see *Siebel System Administration Guide*.

To monitor outbound communication requests as server requests

1. Navigate to Administration - Server Management.
2. Click the Jobs link.
3. In the Jobs list or the Job Detail form, query for all server request records in which the value for the Component/Job field is Communications Outbound Manager (or the component name for your implementation).

Server Request Status Settings

For each outbound communication request record, the Status field in the Jobs view displays one of the following values:

- **Queued.** The request has been queued by the Server Request Processor and Server Request Broker, and will be passed to the Communications Outbound Manager.
- **Active.** The request is currently being processed.
- **Success.** The request has been successfully processed.
- **On Hold.** An administrator has placed the request on hold. Placing a request on hold removes it from the queue.
- **Cancelled.** An administrator has cancelled the request.

  NOTE: Once an outbound communication request has been submitted for processing, it can be cancelled only as a server request, from the Jobs view. A request can be cancelled only if the request has a status of On Hold or Queued.
- **Error.** An error occurred during processing, and the Server Request Processor and Server Request Broker could not successfully pass the request to the Communications Outbound Manager.

  NOTE: If a request has the Error status, use one of the communication request views to obtain more detailed status information. For more information, see "Monitoring Outbound Communication Request Status" on page 300.

Viewing Activity Records for Communication Requests

Users and administrators can monitor and troubleshoot communication requests that include particular templates by viewing activity logs or Siebel Communications Server logs.

You specify activity logging by checking the Create Activity check box when creating an advanced template. Activity records are viewed in the Activities screen. For more information, see "Fields for Templates" on page 280.

The procedure below, for the All Templates view in the Administration - Communications screen, also applies to the My Templates view in the Communications screen.

If Create Activity is checked for the template, one activity is created for each recipient, after the message is sent to that recipient.

The message body text (and some other data, such as the sender name and the To line) is written to the Email Body field, up to a size of at least 16,000 characters (actual length depends on your database).

NOTE: When a message is sent to a large number of recipients, an equally large number of activity records is generated.

To set activity logging for any requests using a particular template

1. Navigate to Administration - Communications > All Templates.
2. In the Templates list, select a template that will be included in an outbound communication request.
3. Click the Advanced tab.
4. Check the Create Activity check box.

   Any outbound communication requests that include this template will generate activity records.

To view activity data for requests

1. Navigate to Activities > Activity List.
2. Choose All Activities.
3. In the All Activities view, query the listed activities by searching for all relevant activity records.

   For example, the Activity Type differentiates Communications Outbound Manager activities as follows:

   - For activities logged for email templates, Email - Outbound is the activity type.
   - For activities logged for fax templates, Fax - Outbound is the activity type.
   - For activities logged for pager templates, Paging is the activity type.

11 Communications Operations for End Users

This chapter provides information about end-user operations relating to communications capabilities. It includes the following topics:

Most of the tasks described in this chapter are typically performed by end users such as call center agents who are primary users of communications features. These tasks may also be performed by users such as call center managers or communications administrators.

NOTE: End users are not expected to directly use this guide. Your Siebel application implementation may differ from what is described in this chapter. This chapter contains references to administration topics that relate to the end-user operations described here. Chapter 9, "Configuring Communications Templates," and Chapter 10, "Defining Outbound Communication Requests," contain additional tasks that may be performed by end users.

Setting Communications User Preferences

The User Preferences screen contains two sets of communications-related settings, Outbound Communications and Communications. Users can modify, or may need to modify, these settings depending on the needs of your business or on the users' preferences.

The Communications options also includes a list of the agent's ACD queues, which the agent can selectively log in to or log out of.

Preferences are stored separately for each user, and are stored separately based on Siebel client type:

- Preference settings specified in the Siebel Web Client are stored on the Siebel Server machine.
- Preference settings specified in the Siebel Mobile Web Client are stored on the user's local machine.

NOTE: It is recommended that end users only change preference settings at the direction of the system administrator or call-center administrator.

Specifying Communications Preferences

This section describes how to specify communications preferences.

To specify communications preferences

1. From the application-level menu, choose Tools > User Preferences.

   The User Preferences screen is displayed.

2. From the link bar, choose either Outbound Communications or Communications.

   The appropriate set of communications preferences are displayed.

3. Specify the preferences. Preference settings are described in the following sections:
   - "Preference Settings for Outbound Communications" on page 306
   - "Preference Settings for Communications" on page 310
   - "Logging In to or Out of an ACD Queue" on page 314
4. As necessary for the particular setting, log out of the Siebel application and log in again.

   The descriptions for each user preference setting indicate whether this step is required.

Preference Settings for Outbound Communications

This section describes in detail the preferences in the Outbound Communications options of the User Preferences screen. The preference settings are grouped by category. Each preference applies only to certain communications features.

Send Email Preferences (Also for Some Other Send Commands)

The following are the outbound communications preferences in the Send Email category. These preferences apply to using some or all of the Send commands: Send Email, Send Fax, or Send Page. Also refer to the settings under the Advanced Features category.

- **Upon Sending Messages Generate.** Specifies activity generation for messages sent using the Send Email, Send Fax, and Send Page commands. The drop-down list displays the following options:
  - **Public activities.** Sets the Internal flag to FALSE for activity records generated by sending outbound messages. The Internal flag is used to restrict visibility of activity records.
  - **Private activities.** Sets the Internal flag to TRUE for activity records generated by sending outbound messages.
  - **No activities.** Specifies that no activity records are generated by sending outbound messages.

  For more information, see "Creating Activities for Send Commands" on page 334.

- **Default Profile.** Specifies the default communications profile for a user to use, for the Send Email and Send Fax commands.

  The default profile information is copied to the From field in the Send Email and Send Fax windows. Alternatively, agents can specify the profile directly.

  The profile provides access to the communications driver (typically, the Internet SMTP/POP3 Server driver) that serves as the interface from the Siebel software to the email/fax server. For the Send Email and Send Fax commands, the profile handles outbound communications using the SMTP protocol.

  Which profiles are available to choose from depends on the visibility applicable to the user. A profile is visible in the following cases:

  - The user created the profile for personal use, using the My Profiles view in the Communications screen.
  - The user's responsibility is one of the responsibilities associated with the profile.

- **Default Recipient Class.** Specifies the default choice for where recipient email or fax address information will be populated, for the Send Email and Send Fax commands. Choices are To, Cc, and Bcc.

  For example, an individual user might generally want to populate recipients to the To field. However, depending on the user's typical activities or workflow and on your company's business needs, it may be more appropriate to populate recipients to the Cc or Bcc field instead of the To field.

- **Email Client.** Specifies whether to use a supported third-party email client application for the Send Email command. If a third-party option is not specified, then the user can use the native Siebel email client. (You do not need to specify anything here if the administrator has made a global setting.)

  - **Lotus Notes.** Specify that you use Lotus Notes for the Send Email command.
  - **Microsoft Outlook.** Specify that you use Microsoft Outlook for the Send Email command.
  - **Siebel Email Client.** Do not use a third-party email client. Use the email client option provided by Siebel Systems. This option is also referred to as the "native Siebel email client."

  Administrators: Each option requires the administrator to perform additional configuration and integration steps. For details, see "Interfacing with Email and Fax Servers" on page 355 and "Configuring Client-Side Integration for Send Email" on page 383.

  NOTE: The Send Fax command does not support using Lotus Notes or Microsoft Outlook.

- **Siebel/Outlook Form.** If Microsoft Outlook is specified for the Email Client setting, then you can specify the Outlook form you use. The name specified here must match the name of the form the administrator has installed and specified. (You do not need to specify anything here if the administrator has made a global setting.)

- **Siebel/Lotus Form.** If Lotus Notes is specified for the Email Client setting, then you can specify the Notes form you use. The name specified here must match the name of the form the administrator has installed and specified. (You do not need to specify anything here if the administrator has made a global setting.)

- **Default Message Format.** Specifies whether email or fax messages composed in the Send Email or Send Fax windows can be formatted and sent using HTML, or are plain text only.

  Choose the option HTML to enable HTML editing controls for Send Email and Send Fax, or specify Plain Text to edit and send messages as plain text only.

  When the default message format is HTML, formatting is preserved for email and fax messages you send. Email messages are sent as HTML messages. Fax messages are, of course, sent as facsimiles.

  This preference also specifies which templates can be selected for Send Email and Send Fax, based on the setting of the HTML Template check box for each applicable template. HTML templates are listed when the default message format is HTML, and plain-text templates are listed when the default message format is plain text.

  For more information, see:

  - "Sending Email Using Lotus Notes or Microsoft Outlook" on page 327
  - "Sending Email Using the Native Siebel Email Client" on page 325
  - "Sending Faxes" on page 329
  - "Editing and Formatting Controls for Send Email and Send Fax" on page 332
  - Chapter 9, "Configuring Communications Templates"

Email Response Preferences

The following are the outbound communications preferences in the Email Response category. These preferences apply only to using Siebel Email Response. Also refer to the settings under the Advanced Features category.

For more information about some of these preference settings and about using Siebel Email Response, see *Siebel Email Response Administration Guide* and *Applications Administration Guide*.

- **Default Greeting Template.** Specifies a communications template (for the email channel) to use as a greeting for an email reply sent using Siebel Email Response. The template content is inserted automatically into the message.
- **Default Closing Template.** Specifies a communications template (for the email channel) to use as a closing for an email reply sent using Siebel Email Response. The template content is inserted automatically into the message.

- **Default Message Format.** Specifies whether email replies to inbound messages can be formatted and sent using HTML, or are plain text only.

  Choose the option HTML to enable HTML editing controls in the Outbound Message form, or specify Plain Text to edit and send messages as plain text only.

  When the default message format is HTML, formatting is preserved for email messages you send, and messages are sent as HTML messages.

  This preference also specifies which templates are displayed in the Body drop-down list in the Outbound Message form, based on the setting of the HTML Template check box for each applicable template. HTML templates are listed when the default message format is HTML, and plain-text templates are listed when the default message format is plain text.

  For more information, see Chapter 9, "Configuring Communications Templates."

- **Include Original Message in Reply.** Specifies if the content of the original inbound message to which an agent is replying is included in the reply.
- **Remain on Same View After Send (Cancel).** Specifies that, after clicking Send or Cancel in the Outbound Message form, the agent prefers to stay in this view, rather than to return to a display of the Communications list. If the agent clicked Send, the next record is displayed from the current list of communications.

Advanced Features Preferences

The following are the outbound communications preferences in the Advanced Features category. These settings affect both the Send Email command and email replies you compose using Siebel Email Response.

For more information, see:

- "Sending Email Using Lotus Notes or Microsoft Outlook" on page 327
- "Sending Email Using the Native Siebel Email Client" on page 325
- Chapter 9, "Configuring Communications Templates"

For more information about using Siebel Email Response, see *Siebel Email Response Administration Guide* and *Applications Administration Guide*.

- **Language.** Specifies the language that determines which templates may be displayed in the Body drop-down list for each of the Send commands or for the Outbound Message form for email replies.

  Templates are displayed only if the language specified here matches that specified for the template. Template filtering is based on both language and locale, and on other factors. For details, see "Template Visibility and Access" on page 270.

  If this field is blank, then template filtering as described above refers to the language in effect for the Application Object Manager.

  This setting can be overridden at runtime in the window for the Send command or in the Outbound Message form, in order to display other templates.

- **Locale.** Specifies the locale that determines which templates may be displayed in the Body drop-down list for each of the Send commands or for the Outbound Message form for email replies.

  Templates are displayed only if the locale specified here matches that specified for the template. Template filtering is based on both language and locale, and on other factors. For details, see "Template Visibility and Access" on page 270.

  If this field is blank, then template filtering as described above refers to the locale in effect for the Application Object Manager.

  This setting can be overridden at runtime in the window for the Send command or in the Outbound Message form, in order to display other templates.

Preference Settings for Communications

This section describes in detail the preferences in the Communications options of the User Preferences screen. The preference settings are grouped by category. Each preference applies only to certain communications features.

General Preferences

The following are the communications preferences in the General category. These preferences apply to users who are part of a communications configuration. Such users typically handle voice calls using Siebel CTI, or receive inbound work items using Siebel Email Response.

- **Receive Screen Pop.** If a user should not be the recipient of screen pops upon communications events, such as receiving a new or transferred work item, the user can clear this check box to disable receiving screen pops.

  If this option is not checked, no screen pops will be received. By default, each user receives screen pops.

- **Send Screen Pop.** When this check box is checked, attaching Siebel client context data is enabled for work item transfers and conferences and for internal calls placed by the agent.

  If this option is not checked, context data will not be attached—no screen pops will be sent. By default, each communications user sends screen pops.

  For more information about requirements for screen pops, see "Event Responses" on page 110.

- **Bring Siebel to Front.** This drop-down list specifies whether, or when, the Siebel client window is restored from being minimized and moved to the front of the screen on a communications event such as an inbound work item.

  NOTE: The behavior of this feature varies on Microsoft Windows platforms. In some cases, a non minimized Siebel client window may not move to the front. In this case, restore the window from the taskbar.

  By specifying either On All Incoming Work Items or On Matching Events, the agent can make sure that the Siebel application window will be displayed when it is needed—to use the communications toolbar or other features, or to view screen pops. Screen pops, if they are enabled, will still be generated regardless of the setting of the Bring Siebel to Front option.

  The options are:

  - **On All Incoming Work Items.** Brings the Siebel client window to the front on all communications events received at the agent's machine that represent incoming work items. This option is the default and is best for general use by agents running a single instance of the Siebel client.
  - **On Matching Events.** Brings the Siebel client window to the front on communications events that match that instance of the Siebel client.

    This option should be used only by agents running multiple instances of the Siebel client that are each enabled for session communications, but that have different settings, such as for language or locales. In such a situation, different event handlers may be defined, for example, for each language or locale. Use this setting only at the instruction of your administrator.

    Administrators can find more information by referring to the description of the MaxCommToolbars configuration parameter, in "Specifying Parameters for Communications Configurations" on page 52.
  - **Off.** Does not bring the Siebel client window to the front on communications events. The window may remain hidden behind another window or minimized.

- **Auto Login to Call Center at Startup.** When this check box is checked, an agent automatically logs into the call center's communications systems when starting the Siebel client or connecting to the Siebel Database.

  What the agent logs into depends on how your communications environment has been deployed and how login commands are configured in the communications configuration. Agents can log in to ACD queues or Siebel Universal Queuing. Autologin can make it easier for your users to begin receiving inbound work items from these sources.

  Voice agents who are authorized users for more than one teleset may be best advised not to use autologin, so they can verify which extension they are using before logging in.

  NOTE: Generally, end users should change this setting only at the direction of the system administrator or call-center administrator.

  Whether this control can be set by agents depends on the setting of the AutoLogin configuration parameter. Administrators can find more information under "Specifying Parameters for Communications Configurations" on page 52 and "Configuring Communications Log In and Log Out" on page 201.

Sound Settings

The following are the communications preferences in the Sound category. These preferences apply to users who are part of a communications configuration. Such users typically handle voice calls using Siebel CTI, or receive inbound work items using Siebel Email Response:

- **Enable Sound.** Lets the agent specify if a sound file will play when an inbound work item arrives. By default, no sound file plays.

  If Enable Sound is checked, when a work item arrives, if no valid sound file is specified or the specified file does not exist, the computer may beep.

- **Sound File.** Lets the agent specify which sound file to play, when Enable Sound is checked. For example, you can specify ringin.au or another such file that is valid for your environment.

  The default sound file, shown as files/ringin.au in the User Preferences screen, is installed in `webmaster\files\`*language_code*, in the Siebel Server installation directory where *language_code* represents the language code for the installed software, such as ENU for U.S. English.

  NOTE: New or updated files on the Siebel Server are automatically populated to the Web server whenever the Web server (where the Siebel Web Server Extension is installed) is restarted. The Sound File user preference value refers to a "files" directory on the Web server, where files from the above Siebel Server location are copied to. Files can also be manually updated from the Siebel Server without restarting the Web server. For more information, see *Security Guide for Siebel Business Applications*.

  Alternatively, you can specify the full path for a sound file that is to be loaded, if the file is located in a different directory, *and* if the Java applet for the communications toolbar is signed. For example, the following may be valid paths for specifying a sound file on a user's local system or on a network location:

  - e:\dev\ringin.au (for Microsoft Windows)
  - \\host\dev\ringin.au (for Microsoft Windows)
  - file://e:\dev\ringin.au (for Microsoft Windows)
  - http://*network_path*/ringin.au (for Microsoft Windows or UNIX)
  - /usr/ringin.au (for UNIX)

Teleset and Configuration Settings

The following are the communications preferences in the Teleset and Configuration category. These preferences apply to users who are part of a communications configuration. Such users typically handle voice calls using Siebel CTI, or handle other types of inbound work items through the communications toolbar.

The first two options described apply only to agents who handle voice calls using Siebel CTI.

- **Teleset.** Lets the agent specify the teleset to use the next time the agent logs in.

  The telesets an agent can choose from are those that have been associated with the agent, and will have been specified when the teleset, agent, and extension data was entered in the views in the Administration - Communications screen. The Standard Extension field lists extensions for the currently selected teleset.

  An agent can choose a teleset from the Teleset field only if at least one teleset has been associated with the agent, and the agent is not using a hoteling teleset.

  If the agent has logged into a hoteling computer, the current hoteling teleset is shown in this field—the agent cannot change it.

  NOTE: After you choose a different teleset and extension, you must log out and log in again in order for the new extension to be in effect.

  Administrators can find more information under "Specifying Telesets" on page 62.

- **Standard Extension.** Lets the agent specify the standard extension to use, from the teleset selected in the Teleset field, the next time the agent logs in.

  The available extensions to choose from are those of the current teleset, and will have been specified when the teleset, agent, and extension data was entered in the views in the Administration - Communications screen. Standard extensions (of type "S"—for standard DN) are listed for each teleset the agent is assigned to, or for each hoteling teleset.

  If the agent is using a computer associated with a hoteling teleset, the extensions of this teleset are shown in this field.

  NOTE: After you choose a different extension, you must log out and log in again in order for the new extension to be in effect.

  Administrators can find more information under "Specifying Telesets" on page 62.

- **Configuration.** This option specifies which communications configuration is in effect for the agent.

  This field lists all configurations with which the agent has been associated. If this option has not previously been chosen, the default configuration displayed here is the agent's primary configuration, as specified by the administrator. Otherwise, the configuration displayed is the last configuration used.

  Although an agent may be specified for multiple configurations, only one configuration can be in effect at one time.

  NOTE: After you choose a different configuration, you must log out and log in again in order for the new configuration to be in effect. Just before you log out to end a session for any configuration that is not your regular configuration, change this field back to the configuration you regularly use.

  Administrators can find more information under "Specifying Agents" on page 59.

Logging In to or Out of an ACD Queue

This section describes how to log in to or out of an ACD queue. This procedure is performed in the Communications options of the User Preferences screen.

NOTE: This section applies only to users who handle voice calls (Siebel CTI users) and receive inbound calls routed from one or more ACD queues. Logging in to ACD queues is not always necessary for voice call implementations, depending on the switch type. Consult your administrator.

Use the following procedure with the Log In and Log Out buttons on the communications toolbar. Using these buttons, you can log in to or out of all ACD queues designated as primary queues for you, or log in to or out of other communications systems such as Siebel Universal Queuing.

In the Agent Queues list in the User Preferences screen, you can log in to or out of individual queues selectively, including those which are not designated as primary.

For more information, see "Using the Communications Toolbar" on page 314. Administrators can find more information in the section "Configuring Communications Log In and Log Out" on page 201.

To log in to or out of an ACD queue

1. From the application-level menu, choose Tools > User Preferences.

   The User Preferences screen is displayed.

2. From the link bar, choose Communications.

   The Agent Queues list appears, below the Communications form. All ACD queues with which the agent is associated are listed, and the current login status for each queue is indicated.

3. To log in to a queue, select the record for the queue, then click Login in the Agent Queues list.
4. To log out of a queue, select the record for the queue, then click Logout in the Agent Queues list.

Using the Communications Toolbar

The communications toolbar allows you to manage many types of inbound and outbound communications work items. Channel types can include voice, email, fax, and others.

Supported channel types depend on the Siebel and third-party products your company is using and on your job function. Your company may support types of communications work items not listed here.

The toolbar buttons can be configured based upon the Siebel modules and third-party communications systems available to you. The communications toolbar is used by various Siebel modules that support communications functions.

The Siebel administrator or configurator can customize the communications toolbar, such as to modify the command a button invokes or add buttons for new commands. For details, see "About Communications Toolbar Configuration" on page 151.

The agent can use a menu command to refresh the status of the communications toolbar. For details, see "Menu Command for Refreshing the Communications Toolbar" on page 322.

NOTE: If a record is being created or edited, and has not been committed, then clicking a button on the communications toolbar might not perform the desired function correctly. You will not be able to save the record if some required fields have not yet been filled in appropriately. Before performing an action such as to transfer a call (and send a screen transfer), agents should complete and commit all changes or undo the record, as appropriate.

Communications Toolbar Controls

The communications toolbar controls work together to allow you to perform communications tasks within your Siebel application. Toolbar controls have ToolTips, so you can point to any control to learn its name or to determine its current state or availability:

NOTE: All communications features in the Siebel application user interface are subject to the configuration decisions for your company's implementation.

- **Channel Type Indicator.** Indicates the channel type of the active work item, such as email, phone, and so on. If there are no active work items, the icon will be blank.
- **In-Queue Time Indicator.** Indicates, in the ToolTip text, how long the active work item has been assigned to you (in-queue time).
- **Elapsed Time Indicator.** Indicates the working time for the active work item, in the form *HH*:*MM*:*SS*, where *HH* is the hours, *MM* is the minutes, and *SS* is the seconds. This is the time elapsed since a work item became active, such as when an inbound call was answered.
- **Text Input Field.** Allows you to enter data such as the extension of a person to whom you want to transfer a voice call or of a person to whom you want to make a new call.

  NOTE: Keyboard shortcuts for communications commands do not use as input any data entered into this field.
- **Initiate Work Item.** Allows you to initiate a communications work item. Use the pop-up menu to initiate a work item of the desired channel:
  - Make Call (voice/phone channel)
  - Send Email (email channel)
  - Send Fax (fax channel)
  - Send Page (pager channel)
- **Accept Work Item.** Indicates the channel type of an active inbound communications work item that has arrived. You can click the main button to accept the work item, or use the pop-up menu.
  - Accept Call (voice/phone channel)
  - Accept Email (email channel)
  - A multichannel icon indicates that work items of multiple channels have arrived, such as a voice call and an email message. Use the pop-up menu to accept an individual work item of a specific channel.

- **Release Work Item.** Allows you to release or disconnect the active work item.
- **Blind Transfer.** Allows you to perform a one-step transfer of a work item such as a voice call.
- **Consultative Transfer.** Allows you to initiate, and to complete, a two-step transfer of a voice call.
- **Conference.** Allows you to initiate, and to complete, a conference call.
- **Retrieve Call.** Allows you to retrieve the original call when the intended recipient declines a consultative transfer or conference call.
- **Pause Work Item.** Allows you to pause the active work item, such as to put a voice call on hold.
- **Work Items list.** Allows you to select a paused work item on which to resume work.
- **Resume Work Item.** Allows you to resume work on a paused work item selected from the Work Items list.
- **Forward Work Items/Cancel Forward.** Allows you to forward inbound work items, such as forwarding voice calls to a different extension, and to cancel forwarding.
- **Change Ready State.** For each supported inbound channel type, allows you to specify that you are unavailable to accept work items (Not Ready state), or that you are available again (Ready state). Use the pop-up menu to change the state for a specific channel. For example:
  - Ready/Not Ready for Call (voice/phone channel)
  - Ready/Not Ready for Email (email channel)
- **Log In.** Allows you to log in to a communications system such as an ACD queue or Siebel Universal Queuing.
- **Log Out.** Allows you to log out of a communications system such as an ACD queue or Siebel Universal Queuing.

Logging In to the Communications System

If this capability is supported by your company's call center implementation, you can log in to and out of elements of your communications system by using the Log In and Log Out buttons.

Your Siebel application may be configured to log you into ACD call queues or Siebel Universal Queuing, and may be configured to support automatic log in or automatic log out. For voice calls, these capabilities depend on the CTI middleware your company uses. Siebel CTI Connect (based on Intel NetMerge) supports automatic log in and automatic log out.

If the option Auto Login to Call Center at Startup is checked in your communications user preferences, starting the Siebel application automatically logs you in. Alternatively, automatic login may be set by the administrator in the communications configuration.

If automatic log out is set by the administrator, exiting the Siebel application automatically logs you out of applicable communications systems, such as ACD queues.

Use Log In and Log Out commands and the Auto Login to Call Center at Startup setting as instructed by your call center manager.

For more information about the Auto Login to Call Center at Startup user preference, see "Preference Settings for Communications" on page 310.

For information about logging in to or out of ACD queues selectively, see "Logging In to or Out of an ACD Queue" on page 314.

To log in to the communications system manually

- After starting the Siebel application, click Log In.

To log out of the communications system

- Click Log Out.

Receiving Inbound Work Items

When an inbound work item is routed to you, the Accept Work Item button blinks, and you can accept and begin working on the item.

The Accept Work Item button shows what type of work item is coming in: a phone icon represents a voice call, an envelope icon represents an email message, and so on.

When an inbound work item arrives, the following events may also occur:

- Customer data may be displayed in the customer dashboard.
- A screen pop with relevant data may appear, or you may need to navigate to the appropriate part of the Siebel application.

To accept an inbound work item

- Click Accept Work Item when the pause blinks to indicate the arrival of a work item.

You can have multiple work items at the same time, such as voice calls on hold or other paused work items. However, only one work item is active at a time. The channel type of the active work item is indicated on the left side of the communications toolbar.

To release the current work item

1. Select the work item from the Work Items list.
2. Click Release Work Item, as appropriate for the channel type for the work item.

NOTE: If you are working with an activity record or another type of record that is automatically associated with a work item, you must save any changes to the record *before* you release the work item. In addition, in order to make changes after releasing the work item, you must first refresh the record. Otherwise, changes you make to the record may be lost.

Initiating Work Items

You can initiate communications work items, such as voice calls or email messages, from the communications toolbar. The Initiate Work Item button allows you to choose the channel type for the work item.

NOTE: Using the Send Email, Send Fax, or Send Page commands in the File menu is equivalent to using the communications toolbar to initiate these activities. For more information, see "Sending Email, Fax, and Page Messages" on page 324.

To initiate a work item

1. Optionally, specify the recipient for the work item by selecting a contact or employee record, or by entering a phone number into the text input field.

   NOTE: Recipient information entered into the text input field of the communications toolbar applies to initiating voice calls only. It does not apply to initiating email, fax, or page messages.

2. Do *one* of the following:

   - If no work item has been initiated within the current session, click Initiate Work Item to initiate a work item of the channel type that corresponds to the current context.
   - If a work item of a particular channel was previously initiated, click the Initiate Work Item button to initiate another work item of the same channel type.
   - If you want to choose the channel type explicitly, click the arrow to the right of Initiate Work Item, then select a supported channel type from the displayed menu.

3. If you did not specify contact information first, do so now in the dialog box that appears.

To release the current work item

1. Select the work item from the Work Items list.
2. Click Release Work Item, as appropriate for the channel type for the work item.

NOTE: If you are working with an activity record or another type of record that is automatically associated with a work item, you must save any changes to the record *before* you release the work item. In addition, in order to make changes after releasing the work item, you must first refresh the record. Otherwise, changes you make to the record may be lost.

Transferring or Conferencing Work Items

You can transfer a current work item to another person (such as when you need to escalate an item to a supervisor, or conference another person into a customer call). A voice call can also be transferred to an ACD call queue, depending on your call center.

Three transfer and conference operations are supported:

- Blind transfer (one-step transfer)
- Consultative transfer (two-step transfer, voice calls only)

- Conference (two steps, voice calls only)

NOTE: If a screen transfer of the transferring agent's current record will be sent to the receiving agent of a transferred work item, then any changes to this record must be saved before the agent performs the transfer. If a single record only is displayed in a list, the transferring agent must save changes explicitly before transferring.

To perform a blind transfer on the active work item

1. Specify the person to whom you want to transfer the active work item by selecting an employee record or by entering contact information, such as phone extension, in the text input field.
2. Click Blind Transfer.

   The work item is released immediately.

To perform a consultative (two-step) transfer on the active voice call

1. Specify the person to whom you want to transfer the active voice call by selecting an employee record or by entering the recipient's extension in the text input field.
2. Click Consultative Transfer.

   The current call is paused, and the transfer recipient is dialed.
3. Do *one* of the following:
   - When the transfer recipient answers and indicates acceptance of the transfer, click Consultative Transfer again (toggle the button) to complete the transfer and release the call.
   - If the transfer recipient does not accept the transfer, click Retrieve Call to retrieve the call.

To create a conference for the active voice call

1. Specify the person with whom you want to conference the active voice call by selecting an employee record or by entering the recipient's extension in the text input field.
2. Click Conference.

   The current call is paused, and the conference recipient is dialed.
3. Do *one* of the following:
   - When the conference recipient answers and indicates acceptance of the conference, click Conference again (toggle the button) to complete the conference and include all parties on the active call.
   - If the conference recipient does not accept the conference, click Retrieve Call to retrieve the call.
4. Repeat Step 1 through Step 3 to add participants, as appropriate.

Pausing and Resuming Work Items

You can pause the active work item, such as putting a voice call on hold, and you can resume a paused work item.

To pause the active work item

- Click Pause Work Item.

  A voice call is put on hold, or a work item of another channel is paused, until you resume it.

To resume a paused work item or activate a different item

1. If you have an active work item, pause it, as described above.
2. Select a work item from the Work Items list.
3. Click Resume Work Item to activate the selected work item.

Forwarding Work Items

You can have inbound work items of supported channel types forwarded to you, such as having voice calls forwarded to you at a different extension. Only work items subsequently routed to you will be forwarded. The toolbar button is a toggle, where the button name changes depending on the state.

To specify forwarding for inbound work items

1. In the text input field, enter the contact information, such as a phone extension, where you want your work items forwarded.
2. Click Forward Work Items.
3. Toggle the button to cancel forwarding.

Changing Ready State

You can specify that you are unavailable to receive new inbound work items of each supported channel type (such as when you need time to complete after-call work, or you are taking a break). For each channel type, new work items will not be assigned to you while the Not Ready state is set for that channel.

You can choose Change Ready State separately for each channel type, or for work items of all supported channel types. The toolbar button is a toggle.

If all channels are in Not Ready state, the button is toggled down (appearing to be depressed). If any, or all, channels are in Ready state, the button is toggled up.

To indicate that you are not ready to receive inbound work items

- Do *one* of the following:
  - Click Change Ready State to indicate that you are not ready to receive any inbound work items of all channel types for which you are eligible.
  - Click the arrow to the right of Change Ready State, then choose one or more options from the displayed menu to indicate that you are not ready to receive work items for the applicable channel types.

NOTE: If you click the Change Ready State button to set the Not Ready State for all channel types, but the button does not display as "toggled-down" (depressed), this may be due to a configuration error or some other issue for one or more channels. Please report this issue to your supervisor. You can still choose the Not Ready State for individual channels, as described in the second bullet item above.

To indicate that you are ready to receive inbound work items

- Do *one* of the following:
  - Click Change Ready State to indicate that you are ready to receive any inbound work items of all channel types for which you are eligible.
  - Click the arrow to the right of Change Ready State, then choose one or more options from the displayed menu to indicate that you are again ready to receive work items of the applicable channel types.

Using Communications Menu Commands

This section describes using the Communications submenu and communications commands in the applet-level menus.

The Communications submenu, which is a submenu of the Tools application-level menu in a communications-enabled Siebel application, contains several communications commands for agents to use. The applet-level menus also contain communications commands. Some of the menu commands may perform functions equivalent to buttons in the communications toolbar.

The communications-related menu options support context sensitivity. Agents can initiate several communications operations that extract data from Siebel Database records, such as phone numbers for contacts or service requests, in order to initiate communications or take actions affecting the current communications work item.

NOTE: If a record is being created or edited, and has not been committed, then choosing a communications menu command might not perform the desired function correctly. You will not be able to save the record if some required fields have not yet been filled in appropriately. Before performing an action such as to transfer a call (and send a screen transfer), agents should complete and commit all changes or undo the record, as appropriate.

To choose communications commands

- Do *one* of the following:

- From the application-level menus, choose Tools > Communications, then choose one of the displayed submenu commands.
- From the applet-level menu, choose one of the displayed communications commands.

Example commands may include Answer Call, Associate, Blind Transfer, View Work Item, and so on.

The available commands in these menus vary according to the communications configuration in effect for the agent. Administrators can customize the content and functioning of the Communications submenu and the applet-level menus by working with the commands in the All Commands view in the Administration - Communications screen. For more information, see "Configuring Communications Menu Commands" on page 160.

NOTE: If you are working with an activity record or another type of record that is automatically associated with a work item, you must save any changes to the record *before* you release the work item. In addition, in order to make changes after releasing the work item, you must first refresh the record. Otherwise, changes you make to the record may be lost.

Menu Commands for Displaying Error Messages

Agents can choose the following menu commands in order to redisplay messages, such as error messages, that had previously appeared in the area to the right of the application-level menus. These commands may be helpful for troubleshooting purposes. These commands are available through the communications configurations provided by Siebel Systems.

To display the previous message

- Choose Tools > Communications > Toolbar > Previous Message.

  The previous message appears, from the stack of messages that had appeared since the time the agent logged on.

To display the next message

- Choose Tools > Communications > Toolbar > Next Message.

  The next message appears, assuming the agent previously invoked the Previous Message command.

Menu Command for Refreshing the Communications Toolbar

Agents can choose the following menu command in order to refresh the status of buttons on the communications toolbar. (The message area to the right of the application-level menus is not refreshed.)

To refresh the communications toolbar

- Choose Tools > Communications > Toolbar > Refresh.

  The communications toolbar is refreshed to the current status.

Creating Communications Profiles for Personal Use

This section describes creating communications profiles for personal use. Each user can create profiles for personal use in the My Profiles view in the Communications screen. Such profiles will be available to this user only.

NOTE: By default, end-user responsibilities such as Email Response Agent and Universal Agent do *not* have visibility to the My Profiles view (CommSrv CM Profile Personal View). Administrators can add this view to any responsibility, as required.

Profiles created by end users are generally applicable only to the Send Email and Send Fax commands.

For more information about using these commands, see "Sending Email, Fax, and Page Messages" on page 324.

Communications administrators, who create communications profiles for various purposes, can provide guidance to end users about creating personal profiles, addressing issues such as:

- How to name personal profiles
- For which drivers to create profiles
- Specify the profile parameters, and the override values
- How to use personal profiles

For more information about communications drivers and profiles, see "About Communications Drivers and Profiles" on page 35 and "Configuring Communications Drivers and Profiles" on page 44.

For more information about communications drivers provided by Siebel Systems and about driver parameters for which you can provide override values in profiles, see Chapter 13, "Using Email, Fax, and Other Systems."

To create a communications profile for personal use

1. Navigate to Communications > My Profiles.

   The Profiles list displays the profiles that the user has created.

2. In the Profiles list, add a new record.

3. Enter the name of the profile.

   The name can be any unique name among existing profiles—including profiles for which you do not have visibility. Typically, you specify your email address as the name—which you also specify as a parameter override value for the From Address parameter.

4 Specify a communications driver on which to base this profile. For profiles for email or fax, select Internet SMTP/POP3 Server.

5 Specify parameter override values for the profile:

   a In the Profile Parameter Overrides list, add a new record.

   b For the Name field, specify a driver parameter name for which you provide an override value.

      For example, you want to add the From Address parameter. You may also want to provide a value for the Reply-To Address parameter. In some cases, your administrator may instruct you to add the SMTP Server parameter.

   c For the Value field, specify the value for the parameter. For example:

      - For From Address, you want to specify your own email address, using the appropriate form.
      - For Reply-To Address, specify a reply address, if it differs from that specified using From Address.
      - For SMTP Server, specify the name of the SMTP server that is to process your outbound email communications.

Sending Email, Fax, and Page Messages

This section describes using the Send Email, Send Fax, and Send Page commands.

When you use the Send commands, you can specify communications templates to provide content, into which Siebel field data may be substituted. Alternatively, you can enter text or (for email or fax) attachments directly. The templates may have been created for you, or you can create your own templates.

When you use the Send Email (with the native Siebel email client) or Send Fax commands, you can specify a profile to indicate who is sending the message. The profile may have been created for your individual use or for group use, and typically includes your own email address or another email address as the sender. Alternatively, you (end user) may be allowed to create your own profiles. Among the profiles available to you, you can specify a default profile as a user preference.

NOTE: When you use the Send Email command, information about the email, such as recipients for To/CC/Bcc fields, is stored as an activity in the extension table S_EVT_MAIL. The base table for this extension table is S_EVT_ACT.

Communications sent from the Siebel Web Client are processed immediately. Communications for email or fax channels sent from the Siebel Mobile Web Client, while you are connected to the local database and disconnected from the enterprise database, are saved until you synchronize. They are then processed for delivery by the Communications Outbound Manager server component.

For information about creating your own templates, see Chapter 9, "Configuring Communications Templates."

For information about creating your own profiles for email or fax, see "Creating Communications Profiles for Personal Use" on page 323. For information about specifying communications preferences for the Send commands, such as a preferred default profile, see "Setting Communications User Preferences" on page 305.

Settings in the Spelling tab of the User Preferences screen generally apply to all of the Send commands. However:

- Always Check Spelling Before Sending applies to Siebel Email Response only.
- Ignore HTML Tags applies to Siebel Email Response, Send Email, and Send Fax only.

For more information about the spelling user preferences, see *Applications Administration Guide*.

Sending Email Using the Native Siebel Email Client

This section describes using the Send Email command when your default email client is the native Siebel email client. You can send email, optionally including Siebel application content, to any recipient. Email addresses can be retrieved from the Siebel Database, such as for email to employees or contacts.

Editing and formatting controls for Send Email message text are described in "Editing and Formatting Controls for Send Email and Send Fax" on page 332.

Whether the Pick Recipient dialog box appears after Step 2 in the following procedure depends on what kind of data currently has the focus when you choose the Send Email command:

- If the current record is a person, such as a contact or employee, then the Pick Recipient dialog box does not appear. The Send Email window appears.
- If the current record is an element such as a service request or account, then the Pick Recipient dialog box appears if generic recipients have been configured. Choose generic recipients such as the contacts associated with the current service request.

  NOTE: Users should generally select a single record before invoking Send Email when using templates for which field substitution will be performed. If multiple records are selected, generic recipients chosen will be drawn from all selected records. However, field substitution applies only to the first selected record, and a single email message is sent to all recipients.

- If the current record is a Siebel attachment or literature item, then the Pick Recipient dialog box does not appear. The Send Email window appears, and the item is added as an attachment to the pending email message. (Siebel attachments of type URL are not added as email attachments.)

To send email with the native Siebel email client

1. Optionally, select one or more records of contacts, employees, or other persons for whom email addresses are defined in the Siebel Database. Alternatively, select one or more records of attachments or literature items to send as attachments to the email message.
2. Do *one* of the following:
   - From the application-level menu, choose File > Send Email.
   - Press F9.

- Click to the right of Initiate Work Item on the communications toolbar, then click Send Email, or click Initiate Work Item when the displayed ToolTip text is "Send Email." See also "Initiating Work Items" on page 318.

The Pick Recipient dialog box appears, if you did not specify recipients in Step 1. If you chose recipients before invoking the command, the Send Email window appears; go to Step 4.

3. In the Pick Recipient dialog box, specify the recipient or specify no recipient yet.

   The available recipients to choose from depend on the application context. For example, if Send Email was invoked from a Service Requests list, the choices are Service Request Contact and Service Request Owner.

   The Send Email window appears.

4. For the From field, specify the profile to represent who is sending the message.

   The profiles listed are those created for communications drivers that support email, such as Internet SMTP/POP3 Server. In some cases, a profile may be inserted automatically. You can specify a default profile in the User Preferences screen, Outbound Communications options.

5. For the To, Cc, or Bcc fields, do one or more of the following:

   - Verify any recipients that were inserted automatically in previous steps.
   - Type any additional recipient email addresses. Note that email addresses you enter are not automatically validated. You must verify that the email addresses you enter are valid.
   - Use the address book to enter additional recipients. Click the To, Cc, or Bcc button and explicitly specify individual persons from the address book dialog box. For each intended recipient, check the To, CC, or BCC check box. After specifying all recipients, click OK. Verify that the addresses appear correctly.

6. Optionally, for the Body drop-down list, choose the name of a communications template (of type Body) to insert into the message body.

   A template may be configured to be inserted automatically, according to where you invoked the Send Email command. (Administrators: for more information, see "Configuring Default Templates for Send Email Command" on page 179.)

   Field substitution applies to template text upon insertion. For substitution to work correctly, the fields must exist in the current list or form (containing the focus) in the Siebel application. Field substitution applies to a single selected record only.

   You can insert more than one template. Any template you choose is appended to the existing text.

   Templates listed here are subject to filtering based on several factors, including channel, language, locale, and whether the template is HTML or plain text. For details, see "Template Visibility and Access" on page 270.

7. Click the Change Language/Locale button to change the language or locale, as necessary.

   Setting the language and locale changes the list of available templates to those associated with the language and locale you specify.

8 Optionally, verify or enter text for the subject line.

Text may have been inserted into the subject line automatically when you chose a template.

9 Optionally, enter and format free-form text, or modify or add to the template text you inserted in Step 6.

For information about the available text-editing controls, see "Editing and Formatting Controls for Send Email and Send Fax" on page 332.

10 Optionally, if you are editing in plain text mode, you can click the Remove HTML Tags button, at upper right, to remove any HTML tagging that may be present in your message text.

11 Optionally, click the icons on the lower right to specify the operating system files (paper clip icon) or literature items ("document" icon) as attachments to the email message.

The attachments icon lets you specify files from the operating system as attachments, and displays a list of all files and literature items previously specified for the email message.

The literature icon displays a list of literature items you can specify as attachments.

The Attachments field displays the most recently specified attachment, if more than one attachment has been specified. All email attachments are saved as attachments to the corresponding Siebel activity record, which is updated when the message is sent.

12 Optionally, check spelling for your message.

13 Click Send, or click Cancel if you decide not to send the email message.

NOTE: If you cancel the email, the corresponding Siebel activity record (created to track the send operation) is deleted.

Sending Email Using Lotus Notes or Microsoft Outlook

This section describes using the Send Email command when your default email client is set to Lotus Notes or Microsoft Outlook. You can send email, optionally including Siebel application content, to any recipient. Email addresses can be retrieved from the Siebel Database, such as for email to employees or contacts.

For additional information on using Lotus Notes or Microsoft Outlook, refer to your documentation for these third-party products. For more information about setting user preferences for using third-party email clients, see "Preference Settings for Outbound Communications" on page 306.

Whether the Recipient field appears in the Recipient/Template dialog box after Step 2 on page 328 in the procedure below depends on what kind of data currently has the focus when you choose the Send Email command:

- If the current record is a person, such as a contact or employee, then the Recipient field does not appear.

- If the current record is an element such as a service request or account, then the Recipient field appears if generic recipients have been configured. Choose generic recipients such as the contacts associated with the service request.

  NOTE: Users should generally select a single record before invoking Send Email when using templates for which field substitution will be performed. If multiple records are selected, generic recipients chosen will be drawn from all selected records. However, field substitution applies only to the first selected record, and a single email message is sent to all recipients.

- If the current record is a Siebel attachment or literature item, then the Recipient field appears if generic recipient have been configured. The item will be added as an attachment to the pending email message. (Siebel attachments of type URL are not added as email attachments.)

To send email with Lotus Notes or Microsoft Outlook

1. Optionally, select one or more records of contacts, employees, or other persons for whom email addresses are defined in the Siebel Database. Alternatively, select one or more records of attachments or literature items to send as attachments to the email message.
2. Do *one* of the following:
   - From the application-level menu, choose File > Send Email.
   - Press F9.
   - Click to the right of Initiate Work Item on the communications toolbar, then click Send Email, or click Initiate Work Item when the displayed ToolTip text is "Send Email." See also "Initiating Work Items" on page 318.
3. In the Recipient/Template dialog box:
   a. From the Recipient field, specify the recipient, or specify no recipient yet.

      If you chose recipients before invoking the command, the Recipient field does not display.

      The available recipients to choose from depend on the application context. For example, if Send Email was invoked from a Service Requests list, the choices are Service Request Contact and Service Request Owner.

   b. From the Message Template field, choose the name of a communications template (of type Body) to insert into the message body.

      Field substitution applies to template text upon insertion. For substitution to work correctly, the fields must exist in the current list or form (containing the focus) in the Siebel application. Field substitution applies to a single selected record only.

      Templates listed here are subject to filtering based on several factors, including channel, language, locale, and whether the template is HTML or plain text. For details, see "Template Visibility and Access" on page 270.

      NOTE: If the current record type (business component) has a default template associated with it, then you do not need to specify a template. If you do specify a template, its text is appended to the default template text in the email message. Consult your administrator or verify the software behavior to see if a default template has been configured.

c From the Attachments field, specify literature items from the Siebel Database to add as attachments, as necessary.

The literature icon ("paper clip" icon) displays a list of Siebel literature items you can specify as attachments.

The Attachments field displays the most recently specified attachment, if more than one attachment has been specified. You can add file attachments later, from the Notes or Outlook window. When the message is sent, all email attachments are saved as attachments to the corresponding Siebel activity record.

d Click the Change Language/Locale button to change the language or locale, as necessary.

Setting the language and locale changes the list of available templates to those associated with the language and locale you specify.

e In the Recipient/Template dialog box, click OK to continue to the email message window for Lotus Notes or Microsoft Outlook.

4 In the Lotus Notes or Microsoft Outlook email message window:

a For the To, Cc, or Bcc fields, do one or more of the following:

- Verify any recipients that were inserted automatically in previous steps.
- Specify any additional recipient email addresses from the email directories available through your email client program.

b Optionally, verify or enter text for the subject line.

Text may have been inserted into the subject line automatically when you chose a template.

c Optionally, enter and format free-form text, or modify or add to the template text you inserted previously.

d Optionally, specify operating system files as attachments to the email message.

Siebel literature items you previously specified are also saved as attachments for the email message.

All email attachments are saved as attachments to the corresponding Siebel activity record, which is updated when the message is sent.

e Optionally, check spelling for your message, according to what your email client program supports.

f Send the message, or cancel if you decide not to send the email message.

NOTE: If you cancel the email, the corresponding Siebel activity record (created to track the send operation) is deleted.

Sending Faxes

You can send a fax, optionally including Siebel application content, to any recipient. Fax addresses can be retrieved from the Siebel Database, such as for faxes to employees or contacts.

Whether the Pick Recipient dialog box appears after Step 2 in the following procedure depends on what kind of data currently has the focus when you choose the Send Fax command:

- If the current record is a person, such as a contact or employee, then the Pick Recipient dialog box does not appear. The Send Fax window appears.

  NOTE: In order for the fax activity record to be associated with a recipient derived from the current record (such as a contact or employee), the fax addressing information in the To, Cc, or Bcc line must not be changed. In addition, the fax addressing field must have been configured in Siebel Tools for your deployment.

- If the current record is an element such as a service request or account, then the Pick Recipient dialog box appears if generic recipients have been configured. Choose generic recipients such as the contacts associated with the current service request.
- If the current record is a Siebel attachment or literature item, then the Pick Recipient dialog box does not appear. The Send Fax window appears, and the item is added as an attachment to the pending fax message. (Siebel attachments of type URL are not added as email attachments.)

To send a fax

1. Optionally, select one or more records of contacts, employees, or other persons for whom fax numbers are defined in the Siebel Database. Alternatively, select one or more records of attachments or literature items to send as attachments to the fax message.
2. Do *one* of the following:

   - From the application-level menu, choose File > Send Fax.
   - Press Ctrl+F9.
   - Click to the right of Initiate Work Item on the communications toolbar, and then click Send Fax, or click Initiate Work Item when the displayed ToolTip text is "Send Fax." See also "Initiating Work Items" on page 318.

   The Pick Recipient dialog box appears, if you did not specify recipients in Step 1. If you chose recipients before invoking the command, the Send Fax window appears; go to Step 4 on page 330.

3. In the Pick Recipient dialog box, specify the recipient or specify no recipient yet.

   The available recipients to choose from depend on the application context. For example, if Send Fax was invoked from a Service Requests list, the choices are Service Request Contact and Service Request Owner.

   The Send Fax window appears.

4. For the From field, specify the profile to represent who is sending the message.

   The profiles listed are those created for communications drivers that support email or fax, such as Internet SMTP/POP3 Server. In some cases, a profile may be inserted automatically. You can specify a default profile in the User Preferences screen, Outbound Communications options.

5. For the To, Cc, or Bcc fields, do one or more of the following:

   - Verify any recipients that were inserted automatically in previous steps.

- Type any additional recipient fax addresses, using the appropriate format for your fax integration.
- Use the address book to enter additional recipients. Click the To, Cc, or Bcc button and explicitly specify individual persons from the address book dialog box. For each intended recipient, check the To, CC, or BCC check box. After specifying all recipients, click OK. Verify that the addresses appear correctly.

6 Optionally, for the Body drop-down list, choose the name of a communications template (of type Body) to insert into the message body.

A template may be configured to be inserted automatically, according to where you invoked the Send Fax command. (Administrators: for more information, see "Configuring Default Templates for Send Email Command" on page 179.)

Field substitution applies to template text upon insertion. For substitution to work correctly, the fields must exist in the current list or form (containing the focus) in the Siebel application.

You can insert more than one template. Any template you choose is appended to the existing text.

Templates listed here are subject to filtering based on several factors, including channel, language, locale, and whether the template is HTML or plain text. For details, see "Template Visibility and Access" on page 270.

7 Optionally, verify or enter text for the subject line.

Text may have been inserted into the subject line automatically when you chose a template.

8 Optionally, enter and format free-form text, or modify or add to the template text you inserted in Step 6.

For information about the available text-editing controls, see "Editing and Formatting Controls for Send Email and Send Fax" on page 332.

9 Optionally, if you are editing in plain text mode, you can click the Remove HTML Tags button, at upper right, to remove any HTML tagging that may be present in your message text.

10 Optionally, click the icons on the lower right to specify operating system files ("paper clip" icon) or literature items ("document" icon) as attachments to the fax message.

The attachments icon lets you specify files from the operating system as attachments, and displays a list of all files and literature items previously specified for the fax message.

The literature icon displays a list of literature items you can specify as attachments.

The Attachments field displays the most recently specified attachment, if more than one attachment has been specified. All fax attachments are saved as attachments to the activity record that is created when the message is sent.

11 Optionally, check spelling for your message.

12 Click Send, or click Cancel if you decide not to send the fax.

NOTE: If you cancel the fax, the corresponding Siebel activity record (created to track the send operation) is deleted.

Editing and Formatting Controls for Send Email and Send Fax

For the Send Email command (native Siebel email client only) and for the Send Fax command, you can edit and format your message text before sending it, using controls in an editing bar that appears when you click in the message area in the Send Email or Send Fax window.

The editing bar appears only when HTML is the setting for the Send Email: Default Message Format option in the Outbound Communications options of the User Preferences screen. For more information about this user preference setting, see "Preference Settings for Outbound Communications" on page 306.

NOTE: If you are using Lotus Notes or Microsoft Outlook for the Send Email command, the editing and formatting capabilities are those provided by the third-party vendor. The Siebel software attempts to specify an editing environment according to the user's Send Email Default Message Preference setting. However, the behavior of the third-party email client software cannot be guaranteed. Users can also specify editing preferences in Lotus Notes or Microsoft Outlook.

For email messages, message recipients who cannot view HTML messages in their email client program will instead view a plain-text version of each message, from which any formatting has been stripped out.

Managing Line Breaks for HTML Messages

When HTML editing is enabled, note the following behavior regarding line breaks:

- Pressing ENTER creates a new paragraph.
- Pressing SHIFT+ENTER creates a new line.

Managing Links and HTML Elements from Other Sources

For email messages, when HTML editing is enabled, you can enter URL or mailto links directly and they will be automatically converted to links in the HTML output. For example:

- www.siebel.com
- http://www.siebel.com
- ftp://ftp.topsecretclassified.gov
- mailto:user@siebel.com

Graphics or other elements displayed on Web pages, such as horizontal rules or tables, may be selectable by dragging the mouse pointer. In some cases, such elements can be copied and pasted into the Send Email/Fax message body.

If you are using plain-text mode in the Send Email or Send Fax window, you can remove any remnant HTML formatting by clicking the button Remove HTML Tags.

NOTE: Content other than text or HTML tag elements cannot be added directly to the message text. For example, you cannot copy and paste graphics or other files into the message text. You cannot drag and drop text but you can cut and paste text.

Editing and Formatting Options

The HTML formatting options include, from left to right:

- **Find/Replace.** Click the arrow to display the Find controls above the editing bar, or to hide these controls. When the Find controls appear above the editing bar, click the arrow on the left to toggle between the Find controls and the Find and Replace controls. Enter text to find, or enter replacement text, then click Go. Find operations are not case-sensitive.
- **Cut.** Click to cut selected text to the clipboard.
- **Copy.** Click to copy selected text to the clipboard.
- **Paste.** Click to paste text from the clipboard into the message area. Depending on the source, text you paste may include HTML formatting.
- **Font.** Choose a font from a drop-down list to apply to selected text. Available fonts include Arial (the default), Verdana, Times New Roman, and Courier.
- **Size.** Choose a size from a drop-down list to apply to selected text. Point sizes include 8 (the default), 10, 12, 14, 18, 24, 36.
- **Font Color.** Click the arrow to display font colors above the editing bar, then click to choose a color to apply to selected text.
- **Bold.** Click to apply bold formatting to selected text, or to remove bold.
- **Italic.** Click to apply italic formatting to selected text, or to remove italics.
- **Underline.** Click to apply underlining to selected text, or to remove underlining.
- **Ordered List.** Click to apply numbering to selected text (make it an ordered list), or to remove numbering.
- **Unordered List.** Click to apply bullets to selected text (make it an unordered list), or to remove bullets. Bullets appear differently at different levels of indenting.
- **Indent.** Click to increase indenting for selected paragraphs.
- **Outdent.** Click to decrease indenting for selected paragraphs (outdent).
- **Left Align.** Click to left-align selected paragraphs.
- **Center Align.** Click to center selected paragraphs.
- **Right Align.** Click to right-align selected paragraphs.

Sending Pages

You can send a page, optionally including Siebel application content, to any recipient whose paging data is stored in the Siebel Database.

NOTE: The Send Page command requires the Page Manager server component. For more information about setting up and using Page Manager, see *Siebel Business Process Designer Administration Guide*.

To send a page

1. Optionally, select a record of a contact, an employee, or another person for whom paging data is defined in the Siebel Database.
2. Do *one* of the following:
   - From the application-level menu, choose File > Send Page.
   - Press Shift+F9.
   - Click to the right of Initiate Work Item on the communications toolbar, then click Send Page, or click Initiate Work Item when "Send Page" is the displayed ToolTip text. See also "Initiating Work Items" on page 318.
3. For the To field, do one or more of the following:
   - Verify any recipient that was inserted automatically in previous steps.
   - Click the select button to the right of the To field and explicitly specify an individual person from the Siebel Database.
4. Specify page addressing information, including the pager company, paging phone number, and pager PIN, as appropriate.
5. Optionally, for the Body drop-down list, choose the name of a communications template (of type Body) to insert into the message body.

   Field substitution applies to template text upon insertion. For substitution to work correctly, the fields must exist in the current list or form (containing the focus) in the Siebel application.

   You can insert more than one template. Any template you choose is appended to the existing text.

   Templates listed here are subject to filtering based on channel and other factors. For details, see "Template Visibility and Access" on page 270.
6. Optionally, enter free-form text, such as to modify or add to the template text, as appropriate.
7. Optionally, check spelling for your message.
8. Click Send.

Creating Activities for Send Commands

Activity records may be generated for all of the Send commands that you use to send outbound communications: Send Email, Send Fax, and Send Page.

For more information about using the Send commands, see "Sending Email, Fax, and Page Messages" on page 324.

In the User Preferences screen, Outbound Communications options, you can specify whether to generate activity records for the Send commands, and, if so, whether the activity records will be public or private. For details, see the setting Upon Sending Messages Generate, in "Preference Settings for Outbound Communications" on page 306.

Each activity record documents a communication initiated within the Siebel application. Generally, this activity is also a child record of another record which was current when the command was invoked. For example, if the Send Email command is invoked when a service request record was active, then the activity record for the email message is a child of the service request record, and can be accessed from the service request. You can fill in other fields to complete the activity record, as may be appropriate for your company's use of activities.

NOTE: For an activity generated by using Send Email, the Status field is set to Done when the message is sent. If the message is sent to an invalid address, the sender is notified by email that the message could not be delivered. Note, however, that the Status field is not updated in this case. The email system does not update the activity record that a message could not be delivered.

The owner for an activity generated by a Send command is the sender of the communication. Recipients are associated with the activity as contacts, if the Siebel application can match the recipients with persons in the Siebel Database. For details, see "Contact Matching for Send Email Activity Records" on page 336.

Activity Types for Each Send Command

The activity records generated by each Send command are of different types, as outlined in Table 65 on page 335.

Activity records for outbound communication requests also use the same activity types described in this section, but are not generated in the same way. For more information about these activity records, see "Viewing Activity Records for Communication Requests" on page 302.

For Siebel Email Response, activity records for inbound email messages use the Email - Inbound activity type, while activity records for outbound email messages use the Email - Outbound activity type. For more information, see *Siebel Email Response Administration Guide*. See also the subsection about activity records and attachments, in "Administering Communications Inbound Receiver" on page 256.

Table 65. Activity Records for Outbound Communications

Send Command	Activity Type
Send Email	Email - Outbound
Send Fax	Fax - Outbound
Send Page	Paging

Text from the Subject line of the message is written to the Description field for the activity record. (For pages, this text is from the Subject field of the template.) Message body text is written to the Comments field for the activity record, up to a maximum possible size of 1,500 characters.

For Send Email and Send Fax, message body text is also written to the Email Body field, up to a size of at least 16,000 characters (actual length depends on your database).

For Send Email and Send Fax, message body content that is longer than the Email Body field allows is also saved as activity attachments. The attachment files are named SiebelLongEmailBody.txt (for plain-text messages) or SiebelLongEmailBody.htm (for HTML messages). If you are using Microsoft Outlook as your email client, RTF messages are saved in HTML format.

For the Send Email and Send Fax commands, attachments (such as files or literature items) to email or fax messages are also saved as attachments to the activity record.

Contact Matching for Send Email Activity Records

As noted, recipients are associated with an activity as contacts, if the Siebel application can match the recipients with persons in the Siebel Database. For Send Email, this matching is based primarily on the email address. If more than one record exists in the database that includes the same email address, the first such record is associated with the activity as a contact.

For a third-party email integration with Lotus Notes, an email address specified through Notes is passed back to the Siebel application in order to match recipients to contacts and associate these contacts with the activity. If no match is found in the Siebel Database, then the recipients are not associated as contacts.

For a third-party email integration with Microsoft Outlook, an email address may not always be passed back to the Siebel application, in order to match recipients to contacts and associate these contacts with the activity. If the email address is *not* passed, then the Siebel application tries to match recipients to contacts using the first and last names of the recipient, or using the first, middle, and last names. If no match is found in the Siebel Database, then the recipients are not associated as contacts.

NOTE: If the option Collaborative Data Object (CDO) is installed on each client machine running Microsoft Outlook 2000 or XP, then the email address is passed to the Siebel application, not just the name, and a match can generally be made.

12 Using Siebel CTI Connect

This chapter provides information about implementing Siebel CTI Connect to support handling voice communications work items using Intel NetMerge (formerly Dialogic) CTI middleware. It includes the following topics:

About Siebel CTI Connect

This section describes communications driver parameters, commands, and events for configuring Siebel Communications Server to work with Siebel CTI Connect, which is based on Intel NetMerge, and the applicable communications driver.

Siebel CTI Connect, available from Siebel Systems, includes the following:

- Intel NetMerge Call Processing Software (formerly *Dialogic CT Connect Server*)
- Intel NetMerge Call Information Manager (formerly *Dialogic Call Information Manager*)
- Communications driver: Siebel CTI Connect (formerly *Dialogic CTI*)
- Sample communications configuration data for Siebel CTI Connect

NOTE: Siebel CTI Connect is licensed as a separate module in this version of Siebel Business Applications. If you have not licensed the Siebel CTI Connect module, it will not be enabled for use.

The sample Siebel CTI Connect configuration data can be found in the Siebel Database or the Sample Database. For more information on particular parameters, commands, or events, refer to Intel NetMerge documentation. For more information, see "Additional Documentation for Siebel CTI Connect" on page 338.

For more information about configuring Siebel Communications Server, see Chapter 4, "Configuring Session Communications," and Chapter 5, "Configuring Events and Commands," and other parts of this book.

NOTE: For additional or different parameters, commands, or events, you can extend the Siebel-provided communications driver or develop a custom driver to support these parameters. For more information, see Appendix A, "Developing a Communications Driver."

Additional Documentation for Siebel CTI Connect

In addition to the *Siebel Communications Server Administration Guide* and other *Siebel Bookshelf* documentation, Siebel CTI Connect includes Intel NetMerge documentation.

These documentation files, provided as PDF files, are located in the *Siebel Business Applications Third-Party Bookshelf* CD-ROM. They are also available with the Intel NetMerge CPS and CIM installation.

NOTE: For information about hardware and software requirements for using Intel NetMerge, refer to the Intel NetMerge documentation.

Installing Siebel CTI Connect Server Components

For information about installation and configuration of Siebel CTI Connect server components, including Intel NetMerge Call Processing Software (CPS) and Intel NetMerge Call Information Manager (CIM), refer to documentation provided for those products.

Using Windows Explorer, navigate from the Siebel network image directory to the following folders to access the documentation:

- For CPS:

  `Windows\Server_Ancillary\Intel_Netmerge_Server\enu\cps\Docs`

- For CIM:

  `Windows\Server_Ancillary\Intel_Netmerge_Server\enu\cim\CIM3.5SP1`

You install from the Siebel network image. Using Windows Explorer, navigate from the Siebel network image directory to the following folders to locate the installation files:

- `Windows\Server_Ancillary\Intel_Netmerge_Server\enu\cps` (for CPS)
- `Windows\Server_Ancillary\Intel_Netmerge_Server\enu\cim` (for CIM)

  NOTE: You also need to install the service pack that is located in the above directory for CIM.

For more information about creating Siebel network images, see *Siebel Installation Guide* for the operating system you are using.

Limitations with Nortel Meridian Switch

The following limitations apply to call scenarios when you are using Siebel CTI Connect software with a Nortel Meridian switch:

- When an agent initiates a consultative or conference transfer to another agent, the communications toolbar of the originating agent indicates that the call is on hold. If this agent needs to retrieve the call from hold, it is recommended that the agent click the Release Work Item button to release the active call and then click the Resume Work Item button to retrieve the call from hold.

- Call forwarding is not supported from an agent's ACD/Position DN. The agent must log out of the ACD before call forwarding can be enabled.
- When a customer disconnects during a conference transfer, the communications toolbar of the originating agent does not indicate that the call has been disconnected.

Siebel CTI Connect Driver Settings

Table 66 on page 339 lists the settings for the driver record for the Siebel CTI Connect driver. These settings appear in the Communications Drivers and Profiles view in the Administration - Communications screen.

CAUTION: The values shown in Table 66 on page 339 are presented for reference only. Do not modify the predefined values for this driver.

Table 66. Siebel CTI Connect Driver Settings

Field Name	Value
Name	Siebel CTI Connect
Channel Type	Voice
Inbound	Selected by default
Outbound	Selected by default
Interactive	Selected by default
Channel String	CTC Phone
Library Name	sscmctc

Siebel CTI Connect Driver Parameters

Table 67 lists the supported driver parameters specific to the Siebel CTI Connect communications driver, for Intel NetMerge CTI middleware. This table indicates required parameters and displays applicable default values.

You view and modify these driver parameters in the Communications Drivers and Profiles view, which is one of the views in the Administration - Communications screen.

Each parameter may be prefaced with a keyword indicating how it will be used internally:

- Parameters prefaced with "Driver:" are sent to the driver handle when the driver is being initialized. These parameters are sent to the CreateISCDriverInstance method.
- Parameters prefaced with "Service:" are sent to the driver handle when it requests a service (creates the service handle). These parameters are sent to the RequestService method.
- Parameters without a prefacing keyword are sent to both CreateISCDriverInstance and RequestService.

For more information about the driver handle methods, see Appendix A, "Developing a Communications Driver."

Table 67. Siebel CTI Connect Driver Parameters

Parameter Name	Required	Default Value	Description
Driver: CIMServer	Yes	CHANGE_ME	Host name of the machine on which the CIM Server is running.
Driver: CIMTimeDelay	No	100	Specifies the time, in milliseconds, after which call data is fetched from the CIM Server. Increase this value if you notice call data is missing, such as when a call is transferred.
Driver: CTCServer	Yes	CHANGE_ME	Host name of the machine on which the Call Processing Server is running.
Driver: DriverLogFile	No	ctc.log	If a filename is specified using this parameter, Siebel Communications Server will write detailed information about activity for the active call into a log file. Data for Siebel Communications Server and Intel NetMerge activity is logged, including device events and event data fields. Log files are created by default in the log subdirectory of the Siebel Server installation directory, depending on the client type and on your communications deployment. A complete path can be specified along with the filename in order to write the file to a specific location. Optionally, if you want to use a single log file in a multichannel environment, you can specify the same log file name for all interactive drivers you are using. For debugging purposes, the parameter Driver:LogDebug can be set to TRUE, to cause debug messages to be included in the file. Also, the Communications Server log file can be compared with the log file for Intel NetMerge. In each log file, log records referring to the same call contain the same value for the refId event data field. See also the description for the parameter Service:ServiceLogFile.

Table 67. Siebel CTI Connect Driver Parameters

Parameter Name	Required	Default Value	Description
Driver: LogicalID	Yes	CHANGE_ME	Logical ID of the link for the switch as configured in the Call Processing Server. Example: CTCLINK.
Driver: MaxServices	No	25	Specifies the maximum number of service objects that the driver handle (ISC_DRIVER_HANDLE) can service. If the specified limit is reached, another instance of the driver handle is created. For more information about the driver handle, see Appendix A, "Developing a Communications Driver."
Driver: NetCallPort	No	9400	For any users running the Communications Simulator, this parameter specifies a number representing a simulated port number, so that users can simulate an "inside call" to another teleset.
Driver: NetworkType	Yes	ncacn_ip_tcp	Network communications protocol as configured in the Call Processing Server. Possible values include: ncacn_nb_nb – NetBIOS over NetBEUI ncacn_ip_tcp – TCP/IP ncacn_dnet_nsp – DECnet ncacn_nb_tcp – NetBIOS over TCP/IP ncacn_np – Named Pipes ncacn_spx – Novell SPX
Driver: Simulate	No	FALSE	TRUE or FALSE When this parameter is set to TRUE, this driver supports the Communications Simulator. This parameter need not be explicitly set in order to use the Communications Simulator. For more information, see "Enabling Session Communications and Simulation" on page 240 and "Simulating a Communications Environment" on page 211.

Table 67. Siebel CTI Connect Driver Parameters

Parameter Name	Required	Default Value	Description
Driver: SwitchType	Yes	CHANGE_ME	The type of switch for your call center. Values include: 0 – Avaya (Lucent) Definity G3 2 – Nortel Meridian 3 – Siemens Hicom 300E U.S., Siemens Hicom 300E International For information about the switches supported with Siebel CTI Connect, see *System Requirements and Supported Platforms* on Siebel SupportWeb.
Service: ACCDNList	No	{@ACDDNList}	Uses the macro @ACDDNList to obtain a list of ACD DNs (extensions of type A) associated with the current agent.
Service: AutoLogout	No	FALSE	Determines whether to automatically log the agent out of the ACD queue when the communications session has ended: ■ If FALSE, the agent is *not* automatically logged out of the ACD queue when the agent logs out of the Siebel session. ■ If TRUE, the agent is automatically logged out of the ACD queue when the agent logs out of the Siebel session. This setting applies whether the agent specifically exited the Siebel application or the browser, or if the agent's application session times out.
Service: DNList	Yes	{@DNList}	Uses the macro @DNList to obtain a list of DNs (standard extensions of type "S") associated with the current agent.
Service: HandleRouteRequest	No	FALSE	TRUE or FALSE Set to TRUE if you are using Route DNs for routing calls. Set to FALSE if you are using regular DNs.

Table 67. Siebel CTI Connect Driver Parameters

Parameter Name	Required	Default Value	Description
Service: Has*KeyName*	No	TRUE	TRUE or FALSE Settings that, when set to TRUE, specify whether the connected switch provides programmatic support for the indicated functionality. You can also disable a feature that is supported by the switch but that you do not want your agents to use. If the value is FALSE, the associated feature is disabled. For a disabled feature, an administrator can remove the associated toolbar button from the repository using Siebel Tools. The following capabilities can be enabled or disabled: ■ HasAgentBusy – Set agent state to Busy/ Not Busy ■ HasAnswer – Answer the incoming call ■ HasDisconnect – Disconnect the call ■ HasForward – Set call forwarding ■ HasHold – Put the call on hold For a list of the particular features your switch supports, refer to Intel NetMerge documentation.
Service: IsQueueRequired	No	FALSE	TRUE or FALSE Specifies whether an ACD queue must be specified in order for an agent to log in to a call center. Set to FALSE for an Avaya (Lucent) Definity G3 switch or for other switches that support Expert Agent Selection (EAS). Set to TRUE for all other switches. When this parameter is set to TRUE, the ACD queue number will be provided with the login command.

Table 67. Siebel CTI Connect Driver Parameters

Parameter Name	Required	Default Value	Description
Service: IsSiemens	No	FALSE	TRUE or FALSE When set to TRUE, specifies that you are using a Siemens switch, which requires special handling. If you are using the Siemens Hicom 300E switch, set this parameter to TRUE.
Service: SelectDN	No	CHANGE_ME	Specifies a DN to be used to issue commands such as MakeCall, and so on. Otherwise, the first DN in the list specified with the parameter Service:DNList is used.
Service: ServiceLogFile	No	ctc_ {@Username}. log	If a filename is specified using this parameter, Siebel Communications Server will write detailed information about activity for the active call, for each agent's session, into a log file. Data for Siebel Communications Server and Intel NetMerge activity is logged, including device events and event data fields. Log files are created by default in the log subdirectory of the Siebel Server installation directory, depending on the client type and on your communications deployment. For information about the @Username macro, see "Macros for Parameter Values" on page 182. For more information, see the description for the parameter Driver:DriverLogFile.

Table 67. Siebel CTI Connect Driver Parameters

Parameter Name	Required	Default Value	Description
Service: Use1StepTransfer	No	FALSE	TRUE or FALSE ■ Set to TRUE to specify that the switch you are using supports the OneStepTransfer function. The TransferMute driver command maps to this function. ■ Set to FALSE when OneStepTransfer is not supported. The Blind Transfer toolbar button is disabled. Set this parameter to FALSE for an Avaya (Lucent) Definity G3 switch or for other switches that do not support OneStepTransfer. The TransferMute command is used by the Blind Transfer toolbar button.
LogDebug	No	FALSE	TRUE or FALSE If TRUE, data output to the log file is more detailed. This parameter is generic and applies to log files specified using Driver:DriverLogFile and using Service:ServiceLogFile. If you specify this parameter prefixed with either Service: or Driver:, it will apply to the applicable log file only.

Table 67. Siebel CTI Connect Driver Parameters

Parameter Name	Required	Default Value	Description
MaxLogKB	No	128	Specifies the maximum size of the log file, in kilobytes (KB). This parameter is generic and applies to log files specified using Driver:DriverLogFile and using Service:ServiceLogFile. If you specify this parameter prefixed with either Service: or Driver:, it will apply to the applicable log file only. When the applicable log file is full, it is emptied completely, then logging begins again.
ReleaseLogHandle	No	TRUE	Indicates that the log file handle for each agent is released periodically, after logs have been generated. The default setting of TRUE provides better performance. When ReleaseLogHandle is FALSE, however, some log files may not be generated if the number of concurrent users exceeds the file-handle capacity provided by the operating system. This parameter is generic and applies to log files specified using Driver:DriverLogFile and using Service:ServiceLogFile. If you specify this parameter prefixed with either Service: or Driver:, it will apply to the applicable log file only.

Siebel CTI Connect Commands

Table 68 on page 347 lists and explains usage for commands specific to the Siebel CTI Connect communications driver, for Intel NetMerge CTI middleware.

A Yes value in the Any K/V (Key/Value) column in Table 68 on page 347 indicates that the command allows attachment of *any* user-defined key/value pair to the call. Within the communications configuration, these key/value pairs are represented by event or command parameters and the associated parameter values.

The commands you define, in which driver commands listed here are specified as device commands, support the command parameters documented under Chapter 5, "Configuring Events and Commands." In your command definitions, you can specify custom subparameters for command parameters of type Group.

An asterisk (*) before a parameter name means that the parameter is optional for this command. (The command parameters are described in Table 69 on page 352.)

Most of the commands in Table 68 on page 347 correspond to commands defined in the communications configuration, in which they are specified as the device command. They may also correspond to a communications toolbar button.

For more information, see Chapter 6, "Configuring User Interface Elements." For descriptions of communications toolbar elements, see Chapter 11, "Communications Operations for End Users."

For commands for which the TrackingID parameter is optional, the command operation is performed on the work item identified by the TrackingID parameter value. If this parameter is not provided, the command operates on the first available work item. K/V indicates Key/Value pairs.

Table 68. Siebel CTI Connect Commands

Command Name	Any K/V	Parameters	Description
AnswerCall	No	*TrackingID	Answer incoming call.
AttachData	No	Call data object	Attach user-defined data to the current call. This command can be invoked from a Siebel VB or Siebel eScript script. For example, a script in which "callobject" has previously been defined might include a line such as the following: CTI.InvokeCommand "AttachData," callobject
CancelForwardCall	No		Cancel call forwarding.
ChangeBusyState	No		Toggle agent state between Busy and Not Busy. Busy state is defined as setting the device state such that incoming calls do not ring at the device.
ChangeNotReady State	No	*ID *AgentId *AgentPin *ACDQueue	Toggle agent state between being available and not available to receive calls from the queue. This command may use either the ID parameter *or* the AgentId, AgentPin, and ACDQueue parameters. For more information, see "Siebel CTI Connect Command Parameters" on page 351.
Conference Complete	No		Complete the conference call. The caller and the agents in conference can now talk to each other at the same time.

Table 68. Siebel CTI Connect Commands

Command Name	Any K/V	Parameters	Description
ConferenceInit	Yes	PhoneNumber *CallNotifyText	Begin conference call. The caller is put on hold and the current agent dials another agent's extension. The CallNotifyText parameter conveys status information to the second agent.
ForwardCall	No	PhoneNumber	Set call forwarding.
HoldCall	No	*TrackingID	Put the current call on hold.
LogIn	No	*ID *AgentId *AgentPin *ACDQueue	Agent login on ACD queue. This command may use either the ID parameter *or* the AgentId, AgentPin, and ACDQueue parameters. The Login command is used for both automatic and manual login. For details, see "Siebel CTI Connect Command Parameters" on page 351, and see "Configuring Communications Log In and Log Out" on page 201.
LogOut	No		ACD agent logout.
MakeCall	No	PhoneNumber	Place the outbound call.
MergeCall	No		Combines an active call and a suspended (held) call into a conference call.
ReleaseCall	No	*TrackingID	Release (disconnect) current call.
ResetState	No		Reset internal information about current call states for the monitored teleset. This command resets call-status tracking information to the initial state when the agent started the Siebel application. ResetState resets all tracking information, regardless of the actual current status. For this reason, use it only when the teleset and Siebel Communications Server are out of sync and there are no current calls. **NOTE:** Although this command is part of the Siebel CTI Connect communications driver, it is specific to Siebel Communications Server and is not passed to Intel NetMerge Call Processing Server.
RetrieveCall	No		Retrieve the original call initiated using ConferenceInit or TransferInit, after the current call has been released using ReleaseCall.

Table 68. Siebel CTI Connect Commands

Command Name	Any K/V	Parameters	Description
RouteCall	No	routeId	Specifies a new destination for the incoming call, if the device is of type Route Point.
SelectDN1... SelectDN5	No		Select Directory Number (DN) #1 (2, 3, 4, or 5) for subsequent use. Using one of these commands selects the DN that will be used in commands such as MakeCall. It is recommended that end users select the preferred teleset and extension in the Communications options of the User Preferences screen to keep teleset information in sync.

Table 68. Siebel CTI Connect Commands

Command Name	Any K/V	Parameters	Description
SetAgent *WorkMode*	No	*ID *AgentId *AgentPin *ACDQueue	Set the agent work mode to one of five settings. The five work mode commands are: ■ SetAgentReady – Sets AgentReady work mode. After each call, the agent is available to take another call. ■ SetAgentNotReady – Sets AgentNotReady work mode. After each call, the agent is not available to take another call and must select the Ready State (for the voice channel) in the communications toolbar to become available. ■ SetAgentAfterCallWork – Sets AgentAfterCallWork work mode. This means that the agent has completed a call and requires wrap-up time before taking another call. The agent is not available to take calls and must select Ready State to become available. (Not supported by Nortel Meridian switches.) ■ SetAgentBusy – Sets AgentBusy work mode. After each call, the agent is available to take another call. (AgentBusy is not supported by Nortel Meridian, Avaya (Lucent) Definity G3, or CSTA Phase I switches.) ■ SetAgentOtherWork – Sets AgentOtherWork work mode. Typically, this means that the agent is not engaged in call center work. The agent is not available to take calls and must select Ready State to become available. (Not supported by CSTA Phase II switches.) The mode the agent is in upon logging in depends on the switch. If an agent is in AgentOtherWork or AgentNotReady mode upon logging in, for example, the agent must select the Ready State before taking any calls. An agent can choose work mode commands only after logging in. This command may use either the ID parameter *or* the AgentId, AgentPin, and ACDQueue parameters. For more information, see "Siebel CTI Connect Command Parameters" on page 351.

Table 68. Siebel CTI Connect Commands

Command Name	Any K/V	Parameters	Description
SimulateCall	Yes	*CallNotifyText	Simulate incoming call. Used for configuration debugging purposes. **NOTE:** Although this command is part of the Siebel CTI Connect communications driver, it is specific to Siebel Communications Server and is not passed to Intel NetMerge Call Processing Server.
ToggleForward	No	*PhoneNumber	Set call forwarding, or cancel call forwarding. When canceling call forwarding, the PhoneNumber parameter is not used.
TransferComplete	No		Complete the consultative transfer. The current agent is disconnected and the caller is connected to the agent to whom the call was transferred.
TransferInit	Yes	PhoneNumber *CallNotifyText	Begin consultative transfer. The caller is put on hold, and the current agent dials another agent's extension. The CallNotifyText parameter conveys status information to the second agent.
TransferMute	Yes	PhoneNumber *CallNotifyText	Initiate a mute (blind) transfer of the caller. The CallNotifyText parameter conveys status information to the second agent.
UnHoldCall	No	*TrackingID	Remove the current call from hold state.

Siebel CTI Connect Command Parameters

Table 69 on page 352 lists command parameters applicable to Siebel CTI Connect, and describes their usage. (Table 68 on page 347 identifies the commands that use each of these parameters.)

Within the communications configuration, command parameters and values to pass to the communications driver can be specified within command data definitions as subparameters of the Param parameter. For more information, see "Command Data" on page 142.

Table 69. Siebel CTI Connect Command Parameters

Command Parameter	Description
ACDQueue	ACD queue name. This parameter can be used with the LogIn, ChangeNotReadyState, and SetAgent*WorkMode* commands, which may be included in the communications configuration for Siebel CTI Connect. The same data can be retrieved from the Siebel Database using the macro @Queued. For details, see "Macros for Parameter Values" on page 182.
AgentId	An agent's login name. This parameter can be used with the LogIn, ChangeNotReadyState, and SetAgent*WorkMode* commands, which may be included in the communications configuration for Siebel CTI Connect. The same data can be retrieved from the Siebel Database using the macro @AgentId. For details, see "Macros for Parameter Values" on page 182.
AgentPin	The password (personal identification number) for an agent's login name. This parameter can be used with the LogIn, ChangeNotReadyState, and SetAgent*WorkMode* commands, which may be included in the communications configuration for Siebel CTI Connect. The same data can be retrieved from the Siebel Database using the macro @AgentPin. For details, see "Macros for Parameter Values" on page 182.
CallNotifyText	Text that will be displayed in the status line to the second agent in order to announce the incoming call (for internal call, call transfer, or call conference).

Table 69. Siebel CTI Connect Command Parameters

Command Parameter	Description
ID	This parameter can be used with the LogIn, ChangeNotReadyState, and SetAgent*WorkMode* commands, which may be included in the communications configuration for Siebel CTI Connect. The ID parameter can be used in place of the AgentID, AgentPin, and ACDQueue parameters. This parameter has the format *QQQQ*/*AAAA*-*PPPP*, where: *QQQQ* is the ACD queue *AAAA* is the agent login *PPPP* is the agent password If the LogIn command is defined to use the ID parameter, you should have the command read current agent data from the database, using macros such as @QueueId, @AgentId, and @AgentPin. For details, see "Macros for Parameter Values" on page 182. The LogIn command as defined in the sample communications configuration data uses the AgentId, AgentPin, and ACDQueue parameters instead of the ID parameter. Each of the *QQQQ*, */AAAA*, and *-PPPP* components of the string is optional—that is, depending on the switch and how you have configured the LogIn command, you may be able to skip some of them.
PhoneNumber	Phone number (dialing pattern). DialingFilter.Rule*N* configuration parameters must be defined that filter full telephone numbers from a database so they can be handled as extensions. For more information, see "Specifying Parameters for Communications Configurations" on page 52.
TrackingID	The designated tracking ID of a work item, to identify the specific work item on which an operation is to be performed.

Siebel CTI Connect Events

Table 70 on page 354 describes an event that may be received by the Siebel CTI Connect driver.

NOTE: See Intel NetMerge documentation for information about events and event data that is specific to Intel NetMerge.

Table 70. Siebel CTI Connect Event

Event Name	Description
EventAnswerCall	Indicates the arrival of a new call. This event is triggered whenever the AnswerCall device command is invoked by the Siebel CTI Connect driver, such as when an agent answers a call using the communications toolbar. This event may be used in your event handler definitions in order to enhance the speed of screen pops. For more information, see "Using Device Event to Enhance Screen Pop Performance" on page 207.

13 Using Email, Fax, and Other Systems

This chapter provides information about configuring and managing integrations with email servers, fax servers, and other communications systems. It includes the following topics:

Interfacing with Email and Fax Servers

This section describes the parameters for communications drivers provided by Siebel Systems that support email and fax. For this release, the Internet SMTP/POP3 Server driver is the only such driver.

NOTE: For information about upgrade issues that relate to communications drivers (adapters) used in release 6.*x*, see Appendix D, "Upgrading from Release 6.x."

The Internet SMTP/POP3 Server driver supports inbound email for Siebel Email Response, and supports outbound email and fax messages for many different Siebel application modules and features. Outbound email and fax messages can be sent using:

- Siebel Email Response (email only)
- Send Email and Send Fax commands
- Outbound communication requests views in the Administration - Communications screen (for administrators) and Communications screen (for end users)
- Workflow policies and workflow processes, created using Siebel Workflow
- Any module that invokes methods of the Outbound Communications Manager business service to send email or fax messages

For more information about these Siebel modules, refer to the appropriate section in this guide and to other documentation on *Siebel Bookshelf*.

For more information about specifying recipients for email or fax messages, see "Configuring Recipient Groups for Requests and Advanced Templates" on page 167 and "Configuring Recipients for Send Commands" on page 176.

For more information about communications drivers, see "About Communications Drivers and Profiles" on page 35 and "Configuring Communications Drivers and Profiles" on page 44.

Integrating with Email Systems

Sending outbound email messages and receiving inbound email messages using Siebel Communications Server requires that you integrate your email system. You must configure profiles for the Internet SMTP/POP3 Server driver, using the parameters described in Table 72 on page 361.

However, client-side email integration with Lotus Notes or Microsoft Outlook, options for the Send Email command, does not use Siebel Communications Server. For details, see "Configuring Client-Side Integration for Send Email" on page 383.

NOTE: After an email message is sent to the email server, the Communications Outbound Manager server component may log that the email has been successfully sent. However, the log file cannot indicate whether the email was sent to a valid email address. If the email server subsequently fails to deliver the message, a nondelivery report will be sent from the email server to your sending mailbox.

For information about supported third-party products, see *System Requirements and Supported Platforms* on Siebel SupportWeb. For information about configuring inbound communications, see *Siebel Email Response Administration Guide*.

Using HTML Email

HTML email messages are supported for both outbound and inbound email, and for all Siebel application features that employ Siebel Communications Server to process email.

For inbound email, the HTML email is rendered in the browser (running the Siebel client) according to the capabilities of the browser.

For outbound email, HTML tagging may be applied in various ways or by various features, including any combination of the following:

- Tagging may be applied by the Web browser when you use HTML editing controls with the Send Email command, with Siebel Email Response replies, or when editing template text.
- Tagging may be included in text or other elements you have copied from a browser and pasted into your email message or template text.
- Tagging may be applied manually to template items for advanced templates.
- Tagging may be specified in HTML wrapper templates.

User preferences determine whether users can access editing controls for directly formatting HTML outbound email. When HTML is specified, the tags applied by the HTML editing controls are subject to the capabilities of the supported browser. The HTML editing preference is specified separately for Send Email or for Siebel Email Response. Some tags are implicitly inserted, regardless of whether a user has used any specific HTML editing controls. Other tags are applied, or removed, by the browser, as the user formats the email text using these controls.

If HTML outbound email is generated, as outlined above, then an email message that is actually sent may be constructed as a MIME multipart/alternative email message, containing an HTML version and a plain-text version of the message. This behavior can be specified by setting the parameter Create Plain Text From HTML to TRUE for the Internet SMTP/POP3 Server communications driver or for applicable profiles.

The Ignore HTML Tags setting, in the Spelling tab of the User Preferences screen, will cause spell checking to ignore any text enclosed in angle brackets (< and >), even text that did not derive from HTML tags. If an open angle bracket (<) appears in the text, then the Check Spelling feature will ignore all subsequent message text, because it expects a matching closing angle bracket (>).

For more information about HTML formatting for Siebel Email Response, see *Siebel Email Response Administration Guide* and *Applications Administration Guide*.

For further information, see:

- "Template Content and Formatting" on page 268
- "Editing and Formatting Controls for Template Text" on page 275
- "Preference Settings for Outbound Communications" on page 306
- "Editing and Formatting Controls for Send Email and Send Fax" on page 332
- "Driver Parameters for Internet SMTP/POP3 Server Driver" on page 360
- "Activity Attachments Stored for Inbound Email" on page 259

Integrating with Fax Systems

Sending outbound fax messages requires that you integrate your email system with Siebel Communications Server and integrate your fax server with your email server. In addition, you must prepare your Siebel applications to format fax addressing data correctly.

Siebel software does not natively send any fax messages. Rather, the software sends an email message to an email address that is formatted to include the recipient fax number. The email-fax gateway server passes this email to the fax server. The fax server converts it to a fax and sends it to the recipient fax number. For information about supported third-party products, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

NOTE: After a fax is sent to the email server for submission to the fax server, the Communications Outbound Manager server component may log that the fax has been successfully sent. However, this log file message does not indicate whether the fax was sent to a valid fax number.

Most fax software sends a notification message when a fax cannot be transmitted to the destination. Some fax software applications can also be set up to send a notification when a fax is transmitted successfully. If the fax server has been configured to send notifications to the sender, the notification appears in the mailbox that corresponds to the profile that was used to send the fax.

Your fax software, such as Captaris RightFax, must be configured to communicate with the email server over a configured gateway connection (or communications link). For example, when installing RightFax, use the gateway for SMTP/POP3. For more information about email-fax gateways, refer to the documentation for your fax server.

You must also create an email account (mailbox) that will be solely monitored by the fax server. For example, email messages addressed to the RightFax email account are transmitted through the gateway connection and automatically picked up by the RightFax server, which converts them into faxes and sends them. Each RightFax email gateway for SMTP/POP3 can monitor a single such email account.

NOTE: If you are using Communications Server with Captaris RightFax, in order to support Unicode configurations download the Unicode patch for Captaris RightFax. For more information, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

Configuring Fax Integration Elements in Siebel Tools

Siebel Business Applications are preconfigured to send messages to be processed by Captaris RightFax. You must modify the configuration data in Siebel Tools in order to have fax recipient information processed correctly for the fax server you are using, as necessary, and to work with the email account you are using with the fax server.

For each business component from which you want to extract fax addressing information for recipients, set the Recipient Fax Address Field user property to the appropriate field, such as the Fax Address field for the Contact business component.

For example, in views based on the Contact business component, your users may enter customer fax information in the Fax Phone # field. The Fax Address field is a calculated field that prepares fax addressing data obtained from the Fax Phone # field for use with your fax server. In this case, the Recipient Fax Address Field user property should specify the Fax Address field.

The formatting for fax-related fields depends on your fax server requirements. The calculated value for Fax Address or equivalent fields contains fax formatting that applies to RightFax. All such field values must be modified as appropriate for your specific fax integration.

For example, for RightFax, the calculated value should specify the email account that is monitored by RightFax as well as the field, such as Fax Phone #, that contains the specific fax address for the recipient.

By default, the Fax Address field for the Contact business component, for example, contains the following value for the Calculated Value field in Siebel Tools:

```
IIF([Fax Phone #] IS NOT NULL, '"/fax='+ToChar([Fax Phone #])+'/"
<replace_me@replace.me.com>', "")
```

The above definition should work correctly for Captaris RightFax, if you modify `replace_me@replace.me.com` to specify your fax integration's email address. Retain the <> brackets enclosing the email address. Also make sure that a space character precedes the opening bracket (<) enclosing the email address.

For more information about configuring user properties and working with Siebel Tools, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*.

Settings for Internet SMTP/POP3 Server Driver

Table 71 on page 359 lists the settings for the driver record for the Internet SMTP/POP3 Server driver. These settings appear in the Communications Drivers and Profiles view in the Administration - Communications screen.

CAUTION: The values shown in Table 71 on page 359 are presented for reference only. Do not modify the predefined values for this driver.

Table 71. Internet SMTP/POP3 Server Driver Settings

Field Name	Value
Name	Internet SMTP/POP3 Server
Channel Type	Email
Inbound	Selected by default
Outbound	Selected by default
Interactive	
Channel String	POP3SMTP
Library Name	sscmpop3

Driver Parameters for Internet SMTP/POP3 Server Driver

Table 72 on page 361 lists the supported parameters for the Internet SMTP/POP3 Server communications driver. Some parameters are applicable only to inbound email processing of messages retrieved from a POP3 server (for Siebel Email Response), and some are applicable only to outbound email sent using an SMTP server.

NOTE: Each parameter indicated as required may be required only for profiles used for inbound POP3 messages, or only for profiles used for oubound SMTP messages, depending on the function of the parameter.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Charset	No	UTF-8	Specifies the character set that may apply to an outbound email message. For an outbound email message that is a reply to an inbound email message, this parameter has effect only when the character set of the inbound message cannot be determined. For all other outbound email messages, the value of this parameter for the driver profile determines the character set used. If this parameter is not defined, the UTF-8 character set is used. When an inbound email message is received that uses a specific character set, the character set is stored on the activity record for the inbound message. When an agent replies to this message, the character set is retrieved from the activity record, so the reply uses the same character set. If, however, the character set of the inbound email message cannot be determined, then the character set to use is the value of the Charset parameter for the driver profile. If this parameter is not defined, the UTF-8 character set is used. For more information about character sets and how they apply to Siebel Business Applications, see *Global Deployment Guide*.
Convert Incoming To Plain Text	No	FALSE	Converts incoming HTML email to text/plain format. You might set this parameter to TRUE, for example, if your incoming email accounts receive HTML spam messages that are image-based and consequently cannot be easily filtered out. When this setting is TRUE, such images will not display to your contact center agents.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Create Plain Text From HTML	No	FALSE	If this parameter is TRUE, and an outbound message being sent contains text tagged with HTML but no plain text, synthesize (extract) the plain text from the HTML and send it as an alternate to the HTML. This situation may apply in these cases: ■ User sends email using Send Email command, when Send Email: Default Message Format is set to HTML in the Outbound Communications options of the User Preferences screen. ■ User sends reply email using Siebel Email Response, when Email Response: Default Message Format is set to HTML in the Outbound Communications options of the User Preferences screen. ■ User sends outbound communication request for the email channel, for which a template contains either or both of the following: HTML template text or at least one HTML template item. Sending separate plain-text and HTML versions of the same message body is called MIME multipart/alternative support. To increase performance when sending HTML messages to recipients known to use email clients that can read or convert HTML messages, set this parameter to FALSE. **NOTE:** The results of setting this option to TRUE may not be compatible with all major email clients. This factor must be considered before you set this parameter to TRUE. For more information about HTML email, see "Using HTML Email" on page 356. For more information about setting outbound communications user preferences, see "Preference Settings for Outbound Communications" on page 306.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Delete Processed Messages	No	TRUE	TRUE or FALSE Specifies whether processed messages are deleted: ■ If TRUE, then messages are deleted from the "incoming" email directory after they are processed. ■ If FALSE, then messages are moved to the "processed" email directory after they are processed. See also the descriptions for the Incoming Email Directory and Processed Email Directory parameters.
Delivery Status Domain	No		Specifies the domain name for the email server that will handle delivery status notification messages. Such messages may be generated in response to outbound mail messages sent using this profile. Together, this parameter and the Delivery Status Mailbox parameter specify the email address used in the SMTP command MAIL FROM—for delivery status notification messages only. **NOTE:** Use these parameters *only* if you require that delivery status notification messages be processed on a separate email server than is used for processing normal customer responses. Regular customer responses to outbound messages are sent to the email address specified using the Reply-To Address or From Address parameters. See also the description of the Delivery Status Mailbox parameter.
Delivery Status Mailbox	No		The email address, in the domain specified using the Delivery Status Domain parameter, that is to receive delivery status notification messages. For more information, see the description of the Delivery Status Domain parameter.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Enable SSL for Backup SMTP	No	FALSE	TRUE or FALSE If TRUE, SSL (Secure Sockets Layer) is used to encrypt the SMTP session (for the backup SMTP server) between the email server and the Siebel Server. Specify the backup SMTP server name using the SMTP Backup Server parameter. For more information, see also the descriptions for the Enable SSL for SMTP, Use EHLO, and SMTP Backup Server parameters.
Enable SSL for POP3	No	FALSE	TRUE or FALSE If TRUE, SSL (Secure Sockets Layer) is used to encrypt the POP3 session between the email server and the Siebel Server. If an SSL session cannot be negotiated, the POP3 connection is aborted. Port 995 is normally used for POP3 over SSL.
Enable SSL for SMTP	No	FALSE	TRUE or FALSE If TRUE, SSL (Secure Sockets Layer) is used to encrypt the SMTP session between the email server and the Siebel Server. The STARTTLS command is executed after the EHLO command has executed successfully. If the STARTTLS command fails, the connection is aborted. An attempt will be made to use the backup SMTP server instead. Specify the SMTP server name using the SMTP Server parameter. See also the descriptions for the Enable SSL for Backup SMTP and Use EHLO parameters.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Failed Email Directory	No		The directory where failed incoming email messages retrieved from the POP3 server are stored. An incoming message is considered to be a failed message if any error or exception occurs while the message is parsed. By default, failed email messages are saved to the bin/failed subdirectory of the Siebel Server installation directory.
From Address	Yes		The sender's email address for outbound communications. Use one of the following two forms: ■ *sendername*@*companyname*.com ■ *sender_display_name* <*sendername*@*companyname*.com> Use the second form above if you want the recipient to see a display name for the sender, in addition to the email address. The specified address is passed to the SMTP server in the "MAIL" command, and is used as the "From:" address in the message header. **NOTE:** End users who create email/fax profiles for personal use can specify their own email address, as appropriate, for the From Address parameter, as a profile parameter override. For more information, see "Creating Communications Profiles for Personal Use" on page 323. If no parameter value is provided, the driver tries to use the "SenderAddress" that is passed to the driver when it is asked to compose and send an email. **NOTE:** For a profile used for inbound communications, which is specified for a response group, the From Address parameter represents the recipient's email address.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Incoming Email Directory	No		The directory where email messages retrieved from the POP3 server are stored prior to processing. By default, incoming email messages are saved to the `bin/incoming` subdirectory of the Siebel Server installation directory. Messages move to the processed email directory after processing. If Delete Processed Messages = TRUE, then messages are deleted from the processed directory. See also the descriptions for the Delete Processed Messages and Processed Email Directory parameters. **NOTE:** As inbound email messages are processed, they are saved in a series of formats and filenames in the incoming directory and related directories. Corresponding attachment files to the Email - Inbound activity record are saved to the Siebel File System. For more information, see the subsection about activity records and attachments, in "Administering Communications Inbound Receiver" on page 256. For more information about inbound message processing, see *Siebel Email Response Administration Guide*.
LogDebug	No	FALSE	TRUE or FALSE If TRUE, data output to the log file is more detailed. The additional detail in the log file can help you to troubleshoot. Log files are created in the log subdirectory of the Siebel Server installation directory. Log files for the Internet SMTP/POP3 Server driver have the name POP3SMTP_*xxxxxxxxxxxxxxx*.log, where *xxxxxxxxxxxxxxx* is a unique string of numerals.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Loopback Detection Interval	No	90	An integer value that specifies the number of seconds to flag any inbound messages as loopback messages that were received from a sender from which another message was just received: ■ A message is determined to be a loopback message if the following are true: ■ The value for Loopback Detection Interval is greater than 0 (zero), and The message is received no later than the specified number of seconds after a previous message was received from the same email address. This parameter is based on the premise that a message received within a short period of time has a high likelihood of being an auto-response. For example, if your system receives a message and sends out an auto-acknowledgement, that auto-acknowledgement message could, when processed by the recipient, generate a return auto-response. Setting this parameter can prevent message looping that may otherwise occur in this instance. If you set this value too high, you risk rejecting valid follow-up messages. If you set it too low, the driver may not correctly detect an inbound message that is actually just an auto-response. Loopback messages are not auto-acknowledged or auto-responded. They may either be moved to a "loopback" email directory, or processed as are other messages. See also the descriptions for the Loopback Email Directory and Process If Loopback Detected parameters.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Loopback Email Directory	No		The directory where email messages determined to be loopback messages are stored, if they are not to be processed. By default, email messages determined to be loopback messages are saved to the bin/loopback subdirectory of the Siebel Server installation directory. See also the description for the Loopback Detection Interval and Process If Loopback Detected parameters.
Max Line Length	No	74	An integer value that specifies the maximum line length, in characters, for outbound messages sent using this profile. The parameter value is enforced only if you are using a text/plain encoding or text/html encoding parameter value of None, 7bit, or 8bit, *and* one or more text lines exceed 998 bytes without a carriage return line feed. (The line length in bytes is also determined by the character set in use.) The specified value must be between 40 and 124, inclusive. If no value is specified, the default line length is 74 characters. If the encoding in effect is quoted-printable or base64, then the Max Line Length parameter never needs to be enforced. See also the descriptions of the text/html encoding and text/plain encoding parameters, and the Charset parameter.
Max Sends Per Session	No	0	An integer value that forces the driver to log off of the SMTP server after sending the number of messages specified using this parameter. The driver will then re-logon for the next send request. (Some SMTP servers may have problems sending a large number (1000 or more) of messages in one session.) A value of "0" (zero) means that the driver never logs off of the SMTP server.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Parse Embedded Messages	No	TRUE	TRUE or FALSE Specifies parsing for embedded messages within inbound email messages. If TRUE (the default), embedded messages are parsed in order to display the message content as attachments within the Siebel application. If FALSE, embedded messages are not parsed, but are instead turned into attachments with the EML file extension. Programs that can view MIME files, including Microsoft Outlook Express, can be used to view EML files. For details, see "Activity Attachments Stored for Inbound Email" on page 259.
PollingInterval	No	30	The number of seconds between checks for new inbound mail arriving on the POP3 server.
POP3 Account Name	Yes		The account name for the POP3 mailbox from which to retrieve inbound communications.
POP3 Account Password	Yes		The password for the POP3 mailbox account.
POP3 Server	Yes		The host name or IP address of the machine on which the Internet POP3 server is running, as appropriate for your network configuration. **NOTE:** If you specify a host name rather than an IP address for this parameter value, a corresponding entry must be included in your DNS server or the local hosts file. If an IP address parameter value works correctly, but a problem occurs when you have specified a host name parameter value, consult your network system administrator to verify that host name lookup is correctly configured.
POP3 Server Port	No	110	The port used by the Internet POP3 server to listen for new socket connections.
POP3 Timeout	No	60	Number of seconds to wait for a response from the POP3 server before timing out.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Priority	No	3	Specifies the priority level for an outbound message, such as a reply to an inbound customer email message. Depending on the email client program a customer uses, additional information may be displayed with the message received by the customer—for example, an exclamation point may be displayed for a high-priority message. The priority level can be specified using an Internet standard number scheme (1 through 5) or the Microsoft priority scheme (Low, Normal, or High).
Processed Email Directory	No		The directory where email messages retrieved from the POP3 server are moved, after they have been processed—unless they are to be deleted instead. The Delete Processed Messages parameter is used to determine whether processed messages are deleted or moved to this directory. By default, if Delete Processed Messages is FALSE, the "processed" Email Directory is the bin/processed subdirectory of the Siebel Server installation directory. See also the description for the Delete Processed Messages parameter.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Process If Loopback Detected	No	TRUE	TRUE or FALSE Specifies how to handle an incoming message that is determined to be a loopback message. ■ If TRUE, then a loopback message is processed by Communications Inbound Receiver and Communications Inbound Processor, and associated workflows, as are any other messages. ■ If FALSE, then a loopback message is saved to the "loopback" directory. Loopback messages are not auto-acknowledged or auto-responded. The Loopback Detection Interval parameter is used to determine whether a message is flagged as a loopback. See also the descriptions for the Loopback Detection Interval and Loopback Email Directory parameters.
Reply-To Address	No		An optional reply-to email address for the message, if it is different than that specified with the From Address parameter. Multiple email addresses can be specified, separated by semicolons. If a reply-to address is specified, then outgoing messages will have a "Reply-To:" header field added to the message envelope with the specified value. Use one of the following two forms: ■ *sendername*@*companyname*.com ■ *sender_display_name* <*sendername*@*companyname*.com> Use the second form above if you want the recipient to see a display name for the sender, in addition to the email address.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
Return Attachments	No	TRUE	TRUE or FALSE Specifies whether to accept or discard attachments: ■ If TRUE, any attachments parsed from the incoming message are passed into the Siebel application. ■ If FALSE, the attachments are simply discarded.
Save Sent Messages	No	FALSE	TRUE or FALSE Specifies whether to save sent messages: ■ If TRUE, sent messages are saved into the "sent" email directory. ■ If FALSE, sent messages are not saved. Each message is saved as a separate, MIME-encoded file, in the exact format in which it was sent to the SMTP server. See also the description of the Sent Email Directory parameter.
Sent Email Directory	No		The directory where sent messages are saved, if Save Sent Messages is TRUE. By default, if Save Sent Messages is TRUE, messages are saved into the bin/sent subdirectory of the Siebel Server installation directory. See also the description of the Save Sent Messages parameter.
Siebel Server	No		The name of the Siebel Server that is to handle the delivery of outbound communications sent using this profile. For more information, see "Specifying the Siebel Server for Communications Outbound Manager" on page 264.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
SMTP Account Name	No		Specifies the account name to use for logging in to the Internet SMTP server (using the AUTH LOGIN command). Use this parameter if your SMTP server requires authentication. When you define this parameter, also specify a password using the SMTP Account Password parameter. Specify the SMTP server name using the SMTP Server parameter. **NOTE:** You can use authentication independent of using SSL encryption using the Enable SSL for SMTP parameter. If you use both authentication and SSL encryption for SMTP, the session is encrypted before authentication occurs, so that the account name and password are not sent in clear text. See also the descriptions of the Enable SSL for SMTP and SMTP Server parameters.
SMTP Account Password	No		Specifies the password for the account you use for logging in to the Internet SMTP server. See also the description of the SMTP Account Name parameter.
SMTP Backup Account Name	No		Specifies the account name to use for logging in to the backup Internet SMTP server (using the AUTH LOGIN command). Use this parameter if your backup SMTP server requires authentication. When you define this parameter, also specify a password using the SMTP Backup Account Password parameter. Specify the SMTP server name using the SMTP Backup Server parameter. For more information, see the description of the SMTP Account Name parameter. See also the descriptions of the Enable SSL for Backup SMTP and SMTP Backup Server parameters.
SMTP Backup Account Password	No		Specifies the password for the account you use for logging in to the backup Internet SMTP server. See also the description of the SMTP Backup Account Name parameter.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
SMTP Backup Server	No		The host name or IP address of the machine on which the backup Internet SMTP server is running, as appropriate for your network configuration. You use this parameter when you have implemented a failover strategy for your SMTP server handling outbound email or fax communications. If the primary SMTP server goes down, outbound communications using profiles in which these parameters are defined will be sent to the backup SMTP server instead, until the primary SMTP server has restarted: ■ If you have implemented failover support for your SMTP server, then you must use both the SMTP Server and SMTP Backup Server parameters. ■ If you have not implemented failover support for your SMTP server, use the SMTP Server parameter only. Do not define the SMTP Backup Server parameter. **NOTE:** If a message cannot be sent to an SMTP server because it is down, the driver returns a message to the Communications Outbound Manager, which will lead to another attempt to send the message at a later time. (This behavior is not specific to a failover configuration.) For more information, see the description of the SMTP Server parameter.
SMTP Backup Server Port	No	25	An integer value that specifies the port to use for connecting to the backup Internet SMTP server. See also the description for the SMTP Backup Server parameter.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
SMTP Server	Yes		The host name or IP address of the machine on which the Internet SMTP server is running, as appropriate for your network configuration. **NOTE:** If a message cannot be sent to an SMTP server because it is down, the driver returns a message to the Communications Outbound Manager, which will lead to another attempt to send the message at a later time. **NOTE:** If you specify a host name rather than an IP address for this parameter value, a corresponding entry must be included in your DNS server or the local hosts file. If an IP address parameter value works correctly, but a problem occurs when you have specified a host name parameter value, consult your network system administrator to verify that host name lookup is correctly configured. See also the description of the SMTP Backup Server parameter.
SMTP Server Port	No	25	The port used by the Internet SMTP server to listen for new socket connections.
SMTP Timeout	No	60	The number of seconds to wait for a response from the Internet SMTP server before timing out. This parameter applies to both the primary SMTP server and the backup SMTP server.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
text/html encoding	No	Auto	Specifies content-transfer encoding behavior for text/html message parts. Values may be one of the following: base64, quoted-printable, 7bit, 8bit, Auto, or None. Auto is the default value if this parameter is not set or a value other than one of these is specified. For Auto, the driver uses the simplest encoding that can safely be transported over SMTP, without needing to wrap the text: ■ If no line exceeds 998 bytes and all characters are 7-bit, 7-bit encoding is used. ■ If the character stream exceeds the SMTP transport restriction of 998 bytes per line, and most of the character data is 7-bit, quoted-printable encoding is used. If much of the character data is 8-bit, base-64 encoding is used. For None, the driver uses 7-bit or 8-bit encoding and wraps the text, if necessary, so that it can be safely transported over SMTP: ■ If the character data is all 7-bit, 7-bit encoding is used. ■ If any of the character data is 8-bit, 8-bit encoding is used. ■ In each case, if the character stream exceeds the SMTP transport restriction of 998 bytes per line, hard carriage return line feeds are inserted, so that no line has more than the number of characters specified using the Max Line Length parameter. See also the description of the Max Line Length parameter.

Table 72. Internet SMTP/POP3 Server Driver Parameters

Name	Required	Default Value	Description
text/plain encoding	No	Auto	Specifies content-transfer encoding behavior for text/plain message parts. **NOTE:** For a detailed description of valid options, see the description for the text/html encoding parameter.
Use EHLO	No	TRUE	TRUE or FALSE Set to TRUE if you are using SMTP extension (ESMTP) commands on a particular email server. Causes the first command sent to the server to be EHLO, which enables the extension commands to be available. Set to FALSE if you are using an SMTP server that cannot handle the EHLO command.

Supporting Email Interactivity

The User-Interactive Email driver supports email interactivity within the Siebel application, enabling agents to use the communications toolbar for communications work items of the email channel type.

The User-Interactive Email driver supports Siebel Email Response when deployed with Siebel Universal Queuing.

Within the applicable communications configuration, events and commands for work items for these products must reference profiles for this communications driver. The communications configurations provided by Siebel Systems support multichannel communications for voice and email work items.

NOTE: Using this user-interactive driver requires that Siebel Universal Queuing or an equivalent queuing integration is deployed in order to route email work items to agents. When such a work item is routed to an agent, the agent is notified by the Accept Work Item button blinking on the communications toolbar.

For more information about Siebel Universal Queuing, see *Siebel Universal Queuing Administration Guide*.

In the Communications Drivers and Profiles view, note that the two user-interactive drivers, along with the CTI drivers, have the Interactive flag set. For more information, see "About Communications Drivers and Profiles" on page 35 and "Configuring Communications Drivers and Profiles" on page 44.

NOTE: The Siebel-provided communications driver Internet SMTP/POP3 Server, although also of channel type Email, does not support user-interactive functionality for email. For more information about the function of this driver, see "Interfacing with Email and Fax Servers" on page 355.

User-Interactive Email Driver

The User-Interactive Email driver enables agents to use the communications toolbar for inbound email communications work items, in support of Siebel Email Response, when deployed with Siebel Universal Queuing.

Settings for User-Interactive Email Driver

Table 73 on page 378 lists the settings for the driver record for the User-Interactive Email driver. These settings appear in the Communications Drivers and Profiles view in the Administration - Communications screen.

CAUTION: The values shown in Table 73 on page 378 are presented for reference only. Do not modify the predefined values for this driver.

Table 73. User-interactive Email Driver Settings

Field Name	Value
Name	User-interactive Email
Channel Type	Email
Inbound	
Outbound	
Interactive	Selected by default
Channel String	Interactive Email
Library Name	sscmimed

User-interactive Email Driver Parameters

Table 74 lists the supported driver parameters for the User-interactive Email communications driver.

Parameters are sent to the driver handle when the driver is being initialized (CreateISCDriverInstance method), and when the driver handle requests a service (RequestService method). For more information about the driver handle methods, see Appendix A, "Developing a Communications Driver."

Table 74. User-interactive Email Driver Parameters

Name	Required	Default Value	Description
AutoAccept	No	FALSE	TRUE or FALSE If TRUE, and Siebel Universal Queuing dispatches a work item, the driver will automatically accept the work item and put the work item in an active state.
AutoSuspend	No	TRUE	TRUE or FALSE If TRUE, and Siebel Universal Queuing dispatches a work item, the driver will automatically accept the work item and put the work item in a suspended (paused) state.
LogDebug	No	FALSE	TRUE or FALSE If TRUE, data output to the log file is more detailed.
LogFile	No	mail_ {@UserName}. log	The name of the file where logging information will be written. If this parameter is empty, no log file is generated. Logging data is always appended to this file. Log files are created in the log subdirectory of the Siebel Server installation directory, depending on the client type and on your communications deployment. Optionally, if you want to use a single log file in a multichannel environment, you can specify the same log file name for all interactive drivers you are using.
MaxLogKB	No	128	Specifies the maximum size of the log file, in kilobytes (KB). When the log file is full, it is emptied completely, then logging begins again.

Table 74. User-interactive Email Driver Parameters

Name	Required	Default Value	Description
ReleaseLogHandle	No	TRUE	Indicates that the log file handle for each agent is released periodically, after logs have been generated. The default setting of TRUE provides better performance. When ReleaseLogHandle is FALSE, however, some log files may not be generated if the number of concurrent users exceeds the file-handle capacity provided by the operating system.
Simulate	No	FALSE	When this parameter is set to TRUE, this driver supports the Communications Simulator. This parameter need not be explicitly set in order to use the Communications Simulator. For more information, see "Enabling Session Communications and Simulation" on page 240 and "Simulating a Communications Environment" on page 211.

User-interactive Email Commands

Table 75 on page 381 lists and explains usage for commands specific to the communications driver User-interactive Email.

The commands you define, in which driver commands listed here are specified as device commands, support the command parameters documented in Chapter 5, "Configuring Events and Commands." In your command definitions, you can specify custom subparameters for command parameters of type Group.

An asterisk (*) before a parameter name means that the parameter is optional for this command. Command parameters are described in Table 76 on page 381.

Most of the commands in Table 75 on page 381 correspond to commands defined in the communications configuration, in which they are specified as the device command. They may also correspond to a communications toolbar button.

For more information, see Chapter 6, "Configuring User Interface Elements." For descriptions of the communications toolbar, see Chapter 11, "Communications Operations for End Users."

For commands for which the TrackingID parameter is optional, the command operation is performed on the work item identified by the TrackingID parameter value. If this parameter is not provided, the command operates on the first available work item.

Table 75. User-interactive Email Commands

Command Name	Parameters	Description
AcceptEmailWork	*TrackingID	Accept a new email work item.
ReleaseEmailWork	*TrackingID	Release a new email work item.
ResetEmailState		Reset the state of the User-Interactive Email driver, in case it has gotten out of sync.
ResumeEmailWork	*TrackingID	Resume a new email work item.
SimNewEmailWork		Simulate a new email work item. Accepts any parameters received (by creating Param subparameters for the command data definition).
SuspendEmailWork	*TrackingID	Suspend a new email work item.

User-interactive Email Command Parameters

Table 76 on page 381 lists User-interactive Email command parameters and describes their usage. Table 75 on page 381 identifies the commands that use each of these parameters.

Within the communications configuration, command parameters and values to pass to the User-Interactive Email driver can be specified within command data definitions as subparameters of the Param parameter. For more information, see "Command Data" on page 142.

Table 76. User-interactive Email Command Parameters

Command Parameter	Description
TrackingID	The designated tracking ID of a work item, to identify the specific work item on which an operation is to be performed.

User-interactive Email Events

Table 77 lists and describes User-interactive Email events that Siebel Communications Server may receive. Events listed may be referenced as device events in communications configurations with which profiles for the User-Interactive Email driver are associated.

Table 77. User-interactive Email Events

Event Name	Description
EventEmailWorkArrived	A new email work item has arrived, having been delivered by Siebel Universal Queuing.
EventEmailWorkReleased	The agent has released an email work item.
EventEmailWorkResumed	The agent has resumed an email work item.
EventEmailWorkRevoked	An email work item has been revoked by Siebel Universal Queuing.
EventEmailWorkStarted	The agent has accepted an email work item and the work item is now active.
EventEmailWorkSuspended	The agent has suspended an email work item.

User-interactive Email Event Data

The attributes received with an email event are determined by the workflow process for Siebel Email Response. The workflow can be extended to pass other attributes to Siebel Universal Queuing, which in turn passes them to the agent's communications toolbar as event attributes. The following attributes must be passed to the workflow process for submission to Siebel Universal Queuing:

- **LongDescription.** The detailed description of the work item. This attribute is used in the Accept/Reject dialog box, where the descriptive information is displayed in order for the agent to decide whether to accept or reject the work item.
- **ShortDescription.** The brief description of the work item. This attribute is used in the Work Items list in the communications toolbar. For example, after an agent accepts an email work item, the Work Items list uses text from this attribute to identify an email. For example, the attribute value might be a phrase like:

  Email from *sender_name*@*company_name*.com
- **ChannelAddress.** This attribute stores applicable channel information for the work item that is intended to be displayed in the Channel Target Address field in the All Channel Items view (in the Administration - Communications screen). The email address the work item was sent to may be stored using an attribute value like:

  support@*company_name*.com

The fixed work item attributes received for any work item are described in "Work Item Attributes" on page 190.

Configuring Client-Side Integration for Send Email

The Send Email command can be deployed to support three different email client options. Users can be configured to use Lotus Notes or Microsoft Outlook to send outbound email messages, or configured to use the native Siebel email client.

NOTE: The third-party email client integrations described in this section, for Lotus Notes or Microsoft Outlook, are supported for end users on Microsoft Windows client machines only. End users on UNIX client machines must use the native Siebel email client.

The following list describes assumptions that are made for Send Email integrations:

- Send Email integrations for Microsoft Outlook are assumed to use Microsoft Exchange Server as the email server.
- Send Email integrations for Lotus Notes are assumed to use Lotus Domino as the email server.

The third-party email client integration options use email client software on each user's client machine, and do not use Siebel Communications Server.

By contrast, the native Siebel email client integrates with your email server, using Siebel Communications Server and the Internet SMTP/POP3 Server communications driver. If third-party email client integration is not configured, then the Siebel application uses the native Siebel email client. For more information, see "Interfacing with Email and Fax Servers" on page 355.

You can configure your system so that all users use the same email client for the Send Email command. Alternatively, you can support multiple email clients and let individual users specify their preferred email client.

For information about using the Send Email command with a third-party email client integration, see "Sending Email Using Lotus Notes or Microsoft Outlook" on page 327.

For information about activity records and email, see "Creating Activities for Send Commands" on page 334, including the subsection about matching contacts for these activity records.

For information about supported versions of third-party software, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

NOTE: If you are using Microsoft Outlook for the Send Email command, note that communications between Outlook and the Siebel application, which is handled by scripts for the email form, may cause alert messages to be displayed. Whether such messages are displayed, and which messages may be displayed, depends on the current security settings for Outlook.

Email-related error messages received by users will use the language applicable to the source of the message. For example, an error message returned from a French version of Microsoft Outlook will be in French.

Errors generated from Siebel software may use the language and locale of the Siebel Server, such as in the execution of a script used in the email form, or use the language and locale the user specified as a user preference or temporary setting for sending messages.

NOTE: For Microsoft Outlook, use separate email forms for each code page. Configure each form according to the requirements for using each language version of Microsoft Windows that you are supporting.

Process of Configuring Third-Party Email Integrations

Configuring third-party email client integrations requires the following general steps, which are described in more detail in the referenced subsections:

1. Create and configure one or more directories to use for temporary storage of email attachment files. See "Creating and Configuring the Attachments Directory" on page 384.
2. Install the Siebel email form for Lotus Notes or Microsoft Outlook. See the following sections:
   - "About Installing the Siebel Email Form" on page 385
   - "Installing the Siebel Email Form for Lotus Notes" on page 386
   - "Installing the Siebel Email Form for Microsoft Outlook" on page 387
3. As appropriate for your Siebel client deployment, set component parameter values for the Application Object Manager, or set parameter values for the application's configuration file, such as uagent.cfg for Siebel Call Center. See the following sections:
   - "Overview of Completing Email Client Configuration" on page 388
   - "Completing Configuration for Lotus Notes" on page 389
   - "Completing Configuration for Microsoft Outlook" on page 390
   - "Email Client Parameters" on page 392
4. (Optional) Set user preferences for each end user. See sections listed for Step 3, and see "Preference Settings for Outbound Communications" on page 306.

Creating and Configuring the Attachments Directory

If the Send Email command is configured to generate activity records, then when a Siebel user sends an email message using Lotus Notes or Microsoft Outlook, email attachment files are used to create attachment records for this activity record. A temporary storage directory must be defined to facilitate transferring attachments in this manner.

When the email message is sent, the email form invokes a script, which in turn invokes a Siebel business service method that transfers the files from this attachments directory to the Siebel Server and deletes the files from this directory.

For more information about activity records and email, see "Setting Communications User Preferences" on page 305 and "Creating Activities for Send Commands" on page 334.

The attachments directory is typically (but not necessarily) located on the Siebel Server machine. For users on the Siebel Mobile Web Client who will use Lotus Notes or Microsoft Outlook for the Send Email command while disconnected from the network, an attachments directory on each local client *must* be specified.

The attachments directory must be accessible from each Microsoft Windows machine where the Siebel client may run. Configure access to this directory as follows:

- For all Siebel users on applicable client machines, provide write access. Do not provide read and execute access.

- For the system user login that started the Siebel Server, read and execute access *must* be provided.

The attachments directory you configure in this section is later specified for use for all users or for individual users, as described in "Overview of Completing Email Client Configuration" on page 388.

Next, install the Siebel email form for Lotus Notes or Microsoft Outlook.

About Installing the Siebel Email Form

In order for users to be able to use either Lotus Notes or Microsoft Outlook for the Send Email command, Siebel-customized files described in this section must be copied to designated locations on server or client machines, and applicable email forms must be made available to your users.

For each of these email client programs, an email form is installed which has been set up for use with Siebel Business Applications. Initiating an email message from the form invokes a script that interacts with the Siebel application. The script updates the activity record for the email message and generates attachment records for this activity record, based on any email attachment files. If the email is canceled, the activity is deleted.

The Notes or Outlook form you install in this section is later specified for use for all users or for individual users, as described in "Overview of Completing Email Client Configuration" on page 388.

NOTE: Siebel Systems does not support customizations of Notes or Outlook forms that are not described in this book.

Locations of Source Files for Siebel Email Forms

The Siebel email form for Lotus Notes is part of a Notes database file. The Siebel email form for Microsoft Outlook is part of an Outlook template file.

When you create a Siebel network image from Siebel media files, the Lotus Notes database file Siebel7Email.nsf is extracted to the directory:

```
Windows\Client_Ancillary\Send_Email_Client\enu\lotus_notes
```

You install Siebel7Email.nsf from this location. This database file supports only ENU (U.S. English) in the user interface. However, error messages reflect the currently deployed supported language, if you choose to use this file in deployments other than ENU.

For information on how to create a Siebel network image, see the *Siebel Installation Guide* for the operating system you are using.

NOTE: The Microsoft Outlook template file containing the Siebel email form is not provided by Siebel Systems. For details on how to configure the email form, see Siebel SupportWeb.

Installing the Siebel Email Form for Lotus Notes

This section describes how to install the Siebel email form for Lotus Notes. The email form may be installed on the server machine where Lotus Domino is running, or on each client machine where Lotus Notes is running.

For users on the Siebel Mobile Web Client who will use Lotus Notes for the Send Email command while disconnected from the network, the email form must be installed locally on the client machine.

If you deploy the form on local client machines, then you must redistribute any subsequent updates to all client machines.

NOTE: For additional deployment options for the Notes database and email form, contact your administrator for Lotus Notes and Lotus Domino.

The Siebel email form for Lotus Notes, Siebel Memo, is contained within the Notes database file Siebel7Email.nsf.

You can rename the default Siebel email form, or add to this database additional Siebel email forms based on the default. If you do either of these, make sure that the names of all supported Siebel forms are included in the List of Values type for the Siebel/Lotus Form drop-down list, located in the Outbound Communications options of the User Preferences screen.

To specify a custom form as a global default, then you must also specify its name using the Siebel/Lotus Form server component parameter (for Siebel Web Client deployments) or the LotusForm configuration file parameter (for Siebel Mobile Web Client), as described in "Email Client Parameters" on page 392.

To install the Siebel email form for Lotus Notes

1. Locate the Notes database file Siebel7Email.nsf on the Siebel network image, as described in "About Installing the Siebel Email Form" on page 385.
2. Copy this file to the clipboard.
3. Paste the Siebel7Email.nsf file into the appropriate destination location. Do *one* of the following:
   - On the Lotus Domino server machine, paste the file into the Data subdirectory in the Lotus Domino installation directory. For example, the location may be:

     `D:\Lotus\Domino\Data\Siebel`

     In this example, a Siebel subdirectory has been created in the Data directory.
   - On each client machine, paste the file into the Data subdirectory in the Lotus Notes client installation directory. For example, the location may be:

     `D:\Lotus\Notes\Data`
4. Right-click on the file Siebel7Email.nsf, select Properties, uncheck the check box for the Read-only attribute (in order to allow read-write access), and click OK.

   NOTE: If the file Siebel7Email.nsf has the Read-only attribute set, then error messages may be displayed to end users when they use the Send Email command.

5 On each client machine, add a line to the notes.ini file in your Lotus Notes installation, to indicate the location of the Notes database file, Siebel7Email.nsf. Do *one* of the following:

- If you pasted Siebel7Email.nsf onto the Lotus Domino server machine, add an entry like the following (on one line):

  `SIEBELFORMPATH = dominoserver_hostname\dominoserver_name:Siebel\Siebel7Email.nsf`

  where *dominoserver_hostname* is the name of the host machine on which the Lotus Domino server is running and *dominoserver_name* is the name of the Lotus Domino server itself.

  In this example, a Siebel subdirectory has been created in the Data directory. A variable definition in the form shown above is appropriate for an actual location such as the following path on the Lotus Domino server machine:

  `D:\Lotus\Domino\Data\Siebel\Siebel7Email.nsf`

- If you pasted Siebel7Email.nsf onto each client machine, add an entry like one of the following (on one line):

  `SIEBELFORMPATH = Siebel7Email.nsf`

  `SIEBELFORMPATH = :D:\Lotus\Notes\Data\Siebel7Email.nsf`

  As shown in the first example above, you can simply specify the name of the file, if it is located in the Notes database directory.

  CAUTION: If a full path on the client machine is specified as the value for SIEBELFORMPATH, then this variable *must* be defined using a leading colon (:), as shown in the second example above. Otherwise, the file will not be able to be opened.

  (If the file is added to users' mail files on each client machine, then the SIEBELFORMPATH variable may not need to be defined. For more information, refer to your third-party vendor documentation for Lotus Notes.)

  The Notes email form Siebel Memo is now ready to be used. To finish configuring your use of Lotus Notes, go to "Overview of Completing Email Client Configuration" on page 388.

Installing the Siebel Email Form for Microsoft Outlook

This section describes how to install the Siebel email form for Microsoft Outlook. The email form may be installed on the server machine where Microsoft Exchange Server is running, or on each client machine.

For users on the Siebel Mobile Web Client who will use Microsoft Outlook for the Send Email command while disconnected from the network, the email form must be installed locally on the client machine. Note that deploying the form on local client machines has implications for subsequent updates. Also, note that a form in the Personal Form Library takes precedence over a form of the same name in the Organizational Form Library.

NOTE: For additional deployment options for the Outlook template and email form, contact your administrator for Microsoft Outlook and Microsoft Exchange Server.

The default name for the Siebel email form for Microsoft Outlook is IPM.Note.Siebel. This form must be contained within an Outlook template file. For more information, see "About Installing the Siebel Email Form" on page 385.

You can rename the default Siebel email form when you publish it, or publish additional Siebel forms based on the default. If you do either of these, make sure that the names of all supported Siebel forms are included in the List of Values type for the Siebel/Outlook Form drop-down list, located in the Outbound Communications options of the User Preferences screen.

To specify a custom form as a global default, then you must also specify its name using the Siebel/Outlook Form server component parameter (set on the AOM component, for Siebel Web Client deployments) or the OutlookForm configuration file parameter (for Siebel Mobile Web Client), as described in "Email Client Parameters" on page 392.

To install the Siebel email form for Microsoft Outlook

1. Locate the Outlook template file containing the Siebel email form. See "About Installing the Siebel Email Form" on page 385.
2. Optionally, you may choose to copy this file to the clipboard, and paste it somewhere on the network before continuing.
3. Publish the email form to the Organizational Form Library on Microsoft Exchange Server, or to the Personal Form Library on a local client machine:
   a. On the server machine or on the client machine, double-click the template file Siebel7Outlook.oft.
   b. At the prompt, specify Enable Macros.
   c. In Microsoft Outlook, choose Tools > Forms > Publish Form As.
   d. From Look In, choose Organizational Form Library (for installation on the Exchange Server machine) or choose Personal Form Library (for installation on a client machine).
   e. Specify the display name "Siebel" for this form, then save the form.

      The actual name of the email form is IPM.Note.Siebel.
   f. At the prompt, specify not to send the form definition to email recipients.
   g. Close the Outlook window without saving a message draft.

   The Outlook email form is now ready to be used.

To finish configuring your use of Microsoft Outlook, go to "Overview of Completing Email Client Configuration" on page 388.

Overview of Completing Email Client Configuration

To configure third-party email integration using Lotus Notes or Microsoft Outlook, several parameter values must be set appropriately, according to your Siebel client deployment:

- For Siebel Web Client deployments, configuration is on the Application Object Manager.

- For Siebel Mobile Web Client deployments, configuration is in the Siebel client's configuration file.

For more information about setting parameters for Siebel Server, including specifying component parameters using the Siebel Server Manager (GUI or command-line interface), see *Siebel System Administration Guide*.

For more information about specifying Siebel client parameters in the configuration file, see *Siebel System Administration Guide*.

Completing Configuration for Lotus Notes

This section describes the remaining steps for configuring the Siebel software to use Lotus Notes for the Send Email command.

To complete configuration for Lotus Notes email client

1. Set the following parameters to TRUE in the configuration file on the Siebel Server (for Siebel Web Clients) or on the Siebel Mobile Web Client, such as uagent.cfg for Siebel Call Center.

   These parameters are located in the [SWE] section of the configuration file. By default, these parameters are FALSE:

   - **EnableWebClientAutomation.** Allows the Siebel desktop integration object (ActiveX control) to be downloaded. (The source CAB file of this object is SiebelAx_Desktop_Integration.cab. The downloaded object name is Siebel Desktop Integration.)
   - **EnableEmailClientAutomation.** Allows the Siebel email automation object (ActiveX control) to be downloaded. (The source CAB file of this object is SiebelAx_OutBound_mail.cab. The downloaded object name is Siebel Email Support for Microsoft Outlook and Lotus Notes.)

   NOTE: For more information about the relevant CAB files and ActiveX controls, see the browser configuration section of the *Siebel System Administration Guide*.

2. As appropriate, specify a global default designating Lotus Notes as the email client your users will use for the Send Email command. (If this is not specified, then the Siebel native email client is the default.)

   - **Default Email Client.** For Siebel Web Client deployments, specify this as a component parameter for the Application Object Manager.
   - **DefaultMailClient.** For Siebel Mobile Web Client deployments, specify this parameter in the [EMail] section of the local Siebel client configuration file, such as uagent.cfg for Siebel Call Center.

   For more information, see "Email Client Parameters" on page 392.

3. Specify a global default designating Siebel Memo as the name of the email form you use with Lotus Notes:

   - **Siebel/Lotus Form.** For Siebel Web Client deployments, specify this as a component parameter for the Application Object Manager.

- **LotusForm.** For Siebel Mobile Web Client deployments, specify this parameter in the [EMail] section of the local Siebel client configuration file.

For more information, see "Email Client Parameters" on page 392.

4 Specify a global default specifying where email attachments will be temporarily stored:

- **Email Temporary Attachment Location.** For Siebel Web Client deployments, specify this as a component parameter for the Application Object Manager.
- **SiebelExtMailClientAttDir.** For Siebel Mobile Web Client deployments, specify this parameter in the [EMail] section of the local Siebel client configuration file.

For more information, see "Creating and Configuring the Attachments Directory" on page 384 and "Email Client Parameters" on page 392.

5 As appropriate for your deployment, end users may specify Siebel user preference settings for their email client. Users may set user preferences, for example, if you are supporting more than one email client option for Send Email.

Setting user preferences overrides the corresponding global defaults specified using the Default Email Client or Siebel/Lotus Form parameters, described in Table 78 on page 392. The global defaults are used if no user preference has been set.

Applicable user preferences in the Send Email section of the Outbound Communications options of the User Preferences screen include:

- **Email Client.** Set this to Lotus Notes.
- **Siebel/Lotus Form.** Set this to Siebel Memo.

For more information, see "Preference Settings for Outbound Communications" on page 306.

Completing Configuration for Microsoft Outlook

This section describes the remaining steps for configuring the Siebel software to use Microsoft Outlook for the Send Email command.

To complete configuration for Microsoft Outlook email client

1 Set the following parameters to TRUE in the configuration file on the Siebel Server (for Siebel Web Clients) or on the Mobile Web Client, such as uagent.cfg for Siebel Call Center.

These parameters are located in the [SWE] section of the configuration file. By default, these parameters are FALSE:

- **EnableWebClientAutomation.** Allows the Siebel desktop integration object (ActiveX control) to be downloaded. (The source CAB file of this object is SiebelAx_Desktop_Integration.cab. The downloaded object name is Siebel Desktop Integration.)

- **EnableEmailClientAutomation.** Allows the Siebel email automation object (ActiveX control) to be downloaded. (The source CAB file of this object is SiebelAx_OutBound_mail.cab. The downloaded object name is Siebel Email Support for Microsoft Outlook and Lotus Notes.)

NOTE: For more information about the relevant CAB files and ActiveX controls, see the browser configuration section of the *Siebel System Administration Guide*.

2 As appropriate, specify a global default designating Microsoft Outlook as the email client your users will use for the Send Email command:

- **Default Email Client.** For Siebel Web Client deployments, specify this parameter for the Application Object Manager.
- **DefaultMailClient.** For Siebel Mobile Web Client deployments, specify this parameter in the [EMail] section of the local Siebel client configuration file, such as uagent.cfg for Siebel Call Center.

If a global default is not specified, then the Siebel native email client is the default. For more information, see "Email Client Parameters" on page 392.

3 Specify a global default designating IPM.Note.Siebel as the name of the email form you use with Microsoft Outlook:

- **Siebel/Outlook Form.** For Siebel Web Client deployments, specify this parameter for the Application Object Manager.
- **OutlookForm.** For Siebel Mobile Web Client deployments, specify this parameter in the [EMail] section of the local Siebel client configuration file.

For more information, see "Email Client Parameters" on page 392.

4 Specify a global default specifying where email attachments will be temporarily stored:

- **Email Temporary Attachment Location.** For Siebel Web Client deployments, specify this as a component parameter for the Application Object Manager.
- **SiebelExtMailClientAttDir.** For Siebel Mobile Web Client deployments, specify this parameter in the [EMail] section of the local Siebel client configuration file.

For more information, see "Creating and Configuring the Attachments Directory" on page 384 and "Email Client Parameters" on page 392.

5 As appropriate for your deployment, end users may specify Siebel user preference settings for their email client. Users may set user preferences, for example, if you are supporting more than one email client option for Send Email.

Setting user preferences overrides the corresponding global defaults specified using the Default Email Client or Siebel/Outlook Form parameters, described in Table 78 on page 392. The global defaults are used if no user preference has been set.

Applicable user preferences in the Send Email section of the Outbound Communications options of the User Preferences screen include:

- **Email Client.** Set this to Microsoft Outlook.

- **Siebel/Outlook Form.** Set this to IPM.Note.Siebel.

For more information, see "Preference Settings for Outbound Communications" on page 306.

Email Client Parameters

The parameters described in Table 78 on page 392 specify settings applicable to your client-side email integration. As previously noted, you set these parameters *either* as server component parameters or as parameters in the [EMail] section of the configuration file, depending on your Siebel client type.

For more information about user preferences mentioned here, see "Preference Settings for Outbound Communications" on page 306.

Table 78. Email Client Parameters

Component Parameter for Application Object Manager	Parameter in [EMail] Section of Configuration File	Description
Default Email Client	DefaultMailClient	Set to one of the following: ■ Siebel Mail Client ■ Lotus Notes ■ Microsoft Outlook Corresponds to the Outbound Communications user preference Email Client. The parameter setting is in effect unless it is overridden by this user preference. If the value is either Lotus Notes or Microsoft Outlook, then the name of the applicable email form must also be specified, using other parameters. The default value is Siebel Mail Client.

Table 78. Email Client Parameters

Component Parameter for Application Object Manager	Parameter in [EMail] Section of Configuration File	Description
Email Client Debug Level	DebugLevel	Specifies the level of debugging information to be logged when Lotus Notes or Microsoft Outlook is used for Send Email. Set it to one of the following: ■ 0 – No logging will occur for the email client. ■ 1 – Logging will occur for ActiveX and JavaScript (Siebel eScript) associated with the email client. Debugging information is logged to the file siebelmail.log on each user's desktop. The siebelmail.log file tracks the execution of scripts associated with the Notes or Outlook form. When this parameter is set to 1, email client debugging information is logged as follows: ■ For Microsoft Outlook, the file siebelmail.log is overwritten for each new email message that is sent. Additional debugging data is displayed as messages in the user's email mail, with a subject line in the form SiebelDebugLog_*activity_ID*. ■ For Lotus Notes, additional debugging data is logged to the file SiebelEmaildbg.log on in the Lotus Notes directory. **NOTE:** Set this parameter to 1 for a development or test environment only. For a production environment, set it to 0.

Table 78. Email Client Parameters

Component Parameter for Application Object Manager	Parameter in [EMail] Section of Configuration File	Description
Email Temporary Attachment Location	SiebelExtMailClient AttDir	Specifies the name of the directory to be used to temporarily store email attachments. Specify the name in the following general form (entered on one line): *clientpath*;*serverpath* where *clientpath* identifies the server machine and path to the attachments directory, as accessed from client machines, and *serverpath* identifies the *same directory* as accessed from the server itself: ■ If the server machine is running Microsoft Windows, then the two elements defined here must be identical. For example: *servername*\attachment; *servername*\attachment ■ If the server machine is running on a supported UNIX platform, then the left element represents how Windows clients access the directory, while the right element represents the path on the UNIX server machine itself. For example: \\aixhost\attachment; /export/user/attachment ■ For Siebel Mobile Web Client deployments on Microsoft Windows platforms, a local attachments directory is required if users will run the Siebel application while disconnected from the network. The two elements defined here must be identical. For example: C:\temp;C:\temp For more information, see "Creating and Configuring the Attachments Directory" on page 384.

Table 78. Email Client Parameters

Component Parameter for Application Object Manager	Parameter in [EMail] Section of Configuration File	Description
Siebel/Lotus Form	LotusForm	Specifies the name of the email form to use as a global default, when Lotus Notes is used as the email client. Typically, set this parameter to Siebel Memo. Corresponds to the Outbound Communications user preference Siebel/Lotus Form. The parameter setting is in effect unless it is overridden by this user preference. For more information, see "About Installing the Siebel Email Form" on page 385 and subsequent sections applicable to Lotus Notes.
Siebel/Outlook Form	OutlookForm	Specifies the name of the email form to use as a global default, when Microsoft Outlook is used as the email client. Typically, set this parameter to IPM.Note.Siebel. Corresponds to the Outbound Communications user preference Siebel/Outlook Form. The parameter setting is in effect unless it is overridden by this user preference. For more information, see "About Installing the Siebel Email Form" on page 385 and subsequent sections applicable to Microsoft Outlook.

Other Communications Drivers

This section describes the settings and parameters for the following communications drivers provided by Siebel Systems:

- The Push Keep Alive communications driver provides a heartbeat message that is useful in maintaining connections for your communications sessions in some environments.

  For more information, see "Using Push Keep Alive Driver for Session Connections" on page 209.

- The Modem-Based TAP Paging communications driver is a feature of Siebel Paging.

  NOTE: Siebel Paging is included with all base applications of the Siebel Business Applications.

- The FTP communications driver is used by Siebel Marketing to send contact lists to vendors who are contracted to execute campaigns. It can also be used to send outbound communication requests from any Siebel application. For more information about Siebel Marketing, see *Siebel Marketing Installation and Administration Guide*.

For more information about communications drivers, see Chapter 3, "Configuring Communications Drivers and Profiles."

Push Keep Alive Driver Settings

Table 79 on page 396 lists the settings for the driver record for the Push Keep Alive driver. These settings appear in the Communications Drivers and Profiles view in the Administration - Communications screen.

CAUTION: The values shown in Table 79 on page 396 are presented for reference only. Do not modify the predefined values for this driver.

Table 79. Push Keep Alive Driver Settings

Field Name	Value
Name	Push Keep Alive
Channel Type	Push Keep Alive
Inbound	
Outbound	
Interactive	Selected by default
Channel String	Push Keep Alive
Library Name	sscmimed

Push Keep Alive Driver Parameters

Table 80 lists the supported parameters for the Push Keep Alive communications driver.

Table 80. Push Keep Alive Driver Parameters

Name	Required	Default Value	Description
LogDebug	No	FALSE	TRUE or FALSE If TRUE, data output to the log file is more detailed.
LogFile	No	push_{@UserName}.log	The name of the file where logging information will be written. If this parameter is empty, no log file is generated. Logging data is always appended to this file. Log files are created in the log subdirectory of the Siebel Server installation directory, depending on the client type and on your communications deployment. Optionally, if you want to use a single log file in a multichannel environment, you can specify the same log file name for all interactive drivers you are using.
MaxLogKB	No	128	Specifies the maximum size of the log file, in kilobytes (KB). When the log file is full, it is emptied completely, then logging begins again.

Table 80. Push Keep Alive Driver Parameters

Name	Required	Default Value	Description
PushKeep AliveTimer	Yes	0	Specifies the interval, in seconds, for the heartbeat message sent by the Keep Push Alive driver to the Application Object Manager and to each agent's Web browser. For example, if this parameter is set to 180, then the heartbeat message is sent every 180 seconds. The default value, 0, specifies that no heartbeat message is sent. Typically, this parameter is defined as a parameter override in a driver profile that has been associated with the communications configuration.
ReleaseLog Handle	No	TRUE	Indicates that the log file handle for each agent is released periodically, after logs have been generated. The default setting of TRUE provides better performance. When ReleaseLogHandle is FALSE, however, some log files may not be generated if the number of concurrent users exceeds the file-handle capacity provided by the operating system.

Modem-Based TAP Paging Driver Settings

Table 81 on page 398 lists the settings for the driver record for the Modem-Based TAP Paging driver. These settings appear in the Communications Drivers and Profiles view in the Administration - Communications screen.

CAUTION: The values shown in Table 81 on page 398 are presented for reference only. Do not modify the predefined values for this driver.

Table 81. Modem-Based TAP Paging Driver Settings

Field Name	Value
Name	Modem-Based TAP Paging
Channel Type	Page
Inbound	

Table 81. Modem-Based TAP Paging Driver Settings

Field Name	Value
Outbound	Selected by default
Interactive	
Channel String	TAP
Library Name	sscmtapm

Modem-Based TAP Paging Driver Parameters

Table 82 on page 399 lists the supported parameters for the Modem-Based TAP Paging communications driver. This driver supports the TAP (Telocator Alphanumeric Paging) protocol.

Error messages received when using CommOutboundMgr with the Modem-Based TAP Paging communications driver may include:

- `Unable to connect to the paging host system. The phone line may be busy or the phone number may be invalid.`

  Check the paging phone number the modem is dialing, including area code. Check if "9" is included, as may be required.

- `Unable to open the communication port.`

  Verify the communications settings for your system.

- `Error communication profile` *`profile_name`* `for CommSrvr paging.`

  An error has occurred using a specific profile (shown here as *profile_name*) for the communications driver Modem-Based TAP Paging. Verify that driver and profile parameter values are defined appropriately for your system.

Table 82. Modem-Based TAP Paging Driver Parameters

Name	Required	Default Value	Description
Dial Prefix	No	9	The prefix for outbound dialing.
Local Area Code	No		The area code of the phone line used for sending pages. There is no default value for this parameter, but a value must be specified at the customer site in order to dial local numbers successfully.

Table 82. Modem-Based TAP Paging Driver Parameters

Name	Required	Default Value	Description
LogDebug	No	FALSE	TRUE or FALSE If TRUE, data output to the log file is more detailed. Log files are created in the log subdirectory of the Siebel Server installation directory.
Long Distance Prefix	No	1	The long-distance prefix to add to long-distance calls.
Modem Dial String	No	ATDT	The modem command for dialing a phone number.
Modem Hangup String	No	ATH	The modem command for hanging up.
Modem Init String	No	AT&FQ0V1	The modem command for initializing the modem. The default value may need to be changed to work with different modems. For example, some modems expect a numeric value such as "AT&F1Q0V1" after the "&F" string. Consult your modem's manual for valid modem commands and values. You may need to add other commands to initialize your modem properly. For example, you may need to turn off any compression or speed-enhancing features on your modem. Consult your modem's manual for more information.
Modem Port	No	COM1	The computer's communications port to which the modem is attached. The exact value for Modem Port is dependent on the platform and on the specific devices. For example, for Solaris, you can use: /dev/cua/a For AIX, you can use: /dev/tty1
Modem Reset String	No	ATZ	The modem command for resetting the modem.

Table 82. Modem-Based TAP Paging Driver Parameters

Name	Required	Default Value	Description
Modem Restore String	No	AT&F	The modem command for restoring the default settings of the modem.
Siebel Server	No		The name of the Siebel Server that is to handle the delivery of the communications. For more information, see "Specifying the Siebel Server for Communications Outbound Manager" on page 264.

FTP Driver Settings

Table 83 on page 401 lists the settings for the driver record for the FTP driver. These settings appear in the Communications Drivers and Profiles view in the Administration - Communications screen.

CAUTION: The values shown in Table 83 on page 401 are presented for reference only. Do not modify the predefined values for this driver.

Table 83. FTP Driver Settings

Field Name	Value
Name	FTP
Channel Type	FTP
Inbound	
Outbound	Selected by default
Interactive	
Channel String	FTP
Library Name	sscmftp

FTP Driver Parameters

Table 84 on page 402 lists the supported parameters for the FTP communications driver. This driver supports FTP (File Transfer Protocol). Note the following requirements for sending files using the FTP driver:

- The name of the FTP server machine (such as a vendor's FTP server) to which the files will be sent using FTP must be identified in the hosts file (or equivalent) on the Siebel Server machine.

- The user specified in the profile for the FTP driver, by using the Username and Password parameters, must be defined as a user locally on the FTP server to which the files will be sent using FTP.

Table 84. FTP Driver Parameters

Name	Required	Default Value	Description
Filename	No		The name of a specific file that is to be uploaded every time the profile for this driver is used.
Hostname	No		The host name of the machine to which the FTP connection will be established. **NOTE:** Either Host name *or* IP Address must be defined for any profile created for this driver.
IP Address	No		The IP address of the machine to which the FTP connection will be established. **NOTE:** Either Host name *or* IP Address must be defined for any profile created for this driver.
LogDebug	No	FALSE	TRUE or FALSE If TRUE, data output to the log file is more detailed. Log files are created in the log subdirectory of the Siebel Server installation directory.
Password	Yes		The password for the user account that is accessing the FTP server.
Port	No	21	The port used by the FTP server.
Siebel Server	No		The name of the Siebel Server that is to handle the delivery of the communications. For more information, see "Specifying the Siebel Server for Communications Outbound Manager" on page 264.
Username	Yes		The login name for the account that is accessing the FTP server.

A Developing a Communications Driver

This appendix provides developer guidelines for writing communications drivers using the Siebel Adaptive Communications application programming interface. It includes the following topics:

Required Skills for Adaptive Communications Developer

To develop an interface to Adaptive Communications, take the time to learn about the following:

- Siebel Business Applications—from an end-user and development standpoint
- Communications concepts and event-handling model
- Technologies and development tools for the languages C++ or C

NOTE: Siebel Professional Services may be engaged to provide guidance and support in the development of custom communications drivers using Adaptive Communications. Custom communications drivers are to be supported and maintained by their originators. Siebel Systems is not liable in any way for custom communications drivers.

Custom Driver Upgrade Issues

This section describes issues that may apply if you are supporting or updating custom communications drivers originally written for a Siebel Business Applications release 7.x that is earlier than version 7.7; that is, for drivers written for version 7.0.*x* or version 7.5.*x*.

Communications drivers for prior 7.*x* releases will not work *as is* with version 7.7.*x*, unless you follow the guidelines in this section. Version 7.7.*x* requires version 2 of the Siebel Adaptive Communications API (as did version 7.5.*x*).

Requirements that *may* apply for your custom communications driver are as follows:

- On all supported platforms, for any communications driver written for version 7.0.*x*, you *must* compile the driver using the current version of the API. A version 7.0.*x* communications driver cannot work with version 7.7.*x* of the Siebel Business Applications, due to version checking. For drivers from version 7.5.*x*, this requirement will already have been met.

- For drivers on all supported platforms, the driver handle method FreeSCStrParamList *must* be implemented. For any driver for version 7.0.*x* or 7.5.*x*, if you previously did not implement this method to function as described, then you must implement it, and recompile your driver. If you previously implemented this method, then this requirement may not apply. For more information about this method, see "Methods of ISC_DRIVER_HANDLE" on page 418.
- For Sun Solaris platforms, you *must* use UTF16, in order to support Unicode, rather than UCS4 (as was used in version 7.0.*x*). For any driver on this platform for version 7.0.x, you must modify your driver code, and recompile your driver. For drivers from version 7.5.*x*, this requirement will already have been met.

Adaptive Communications Design

Siebel Adaptive Communications is a programmable software layer between Siebel applications and the external communications system that a communications driver is written to support, such as a CTI middleware package or email server.

The Adaptive Communications layer has two main parts:

- The client handle, which is how the communications driver can access functionality in the Siebel application
- The communications driver (encompassing the driver handle and the service handle), through which communications pass between the Siebel application and the external communications system

Communications Drivers

Communications drivers can be provided by Siebel Systems or developed by others:

- Siebel Systems provides communications drivers for use with Intel NetMerge (formerly Dialogic CT Connect), standard email servers that support SMTP and POP3, and other systems.

  For more information about configuring communications drivers, see Chapter 3, "Configuring Communications Drivers and Profiles".

  For driver parameter, command, and event details for each driver provided by Siebel Systems, see Chapter 12, "Using Siebel CTI Connect" and Chapter 13, "Using Email, Fax, and Other Systems."

  For information about third-party product support, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

- Siebel customers and vendors of CTI middleware, email servers, and so on, can use Adaptive Communications to write custom drivers to support other communications systems.

  To find out about communications drivers created by other vendors, refer to the Alliances page on the Siebel Web site, http://www.siebel.com.

Drivers can be created that will run on Microsoft Windows or on UNIX platforms. For example, drivers on Microsoft Windows may be Dynamic Link Libraries (DLL files), or drivers on UNIX platforms may be shared object (.so) files. Alternatively, drivers may be created as other types of files than these, such as executable files.

Adaptive Communications Architecture

The communications flow among the elements of Adaptive Communications is shown in Figure 3. As shown, the implementations of the driver handle and the service handle together make up the communications driver, along with any other program data and logic built into the driver.

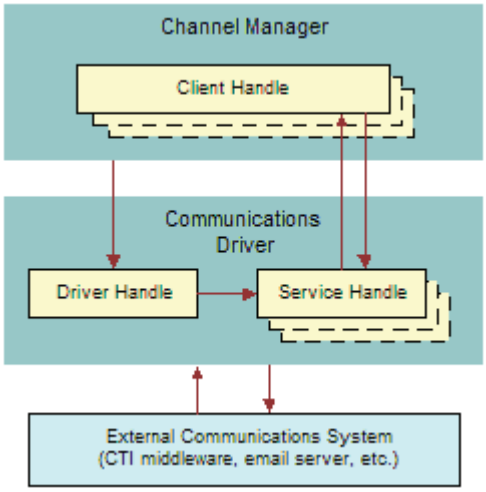

Figure 3. Adaptive Communications Architecture

The channel manager initializes the implementation object for the driver handle, which in turn initializes the implementation object for the service handle.

Multiple instances of the client handle and service handle implementation objects may be initialized, such as for each interactive agent session, depending on the driver implementation and on the client deployment.

For an overview of the entire communications system, see "About Communications Server Architecture" on page 20.

Adaptive Communications Event and Command Model

Siebel Adaptive Communications, when it is used to write interactive drivers such as those used with CTI middleware, uses the concept of events and commands:

- A *command* is a feature of the communications driver that can execute a particular function, such as on an external system like a telephone switch.

  Usually, the command, such as a request from the Siebel client to make a call or send a message, is executed with parameters whose values contain associated data, such as the number to call. The communications driver passes the command and associated data to the communications system for execution.

- An *event* is a notification of a communications occurrence that the Siebel client receives from the communications driver along with some data fields—such as notification of an incoming work item, with associated data such as a caller's telephone number (ANI).

For more information about interactive drivers, see "Interactive Drivers" on page 41.

Once you have written a communications driver, you configure Siebel Communications Server, as described in Chapter 4, "Configuring Session Communications," and Chapter 5, "Configuring Events and Commands."

Events and commands you define in the views in the Administration - Communications screen must be based on what the interactive communications driver supports. Commands, command parameters, events, and event data fields will vary among communications drivers, just as they vary among vendors of communications systems, such as CTI middleware vendors.

You can retrieve driver parameter values from the Siebel Database. Storing the parameters you need in the Siebel Database allows the administrator to set parameter values appropriately for your communications driver.

You can write a communications driver to support the driver parameters you require. If you have created a custom driver to function in place of a Siebel-provided driver, you must then create all the driver parameters in the views in the Administration - Communications screen and assign values to them. If you have created a custom driver as an aggregate driver, to extend a Siebel-provided driver, then you can use parameters defined by Siebel Systems.

The communications driver and the Siebel system software, such as Siebel business services and server components, work together to implement the desired communications behavior for the Siebel application.

Initialization of Communications Drivers

Driver initialization may involve initializing two handles for each communications driver implementation: ISC_DRIVER_HANDLE (the driver handle) and ISC_SERVICE_HANDLE (the service handle).

For drivers loaded on a Siebel Server machine, the driver handle is typically initialized only once, rather than once for each agent session. This depends on your driver implementation.

The driver handle and service handle can be implemented in any appropriate language, but must be implemented together using the same language.

Loading Communications Drivers

The process of loading a communications driver is as follows:

1. The channel manager invokes the driver handle method APIVersion, to check the driver's API version. The API version defined in the driver's header file must match the API version for Communications Server. See "APIVersion" on page 418.
2. The channel manager invokes the driver handle method CreateISCDriverInstance, to create the driver instance. The channel string passed to this method is collected from the driver record that is created in the Communications Drivers and Profiles view, in the Administration - Communications screen. See "CreateISCDriverInstance" on page 418.
3. The channel manager invokes the driver handle method RequestCommandEventList, to retrieve the list of events and commands this driver supports. See "RequestCommandEventList" on page 419.
4. The channel manager invokes the driver handle method RequestService, to create a service object. See "RequestService" on page 419.

If any step listed above fails, the driver loading fails. If multiple driver profiles must be loaded, the above process is repeated for each profile.

ISC_DRIVER_HANDLE

During initialization of an interactive communications driver, the CreateISCDriverInstance method is called to create the driver handle.

This method uses driver and profile parameter data (parameters prefaced with "Driver:" *or* parameters without a prefacing keyword) from the communications configuration, specified in the Communications Drivers and Profiles view.

For more information about the CreateISCDriverInstance method, see "CreateISCDriverInstance" on page 418.

ISC_SERVICE_HANDLE

After the driver handle has been created, the driver handle method RequestService is invoked, to request the driver handle to provide a service. Driver parameters (parameters prefaced with "Service:" *or* parameters without a prefacing keyword) are passed, which were specified in the Communications Drivers and Profiles view.

In response to this request, the driver handle returns the service handle (ISC_SERVICE_HANDLE), one instance for each agent. Calling methods of the service handle makes the feature set of the communications system available.

The driver handle and service handle together make up the implementation of the communications driver, along with other program data and logic.

Your implementation of the service handle invokes methods of the client handle, ISC_CLIENT_HANDLE. The client handle is implemented by Siebel Systems.

For more information about the RequestService method, see "RequestService" on page 419.

Driver Parameters and Initialization

For the drivers provided by Siebel Systems, each driver parameter may be prefaced with a keyword indicating how it will be used. Driver parameters defined for custom Adaptive Communications drivers can also follow this scheme for controlling how parameters are used:

- Parameters prefixed with "Driver:" are sent (with the prefix "Driver:" removed) to the driver handle when the driver is being initialized. These parameters are sent to the CreateISCDriverInstance method. (This applies to the Siebel CTI Connect driver provided by Siebel Systems.)
- Parameters prefaced with "Service:" are sent (with the prefix "Service:" removed) to the driver handle when it requests a service (creates the service handle). These parameters are sent to the RequestService method. (This applies to the Siebel CTI Connect driver provided by Siebel Systems.)
- Parameters that are not prefixed with either "Driver:" or "Service:" (as is the case for most of the drivers provided by Siebel Systems) are sent to the driver handle when the driver is being initialized (CreateISCDriverInstance method), and also when the driver handle requests a service (RequestService method).

Driver Event Attributes

Table 85 on page 408 lists event attributes that are included by the channel manager for all events processed by the client handle method HandleEvent. These attributes apply to any communications driver that invokes the client handle method HandleEvent—whether for drivers provided by Siebel Systems or custom drivers developed by customers or by other vendors.

You can use these attributes with event handler definitions in communications configurations. For more information, see "Event Handlers" on page 104. See also "Work Item Attributes" on page 190.

The event attributes described in Table 85 are also available for workflows used in processing inbound communications.

For more information about the HandleEvent client handle method, see "HandleEvent" on page 416.

Table 85. Driver Event Attributes

Attribute Name	Description
SiebelChannelProfile	The name of the applicable profile for the driver.
SiebelChannelType	The language-independent value of the channel type.
SiebelChannelTypeString	The channel string defined for the driver.
SiebelDriverEventName	The device event name as known by the driver.

Table 85. Driver Event Attributes

Attribute Name	Description
SiebelDriverNotifyWhenDone	The setting of whether the driver requests notification from the client when event-handling is done. The client handle method HandleEvent includes the parameter notifyWhenDone, which is used to request the client to notify the driver when the client has finished handling the event. The client notifies the driver that event handling is done by using the service handle method NotifyEventHandlingFinished.
SiebelWorkItemID	The unique work item ID, a value in the form *profile_ID##driver_tracking_ID*, where *profile_ID* is the ID of the driver profile and *driver_tracking_ID* is the work item's tracking ID, as known by the driver. For example, if the ID of the driver profile is 12-01234, and the driver uses the tracking ID "abcdefg" for this work item, then "12-01234##abcdefg" is the value of SiebelWorkItemID for this work item.

For events received for each interactive driver, additional data is received as event data fields. For more information about the events and event data fields for interactive drivers provided by Siebel Systems, see Chapter 12, "Using Siebel CTI Connect."

Siebel Adaptive Communications API Reference

This section provides a reference for the methods supported by the Adaptive Communications application programming interface. Siebel Adaptive Communications is a programmable software layer between Siebel applications and the communications system.

For more information for developing a communications driver using Siebel Adaptive Communications, see "Adaptive Communications Design" on page 404.

This section describes the client, driver, and service handles and the constants used by Adaptive Communications; and describes the methods for these handles.

Handles for Adaptive Communications

The Adaptive Communications layer includes a set of constants and the three handles described in the following paragraphs. The syntax with which each Adaptive Communications method is expressed is that of a C header file. In the following handle names, ISC stands for Interface for Siebel Communications:

- **ISC_CLIENT_HANDLE.** The client handle, which provides methods called by the communications driver. Siebel Systems has implemented ISC_CLIENT_HANDLE and its methods.

  For descriptions of the client handle methods, see "Methods of ISC_CLIENT_HANDLE" on page 414.

- **ISC_DRIVER_HANDLE.** The driver handle, which is created when communications are initialized (for example, when the first user logs in). ISC_DRIVER_HANDLE then creates ISC_SERVICE_HANDLE, from which services are requested. Customers writing communications drivers must implement methods of the driver handle.

  For descriptions of the driver handle methods, see "Methods of ISC_DRIVER_HANDLE" on page 418.

- **ISC_SERVICE_HANDLE.** The service handle, which is created by ISC_DRIVER_HANDLE when communications are initialized (for example, when a user logs in). The Siebel client calls ISC_SERVICE_HANDLE methods in order to perform actual communications functions. Customers writing communications drivers must implement methods of the service handle.

  For more information about these handles, see "Adaptive Communications Design" on page 404.

  For descriptions of the service handle methods, see "Methods of ISC_SERVICE_HANDLE" on page 420.

Constants for Communications Drivers

Each of the following elements enumerates one or more constants. Values for all fields are stored when information about objects is passed back and forth between the Siebel software and the communications driver.

For example, object information is passed in this manner when the client handle method UpdateObjectInformation is invoked. See "UpdateObjectInformation" on page 416.

SCCommandFlag

Communications commands have dynamic or runtime status, consisting of the bit-flags enumerated by SCCommandFlag. Zero or more bit-flags can apply to a given command at any given time.

NOTE: In developing a driver, specify the names of the status flags, such as SC_CF_DISABLED, SC_CF_BLINKING, and so on, rather than the integer values.

```
enum SCCommandFlag
{
    SC_CF_NOTSUPPORTED  = 1,
    SC_CF_DISABLED      = 2,
    SC_CF_CHECKED       = 4,
    SC_CF_BLINKING      = 8,
    SC_CF_NOPARAMSOK    = 16,
    SC_CF_STRPARAMSOK   = 32
};
```

Each bit-flag that can be associated with a command is described as follows:

- SC_CF_NOTSUPPORTED – The command is not supported.
- SC_CF_DISABLED – The command is supported, but is disabled at this time.
- SC_CF_CHECKED – The command is in a "checked" state—for example, when an agent is in Not Ready mode, the command to set the Not Ready state will be checked. This flag can be used to specify a toggle state for commands in the communications toolbar or in the Communications submenu of the Tools application-level menu.
- SC_CF_BLINKING – Blinking is enabled for communications toolbar buttons, such as those for the Accept Work Item and Resume Work Item commands.
- SC_CF_NOPARAMSOK – The command does not require any parameters to execute. For example, the commands for releasing a call and for toggling the Not Ready state do not require parameters.
- SC_CF_STRPARAMSOK – The command can be executed by providing a single unnamed string parameter (no named parameter is provided). The command for placing a call, for example, requires only a single parameter. Such commands are invoked when the user types something in the input field in the communications toolbar—such as an extension number—and clicks the command's button.

NOTE: If a device command's status excludes SC_CF_NOTSUPPORTED or SC_CF_DISABLED bit flags, then the command is considered enabled.

SCCommandTypeEx

Communications commands can be specified using either the InvokeCommand or InvokeCommandEx methods of the service handle. The values enumerated by SCCommandTypeEx specify the command type for an invocation of the InvokeCommandEx method. See "InvokeCommandEx" on page 421.

NOTE: In developing a driver, specify the names of the command types, such as SC_CT_MAKECALL, rather than the integer values.

```
enum SCCommandTypeEx
{
    SC_CT_MAKECALL        = 1,
    SC_CT_SENDMESSAGE     = 2
};
```

Each command type is described as follows:

- SC_CT_MAKECALL – The command to be invoked is a command for placing a phone call. The InvokeCommandEx command includes predefined parameters concerning the phone call to be made.
- SC_CT_SENDMESSAGE – The command to be invoked is a command to send an email message. The InvokeCommandEx command includes predefined parameters concerning the email message to be sent.

SCErrorCode

The values enumerated by SCErrorCode specify predefined error codes. The error codes from 0 to 1000 are reserved.

NOTE: In developing a driver, specify the names of the error codes, such as SC_EC_OK, SC_EC_ERROR, and so on, rather than the integer values.

```
enum SCErrorCode
{
    SC_EC_OK                        = 0,
    SC_EC_ERROR                     = 1,
    SC_EC_CMD_NOT_SUPPORTED         = 2,
    SC_EC_MEDIA_TYPE_NOT_SUPPORTED  = 3,
    SC_EC_INVALID_HANDLE            = 4,
    SC_EC_OUT_OF_MEMORY             = 5,
    SC_EC_NETWORK_ERROR             = 6,
    SC_EC_LIB_LOAD_ERR              = 7,
    SC_EC_FUNC_NOT_RESOLVED         = 8,
    SC_EC_DRIVER_CREATION_ERR       = 9,
    SC_EC_DRIVER_RELEASE_ERR        = 10,
    SC_EC_SERVICE_CREATION_ERR      = 11,
    SC_EC_SERVICE_RELEASE_ERR       = 12,
    SC_EC_INVALID_ITEM_TRACKING_ID  = 13,
    SC_EC_CLIENT_INTERFACE_ERR      = 14,
    SC_EC_SENDMSG_FAILED_RETRY      = 15,
    SC_EC_IMPOBJ_CREATE_ERR         = 16,
    SC_EC_INVALID_LICENSE           = 17,
    SC_EC_WORK_ITEM_WRONG_STATE     = 18,
    SC_EC_DRIVER_SPECIFIC           = 1000
};
```

Each error code is described as follows:

- SC_EC_OK – The operation is successful.
- SC_EC_ERROR – The operation failed.
- SC_EC_CMD_NOT_SUPPORTED – The command is not supported.
- SC_EC_MEDIA_TYPE_NOT_SUPPORTED – The channel type string is not recognized.
- SC_EC_INVALID_HANDLE – Invalid handle.
- SC_EC_OUT_OF_MEMORY – Out of memory.
- SC_EC_NETWORK_ERROR – Networking error.
- SC_EC_LIB_LOAD_ERR – Failure on loading the driver file.
- SC_EC_FUNC_NOT_RESOLVED – Failure on resolving function address while invoking the command.
- SC_EC_DRIVER_CREATION_ERR – Failure on creating the driver handle.
- SC_EC_DRIVER_RELEASE_ERR – Failure on releasing the driver handle.
- SC_EC_SERVICE_CREATION_ERR – Failure on creating the service handle.

- SC_EC_SERVICE_RELEASE_ERR – Failure on releasing the service handle.
- SC_EC_INVALID_ITEM_TRACKING_ID – Invalid tracking ID.
- SC_EC_CLIENT_INTERFACE_ERR – Failure on invoking the ISC_CLIENT interface.
- SC_EC_SENDMSG_FAILED_RETRY – SC_CT_SENDMESSAGE failed, please resend later.
- SC_EC_IMPOBJ_CREATE_ERR – Unable to create the underlying implementation object.
- SC_EC_INVALID_LICENSE – License error—used by the driver to report its license checking.
- SC_EC_WORK_ITEM_WRONG_STATE – The work item is in the wrong state for the operation.
- SC_EC_DRIVER_SPECIFIC – Specify for driver.

SCObjectProperty

Properties of monitored communications objects.

NOTE: In developing a driver, specify the names of the object properties, such as SC_OP_ONOFF, SC_OP_AGENTID, and so on, rather than the integer values.

```
enum SCObjectProperty
{
        SC_OP_ONOFF          = 1,
        SC_OP_AGENTID        = 2,
        SC_OP_ISNOTREADY     = 4,
        SC_OP_ISBUSY         = 5,
        SC_OP_DESCRIPTION    = 7,
        SC_OP_TIMEINQUEUE    = 9,
        SC_OP_QUEUEID        = 12,
        SC_OP_ISLOGON        = 13
};
```

Each object property is described as follows:

- SC_OP_ONOFF – A Boolean value indicating whether teleset control is on or off.
- SC_OP_AGENTID – The ID number of an agent, for an agent or DN that is being monitored.
- SC_OP_ISNOTREADY – A Boolean value indicating whether an agent has set the Not Ready state, for an agent or DN that is being monitored.
- SC_OP_ISBUSY – A Boolean value indicating whether an agent has set the Busy state, for an agent or DN that is being monitored.
- SC_OP_DESCRIPTION – A string describing the object that is being monitored.
- SC_OP_TIMEINQUEUE – The number of seconds during which the call stayed in the ACD queue before it was answered.
- SC_OP_QUEUEID – A string specifying the name of a queue. This is used together with SC_OP_ISLOGON to update the agent's login status for an ACD queue in the Communications options of the User Preferences screen.
- SC_OP_ISLOGON – A value indicating that the agent logged on or logged off.

SCWorkItemMode

Identifies work item modes, for inbound or outbound work items.

NOTE: In developing a driver, specify the names of the error codes, such as SC_WM_INBOUND, rather than the integer values.

```
enum SCWorkItemMode
{
      SC_WM_INBOUND  = 1,
      SC_WM_OUTBOUND = 2
};
```

Each work item type is described as follows:

- SC_WM_INBOUND – The work item is an inbound item.
- SC_WM_OUTBOUND – The work item is an outbound item, initiated by a user.

Data Types for Communications Drivers

The following are data types referenced in this appendix:

```
struct ISC_KeyValue       /* Key-Value element */
   {
      ISC_STRING    paramName;
      ISC_STRING    paramValue;
   };

struct ISC_KVParamList    /* List of Key-Value parameter */
   {
      struct ISC_KeyValue* dataItems;
      long                 len;
   };

struct ISC_StrParamList   /* List of String */
   {
      ISC_STRING*    dataItems;
      long           len;
   };

struct ISC_LongParamList  /* List of "long" */
   {
      long*          dataItems;
      long           len;
   };
```

Methods of ISC_CLIENT_HANDLE

The client handle, ISC_CLIENT_HANDLE, is implemented by Siebel Systems. The communications driver calls client handle methods in order to communicate with the Siebel application, such as to send communications events and associated data.

When a call to a client handle method is successful, it returns 0 (zero). ISC_RESULT represents the API result type defined for the Siebel Adaptive Communications API.

BeginBatch

Begins a set of client methods to be invoked in a batch when the corresponding EndBatch method is invoked. Using these methods can reduce network overhead. Developers can use these methods at their discretion, subject to the driver implementation design. See also "EndBatch" on page 415.

```
ISC_RESULT (*BeginBatch)
  /* in  */(ISC_CLIENT_HANDLE                handle);
```

CacheCommandInformation

Notify the Siebel client about command-status caching.

For the *i*th command in commandNames, the command description is at the *i*th position in commandDescriptions, and the command status is at the *i*th position in commandStatuses. The SCCommandFlags bit flags are used by commandStatuses.

```
ISC_RESULT (*CacheCommandInformation)
  /* in  */(ISC_CLIENT_HANDLE                handle,
  /* in  */ const struct ISC_StrParamList*   commandNames,
  /* in  */ const struct ISC_StrParamList*    commandDescriptions,
  /* in  */ const struct ISC_LongParamList* commandStatuses);
```

CleanAllWorkItem

Notifies the Siebel client that all work items for this service session have been removed.

```
ISC_RESULT (*CleanAllWorkItem)
   /* in  */(ISC_CLIENT_HANDLE      handle);
```

EndBatch

Ends a set of client methods to be invoked in a batch when this method is invoked, given the use of the corresponding BeginBatch method to begin the batch. See also "BeginBatch" on page 415.

```
ISC_RESULT (*BeginBatch)
  /* in  */(ISC_CLIENT_HANDLE                handle);
```

HandleError

Handle asynchronous errors:

- If clntCmdTrackID is not 0 (zero), then it contains the same value that was passed to the service handle method InvokeCommand that caused the error.
- If clntCmdTrackID is set to 0 (zero), the error occurred out of context or the error was associated with a failed call attempt—with which no call ID data is associated.

```
ISC_RESULT (*HandleError)
   /* in  */(ISC_CLIENT_HANDLE   handle,
   /* in  */ const ISC_STRING    clntCmdTrackID,
   /* in  */ const ISC_STRING    error);
```

HandleEvent

Handle the named event received from the communications driver, using the given fields. By calling this method, the communications driver notifies the Siebel client of a communications event, such as a call coming in on the monitored teleset.

If notifyWhenDone is set to TRUE, then the communications driver is notified when event handling is finished. The event's ID is passed (using the trackingID parameter) in a call to the service handle method NotifyEventHandlingFinished. See "NotifyEventHandlingFinished" on page 421.

```
ISC_RESULT (*HandleEvent)
   /* in  */(ISC_CLIENT_HANDLE              handle,
   /* in  */ const ISC_STRING               name,
   /* in  */ const struct ISC_KVParamList*  fields,
   /* in  */ ISC_BOOLEAN                    notifyWhenDone,
   /* in  */ const ISC_STRING               trackingID);
```

IndicateNewWorkItem

Indicate a new incoming call by bringing the Siebel application to the front. If the end user has set the Enable Sound option in the User Preferences screen (Communications options), the ringin.au or other user-specified sound file will play.

```
ISC_RESULT (*IndicateNewWorkItem)
   /* in  */(ISC_CLIENT_HANDLE    handle,
   /* in  */ const ISC_STRING     trackingID,
   /* in  */ const ISC_STRING     oldTrackingID,
   /* in  */ const ISC_STRING     description,
   /* in  */ enum SCWorkItemMode  workItemMode);
```

ShowStatusText

Display textual status information in the status line of the Siebel client.

```
ISC_RESULT (*ShowStatusText)
   /* in  */(ISC_CLIENT_HANDLE   handle,
   /* in  */ const ISC_STRING    text);
```

UpdateObjectInformation

Notify the Siebel application about status changes of communications objects—for example, that an agent has become busy. Notifications about DN status will be used to keep track of when calls end and new calls start. For the object properties that can be provided using datasetInfo, see "SCObjectProperty" on page 413.

```
ISC_RESULT (*UpdateObjectInformation)
   /* in  */(ISC_CLIENT_HANDLE              handle,
   /* in  */ const ISC_STRING               trackingID,
   /* in  */ const ISC_STRING               mediaTargetAddr,
   /* in  */ const struct ISC_KVParamList*  datasetInfo);
```

WorkItemReleased

After a work item has been released, such as a phone call disconnected, the driver calls this function.

```
ISC_RESULT (*WorkItemReleased)
   /* in  */(ISC_CLIENT_HANDLE     handle,
   /* in  */ const ISC_STRING      trackingID,
   /* in  */ time_t                stopTime);
```

WorkItemResumed

After a work item has been resumed, such as a phone call resumed from hold at the switch, the driver calls this function.

```
ISC_RESULT (*WorkItemResumed)
   /* in  */(ISC_CLIENT_HANDLE     handle,
   /* in  */ const ISC_STRING      trackingID);
```

WorkItemStarted

After a work item has been started, such as a phone call connected, the driver calls this function.

```
ISC_RESULT (*WorkItemStarted)
   /* in  */(ISC_CLIENT_HANDLE     handle,
   /* in  */ const ISC_STRING      trackingID,
   /* in  */ const ISC_STRING      oldTrackingID,
   /* in  */ const ISC_STRING      description,
   /* in  */ const ISC_STRING      mediaTargetAddr,
   /* in  */ time_t                startTime);
```

WorkItemSuspended

After a work item has been suspended, such as a phone call put on hold at the switch, the driver calls this function.

```
ISC_RESULT (*WorkItemSuspended)
   /* in  */(ISC_CLIENT_HANDLE     handle,
   /* in  */ const ISC_STRING      trackingID);
```

Methods of ISC_DRIVER_HANDLE

The driver handle, ISC_DRIVER_HANDLE, is implemented in the communications driver. Driver handle methods are invoked to request the driver to create a service handle, among other things. The service handle represents the functionality of the communications system with which you are integrating.

When a call to a driver handle method is successful, it returns 0 (zero). ISC_RESULT represents the API result type defined for the Siebel Adaptive Communications API.

APIVersion

Request the driver to return the API version (Siebel Communications API, or SCAPI) for which the driver is implemented. The driver should return the SCAPI_VERSION defined in the header file.

```
ISCAPI long           APIVersion();
```

CreateISCDriverInstance

Request the driver to create the driver handle ISC_DRIVER_HANDLE, and to perform other initialization tasks, such as to connect to the CTI middleware.

```
ISCAPI ISC_RESULT   CreateISCDriverInstance
   /* in  */(const ISC_STRING               mediaTypeStr,
   /* in  */ const ISC_STRING               languageCode,
   /* in  */ const ISC_STRING               connectString,
   /* in  */ const struct ISC_KVParamList*  datasetParams,
   /* out */ ISC_DRIVER_HANDLE*             handle);
```

FreeSCStrParamList

Releases memory that was initially allocated for the driver.

```
ISCAPI void   FreeSCStrParamList
   /* in  */(struct ISC_StrParamList        strList);
```

NOTE: Each driver *must* implement this function, in order to free its own memory. If this function is not implemented, the driver will not be loaded.

GetImplementationObject

Return a data structure for an implementation object, such as may be part of a communications driver implementation. This function may be useful for extending the functionality of an existing driver.

If your communications driver implementation aggregates another communications driver, such that you have an additional object that mediates communication with the driver handle and service handle, this method can return implementation objects such as the driver handle, service handle, or some other structure.

An aggregate communications driver implementation typically includes a Siebel-provided driver, such as the Siebel CTI Connect driver, and a separate driver called the aggregate communications driver that extends the main driver's functionality or replaces part of its functionality.

If you write an aggregate communications driver, it may be useful to invoke the aggregate driver for certain functions. The aggregate communications driver intercepts certain commands for its special operation and passes other commands on to the existing communications driver.

The aggregate driver must be added to the Communications Drivers and Profiles view, so that the driver can be properly initialized and used.

Writing an aggregate communications driver to work with an existing communications driver can save much time compared with writing a driver from scratch.

```
ISCAPI ISC_RESULT   GetImplementationObject
   /* in  */(ISC_HANDLE     key,
   /* out */ ISC_HANDLE*    impObj);
```

ReleaseISCDriverInstance

Release the driver handle that was created using CreateISCDriverInstance, release a CTI middleware connection, and so on.

```
ISCAPI ISC_RESULT   ReleaseISCDriverInstance
   /* in  */(ISC_DRIVER_HANDLE              handle);
```

RequestCommandEventList

Return the list of supported commands or events.

```
ISCAPI ISC_RESULT   RequestCommandEventList
   /* in  */(const ISC_STRING            mediaTypeStr,
   /* out */ struct ISC_StrParamList*    commandList,
   /* out */ struct ISC_StrParamList*    eventList);
```

RequestMediaTypeList

Return the supported channel type.

```
ISCAPI ISC_RESULT   RequestMediaTypeList
   /* out */(struct ISC_StrParamList*     mediaTypeList);
```

RequestService

Request the driver to create the service handle ISC_SERVICE_HANDLE, and pass the service type (such as teleset monitoring), user-defined parameters, and the pointer to the Siebel client handle. The driver handle creates the service handle, which in turn communicates with the Siebel application for each agent:

- For Siebel-provided communications drivers, all parameter data is passed to the communications driver in the datasetParams parameter.

- For a customer-implemented communications driver, additional data can be passed to the service handle using the paramString variable.

```
ISCAPI ISC_RESULT   RequestService
   /* in  */(ISC_DRIVER_HANDLE                  handle,
   /* in  */ const struct ISC_CLIENT_INTERFACE  clntInterface,
   /* in  */ const ISC_STRING                   connectString,
   /* in  */ const struct ISC_KVParamList*      datasetParams,
   /* out */ ISC_SERVICE_HANDLE*                serviceHandle);
```

Methods of ISC_SERVICE_HANDLE

The service handle, ISC_SERVICE_HANDLE, is implemented in the communications driver. Services are returned to the Siebel client in response to a call for the driver handle method RequestService. The service handle methods are called in order to communicate with the communications system—for example, to send commands and associated data. See "RequestService" on page 419.

When a call to a service handle method is successful, it returns 0 (zero). ISC_RESULT represents the API result type defined for the Siebel Adaptive Communications API.

When a call to a service method fails, the communications driver can call the client handle method HandleError to pass error data. See "HandleError" on page 415.

AcceptWorkItem

Accept a work item.

```
ISCAPI ISC_RESULT   AcceptWorkItem
   /* in  */(ISC_SERVICE_HANDLE       handle,
   /* in  */ const ISC_STRING         trackingID);
```

HandleQueuedWorkItem

Handle a queued work item.

```
ISCAPI ISC_RESULT   HandleQueuedWorkItem
   /* in  */(ISC_SERVICE_HANDLE             handle,
   /* in  */ const ISC_STRING               name,
   /* in  */ const struct ISC_KVParamList*  fields,
   /* in  */ const ISC_STRING               trackingID);
```

InvokeCommand

Invoke a command by name, using parameters. A basic ISC_SERVICE_HANDLE implementation should include at least the InvokeCommand method, in order to invoke named commands.

When this method is called, a command is invoked that is to be executed—for example, placing a call, transferring a call, or setting the Not Ready state. Most commands, but not all, are passed from the communications driver to the communications system.

```
ISCAPI ISC_RESULT   InvokeCommand
   /* in  */(ISC_SERVICE_HANDLE             handle,
   /* in  */ const ISC_STRING               clntCmdTrackID,
   /* in  */ const ISC_STRING               name,
   /* in  */ const ISC_STRING               stringParam,
   /* in  */ const struct ISC_KVParamList*  datasetParam);
```

InvokeCommandEx

Invoke a command by type, using parameters. The command type is represented by the value of the SCCommandTypeEx constant. You can use this method to invoke commands on an email server or CTI middleware server. The data structure ISC_KVParamList specifies the appropriate predefined parameters, according to the command type. For more information, see "SCCommandTypeEx" on page 411. See also the description for InvokeCommand.

```
ISCAPI ISC_RESULT   InvokeCommandEx
   /* in  */(ISC_SERVICE_HANDLE             handle,
   /* in  */ const ISC_STRING               clntCmdTrackID,
   /* in  */ enum SCCommandTypeEx           commandType,
   /* in  */ const struct ISC_KVParamList*  datasetParam);
```

NotifyEventHandlingFinished

Notify the communications driver that the Siebel client has finished handling the event that was sent by the communications driver to the Siebel client using a call for the client handle method HandleEvent. See "HandleEvent" on page 416.

```
ISCAPI ISC_RESULT   NotifyEventHandlingFinished
   /* in  */(ISC_SERVICE_HANDLE      handle,
   /* in  */ const ISC_STRING        trackingID,
   /* in  */ ISC_BOOLEAN             result);
```

ReleaseISCServiceInstance

Release the service handle that was created using RequestService, and free resources. No subsequent calls to the driver handle will be made. The driver should in turn make no further calls to the client handle.

NOTE: ReleaseISCServiceInstance is the last communication between the Communications Session Manager and the communications driver. After calling this function, Communications Session Manager will no longer call the driver, and the driver should no longer call back to Siebel—because the communications client and the client handle will no longer exist.

```
ISCAPI ISC_RESULT   ReleaseISCServiceInstance
   /* in  */(ISC_SERVICE_HANDLE    handle);
```

ReleaseWorkItem

Release a work item.

```
ISCAPI ISC_RESULT   ReleaseWorkItem
   /* in  */(ISC_SERVICE_HANDLE      handle,
   /* in  */ const ISC_STRING        trackingID);
```

ResumeWorkItem

Resume a work item.

```
ISCAPI ISC_RESULT   ResumeWorkItem
   /* in  */(ISC_SERVICE_HANDLE      handle,
   /* in  */ const ISC_STRING        trackingID);
```

RevokeQueuedWorkItem

Revoke a queued work item.

```
ISCAPI ISC_RESULT     RevokeQueuedWorkItem
   /* in  */(ISC_SERVICE_HANDLE      handle,
   /* in  */ const ISC_STRING        trackingID);
```

SuspendWorkItem

Suspend a work item.

```
ISCAPI ISC_RESULT   SuspendWorkItem
   /* in  */(ISC_SERVICE_HANDLE      handle,
   /* in  */ const ISC_STRING        trackingID);
```

Testing Communications Drivers

The Communications Driver Test Engine (Test Engine) is a standalone tool that simulates parts of a Siebel environment for you to test the reliability and capacity of the communications driver that you develop.

The Test Engine does not test the Communications Server or your communications configuration elements. You can use the Test Engine to test your communications driver when connected to your CTI middleware by specifying appropriate driver parameters in the definition file (.DEF) that you develop for testing your communications driver.

Before using the Test Engine, you configure settings for your communications driver. The Test Engine writes the results of the test to log files. You can specify the log file names by configuring the .DEF file. For example, using Siebel CTI Connect, you can specify log file names as follows:

- `Driver:DriverLogFile = "ctctestengine.log"`
- `Service:ServiceLogFile = "CTC_agent1.log"`

For additional information about communications drivers, see Chapter 3, "Configuring Communications Drivers and Profiles."

Table 86 describes the format that your .DEF must adhere to in order to simulate multiple calls. An example .DEF file follows the procedure below.

Table 86. Format of Definition File

This section ...	Must contain ...
Driver Parameter	Driver parameters to connect to the CTI middleware.
Agent#n Where n = the Agent's identifier	Parameters necessary to initialize agent login.
Job	Description of job to simulate.
Task	Description of tasks that comprise the job. Enter the device commands here that a communications driver supports to make sure that the commands and the communications driver function correctly.

The following procedure describes how you run the Test Engine tool to test a communications driver.

To test a communications driver

1. Copy your .DEF file into the BIN subdirectory of your Siebel Server or Siebel client installation directory.
2. From a command line, navigate to the BIN directory identified in Step 1.
3. At the command prompt, execute the following command:

   On Windows:

   `CommDriverTestEngine.exe` *`definition_file language_code`*

   On UNIX:

   `CommDriverTestEngine` *`definition_file language_code`*

   For example, on Windows:

   `CommDriverTestEngine.exe anExampleDefinitionFile.def ENU`

   The Test Engine tests your communications driver and writes the output to the specified log files.

Sample Definition File

The following sample definition file tests the communications driver that Siebel Systems provides with the Siebel CTI Connect module. It is a limited example that demonstrates some of the entries that can appear in a definition file. This example tests the following scenario:

- An agent (`Agent#1`) performs one job (`Job_MakeCallReleaseCall`) that loops four times.
- This job includes the following tasks:
  - Call extension number 56016 (`Task_MakeCall`).

- Wait for two seconds (Task_Wait2sec).
- Release the call (Task_ReleaseCall).
- Wait for five seconds (Task_Wait5sec).

The content of the .DEF file is as follows:

```
[Driver Parameter]
   Driver = "Dialogic CTI"
   Driver:LogicalID = "V7CTCTSLINK"
   Driver:CIMServer = "EGHTSGPW05"
   Driver:CTCServer = "EGHTSGPW05"
   Driver:NetworkType = "ncacn_ip_tcp"
   Driver:SwitchType = "0"
   Channel Type = "Voice"
   Channel String = "CTC Phone"
   Library Name = "sscmctc"
   LogDebug = "TRUE"
   Driver:DriverLogFile = "ctc.log"
[Agent#1]
   LogFile = "testengine.log"
   StartDelay = "2"
   Elapse = ""
   Service:ServiceLogFile = "CTC_agent1.log"
   Service:DNList = "54615"
   LogDebug = "TRUE"
   Job1 = "Job_MakeCallReleaseCall"
[Job:Job_MakeCallReleaseCall]
   Loop = "4"
   Task1 = "Task_MakeCall"
   Task2 = "Task_Wait2Sec"
   Task3 = "Task_ReleaseCall"
   Task4 = "Task_Wait5Sec"
```

```
[Task:Task_MakeCall]
   DeviceCommand = "MakeCall"
   PhoneNumber = "56016"
   CallNotifyText = "Call from Siebel..."
[Task:Task_Wait2Sec]
   Wait = 2
[Task:Task_Wait5Sec]
   Wait = 5
[Task:Task_ReleaseCall]
   DeviceCommand = "ReleaseCall"
```

B Communications Server Business Services

This appendix provides information about business services for Siebel Communications Server, including applicable business services, methods, and arguments. It includes the following topics:

About Business Services for Communications Server

The following are the business services of Siebel Communications Server. The names shown are the display names for these business services:

- **Communications Client.** Supports the communications user interface features such as the communications toolbar and communications menu commands. This business service aggregates the Communications Session Manager business service. It does not deal with server-based communications functionality.
- **Communications Session Manager.** Provides an interface to session-based communications functionality at the server level. Through Server Request Broker and Server Request Processor, this business service communicates with the Communications Session Manager server component. It does not deal with the communications user interface.
- **Outbound Communications Manager.** Provides an interface to the outbound communications functionality of the Communications Outbound Manager server component.

For more information about integrating business services with Communications Server, see "Using Business Services with Communications Server" on page 216.

Communications Client Methods

This section describes the methods and method arguments of the business service Communications Client. For more information about the role of business services and methods in communications toolbar configuration, see "About Communications Toolbar Configuration" on page 151.

The Communications Client business service is an aggregate business service that is built on top of the Communications Session Manager service. This means that you can invoke all the methods of the Communications Session Manager service as though they were methods of the Communications Client business service.

This section primarily uses the display names for each method (Method Display Name). The internal names, which should be used in any scripts that invoke these methods, are also shown (Method Name).

Table 87 lists the methods for the Communications Client business service.

Table 87. Communications Client Methods

Method Display Name	Method Name	Comment
Agent Sign Off	AgentSignOff	Execute logout command
Agent Sign On	AgentSignOn	Execute login command
Get Selected Work Item Info	GetSelectedWorkItemInfo	Retrieve complete work item information of the selected work item on toolbar
Handle Error	HandleError	Push error information to the agent's communications toolbar
Is Comm Enabled	IsCommEnabled	Are communications enabled?
Is Comm Simulated	IsCommSimulated	Are communications running in simulation mode?
Make Call	MakeCall	Make a phone call
Obtain UI Focus	ObtainUIFocus	Bring the browser to the front (to gain user interface focus)
Send Communication	SendCommunication	Send outbound communication item
Shell UI Update	ShellUIUpdate	Update or refresh the status of the communications toolbar
Show Status Text	ShowStatusText	Push error information to the agent's communications toolbar, and display message on the browser's status bar
Work Item Released	WorkItemReleased	Notification of work item released
Work Item Resumed	WorkItemResumed	Notification of work item resumed
Work Item Started	WorkItemStarted	Notification of work item started
Work Item Suspended	WorkItemSuspended	Notification of work item suspended

Arguments for Communications Client Methods

This section lists the arguments for each method of the business service Communications Client. Each argument's data type is also shown.

NOTE: In Siebel Tools, other arguments than these may also be listed. Disregard any arguments that are flagged as Hidden or Inactive.

Arguments for Agent Sign Off Method

The Agent Sign Off method has no arguments.

Arguments for Agent Sign On Method

The Agent Sign On method has no arguments.

Arguments for Get Selected Work Item Info Method

Table 88 lists the arguments for the Get Selected Work Item Info method.

Table 88. Arguments for Get Selected Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Channel Type (type: String)	ChannelType	Output	Y	The language-independent channel type
Channel Type Locale (type: String)	ChannelTypeLocale	Output	Y	The language-dependent channel type locale
Description (type: String)	Description	Output	Y	Work item description
Driver Work Tracking ID (type: String)	DriverWorkTrackID	Output	Y	The tracking ID of this work item in driver scope
Icon File (type: String)	IconFile	Output	Y	Icon file of the driver profile that owns this work item
Is Active State (type: Number)	IsActiveState	Output	Y	Is this work item active?
Is Data Contained Inside (type: Number)	HasWorkData	Output	Y	Is data contained inside?
Is Inbound Item (type: Number)	IsInboundItem	Output	Y	Is this work item inbound work item?
Parent Work Item ID (type: String)	ParentWorkItemID	Output	Y	ID of parent work item
Profile ID (type: String)	ProfileID	Output	Y	ID of driver profile that owns this work item

Table 88. Arguments for Get Selected Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Profile Name (type: String)	ProfileName	Output	Y	Name of driver profile that owns this work item
Time In Queue (type: String)	TimeInQueue	Output	Y	Time this work item stayed in the queue
UQ Work Item ID (type: String)	UQWorkItemID	Output	Y	The original work item ID when received from UQ
View Bookmark (type: String)	ViewBookmark	Output	Y	View bookmark
Work Duration (type: String)	WorkDuration	Output	Y	Duration of work item
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID
Work Item Key (type: String)	WorkItemKey	Output	Y	Work item key
Work Item Not Exist (type: Number)	WorkItemNotExist	Output	Y	Work item does not exist
Work Object ID (type: String)	WorkObjectID	Input	Y	Object ID
Work Start Time (type: String)	WorkStartTime	Output	Y	Time work item started
Work State (type: String)	WorkState	Output	Y	Work item state
Work Tracking Object Business Component (type: String)	WorkTrackObjBusComp	Output	Y	Name of business component for after-work tracking
Work Tracking Object Business Object (type: String)	WorkTrackObjBusObj	Output	Y	Name of business object for after-work tracking

Arguments for Handle Error Method

Table 89 lists the arguments for the Handle Error method.

Table 89. Arguments for Handle Error Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Error (type: String)	Error	Input	Y	Error text

Arguments for Is Comm Enabled Method

Table 90 lists the arguments for the Is Comm Enabled method.

Table 90. Arguments for Is Comm Enabled Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Is Comm Enabled (type: Number)	IsCommEnabled	Output	Y	Indicate if communications toolbar is enabled. 1 means TRUE and 0 means FALSE.

Arguments for Is Comm Simulated Method

Table 91 lists the arguments for the Is Comm Simulated method.

Table 91. Arguments for Is Comm Simulated Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Is Comm Simulated (type: Number)	IsCommSimulated	Output	Y	Indicate if communications toolbar is in simulation mode. 1 means TRUE and 0 means FALSE.

Arguments for Make Call Method

Table 92 lists the arguments for the Make Call method.

Table 92. Arguments for Make Call Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Apply Dialing Rule (type: Number)	ApplyDialingRule	Input	N	If dialing rule needs to be applied
Phone Number (type: String)	PhoneNumber	Input	Y	Phone number
Profile Name (type: String)	ProfileName	Input	Y	Name of driver profile

Arguments for Obtain UI Focus Method

The Obtain UI Focus method has no arguments.

Arguments for Send Communication Method

Table 93 lists the arguments for the Send Communication method.

Table 93. Arguments for Send Communication Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Channel Type (type: String)	CommType	Input	Y	Supported values: Email Fax Page
Method Argument (type: String)	Method Argument	Input	Y	

Arguments for Shell UI Update Method

The Shell UI Update method has no arguments.

Arguments for Show Status Text Method

Table 94 lists the arguments for the Show Status Text method.

Table 94. Arguments for Show Status Text Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Text (type: String)	Text	Input	Y	The text that will be displayed at the browser's status bar

Arguments for Work Item Released Method

Table 95 lists the arguments for the Work Item Released method.

Table 95. Arguments for Work Item Released Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Stop Time (type: Number)	StopTime	Input	Y	Time that work item is released
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Work Item Resumed Method

Table 96 lists the arguments for the Work Item Resumed method.

Table 96. Arguments for Work Item Resumed Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Work Item Started Method

Table 97 lists the arguments for the Work Item Started method.

Table 97. Arguments for Work Item Started Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Channel Profile (type: String)	MediaProfile	Input	Y	Name of driver profile that owns this work item
Channel Type (type: String)	MediaType	Input	Y	Language-independent value of channel string
Description (type: String)	Description	Input	Y	Work item description
Object ID (type: String)	ObjectID	Input	Y	Object ID of work item
Old Work Item ID (type: String)	OldWorkItemID	Input	Y	Old work item ID
Start Time (type: Number)	StartTime	Input	Y	Time work item is started
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Work Item Suspended Method

Table 98 lists the arguments for the Work Item Suspended method.

Table 98. Arguments for Work Item Suspended Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Communications Session Manager Methods

This section describes the methods and method arguments of the business service Communications Session Manager.

This section primarily uses the display names for each method (Method Display Name). The internal names, which should be used in any scripts that invoke these methods, are also shown (Method Name).

Table 99 lists the methods for the Communications Session Manager business service.

Table 99. Communications Session Manager Methods

Method Display Name	Method Name	Comment
Accept Work Item	AcceptWorkItem	Accept inbound work item
Get Agent Extension	GetAgentExtension	Retrieve agent's active extension number
Get All Work Item ID	GetAllWorkItemID	Retrieve all work item IDs
Get Inbound Work Item Attributes	GetInboundWorkItemAttr	Retrieve inbound work item attribute
Get Inbound Work Item Info	GetInboundWorkItemInfo	Retrieve complete information of inbound work item
Get Top Active Work Item ID	GetTopActiveWorkItemID	Retrieve the ID of most recent active work item
Get Top Idle Work Item ID	GetTopIdleWorkItemID	Retrieve the ID of most recent idle work item
Get Work Item Attributes	GetWorkItemAttr	Retrieve work item attributes
Get Work Item Info	GetWorkItemInfo	Retrieve complete information of a work item
Get Work Item Track Info	GetWorkItemTrackInfo	Retrieve the tracking info of work item
Invoke Command	InvokeCommand	Invoke device command For more information, see related methods in "Siebel Adaptive Communications API Reference" on page 409.

Table 99. Communications Session Manager Methods

Method Display Name	Method Name	Comment
Invoke Extended Command	InvokeCommandEx	Invoke predefined device command For more information, see related methods in "Siebel Adaptive Communications API Reference" on page 409.
Notify Event Handling Finished	NotifyEventHandlingFinished	Notify driver that event handling is over
Release Work Item	ReleaseWorkItem	Release work item
Release Work Item by Activity ID	ReleaseWorkItemEx	Release work item by activity record ID
Resume Work Item	ResumeWorkItem	Resume work item
Set Work Item Attributes	SetWorkItemAttr	Set work item attributes
Suspend Work Item	SuspendWorkItem	Suspend work item
Work Item Released	WorkItemReleased	Notification of work item released
Work Item Resumed	WorkItemResumed	Notification of work item resumed
Work Item Started	WorkItemStarted	Notification of work item started
Work Item Suspended	WorkItemSuspended	Notification of work item suspended

Arguments for Communications Session Manager Methods

This section lists the arguments for each method of the business service Communications Session Manager. Each argument's data type is also shown.

NOTE: In Siebel Tools, other arguments than these may also be listed. Disregard any arguments that are flagged as Hidden or Inactive.

Arguments for Accept Work Item Method

Table 100 lists the arguments for the Accept Work Item method.

Table 100. Arguments for Accept Work Item Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Get Agent Extension Method

Table 101 lists the arguments for the Get Agent Extension method.

Table 101. Arguments for Get Agent Extension Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Agent Login (type: String)	AgentLogin	Input	Y	Agent's login name
Destination DN (type: String)	DestinationDN	Output	Y	Agent's active extension

Arguments for Get All Work Item ID Method

Table 102 lists the arguments for the Get All Work Item ID method.

Table 102. Arguments for Get All Work Item ID Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
ActiveWorkItem_N (type: String)	ActiveWorkItem_N	Output	Y	Attribute names are in ActiveWorkItem_? format, such as ActiveWorkItem_0, ActiveWorkItem_1
IdleWorkItem_N (type: String)	IdleWorkItem_N	Output	Y	Attribute names are in IdleWorkItem_? format, such as IdleWorkItem_0, IdleWorkItem_1

Arguments for Get Inbound Work Item Attributes Method

Table 103 lists the arguments for the Get Inbound Work Item Attributes method.

NOTE: The Attribute Name and Attribute Value arguments are available for use, though they are not listed in Siebel Tools as arguments for this method.

Table 103. Arguments for Get Inbound Work Item Attributes Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Attribute Name (type: String)	AttrName	Input	N	Name of the attribute
Attribute Value (type: String)	AttrValue	Output	N	Value for the attribute
Channel Type (type: String)	ChannelType	Input	Y	Name of language-independent channel type
Item Index (type: Number)	ItemIndex	Input	Y	Index of inbound work item

Arguments for Get Inbound Work Item Info Method

Table 104 lists the arguments for the Get Inbound Work Item Info method.

Table 104. Arguments for Get Inbound Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Channel Type (type: String)	ChannelType	Input	Y	The language-independent channel type
Channel Type Locale (type: String)	MediaTypeLocale	Output	Y	The language-dependent channel type locale
Description (type: String)	Description	Output	Y	Work item description
Driver Work Tracking ID (type: String)	DriverWorkTrackID	Output	Y	Work tracking ID in driver scope
Icon File (type: String)	IconFile	Output	Y	Icon file of driver profile that owns this work item

Table 104. Arguments for Get Inbound Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Is Active State (type: Number)	IsActiveState	Output	Y	Is work item active?
Is Data Contained Inside (type: Number)	HasWorkData	Output	N	Is data contained inside?
Is Inbound Item (type: Number)	IsInboundItem	Output	Y	Is inbound work item?
Item Index (type: Number)	ItemIndex	Input	Y	The index of work item
Parent Work Item ID (type: String)	ParentWorkItemID	Output	Y	ID of parent work item
Profile ID (type: String)	ProfileID	Output	Y	Row ID of driver profile
Profile Name (type: String)	ProfileName	Output	Y	Name of driver profile
Time In Queue (type: Number)	TimeInQueue	Output	Y	Time work item in queue
UQ Work Item ID (type: String)	UQWorkItemID	Output	Y	The original work item when received from UQ
View Bookmark (type: String)	ViewBookmark	Output	Y	View bookmark
Work Duration (type: String)	WorkDuration	Output	Y	Duration of work item
Work Item ID (type: String)	WorkItemID	Output	Y	Work item ID
Work Item Not Exist (type: Number)	WorkItemNotExist	Output	Y	If such work item exists
Work Object ID (type: String)	WorkObjectID	Output	Y	Work object ID

Table 104. Arguments for Get Inbound Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Start Time (type: String)	WorkStartTime	Output	Y	Time work item started
Work State (type: String)	WorkState	Output	Y	Work item state
Work Tracking Object Business Component (type: String)	WorkTrackObjBusComp	Output	Y	Name of business component for after-work tracking
Work Tracking Object Business Object (type: String)	WorkTrackObjBusObj	Output	Y	Name of business object for after-work tracking

Arguments for Get Top Active Work Item ID Method

Table 105 lists the arguments for the Get Top Active Work Item ID method.

Table 105. Arguments for Get Top Active Work Item ID Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Output	N	Work item ID

Arguments for Get Top Idle Work Item ID Method

Table 106 lists the arguments for the Get Top Idle Work Item ID method.

Table 106. Arguments for Get Top Idle Work Item ID Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Output	N	Work item ID

Arguments for Get Work Item Attributes Method

Table 107 lists the arguments for the Get Work Item Attributes method.

NOTE: The Attribute Name and Attribute Value arguments are available for use, though they are not listed in Siebel Tools as arguments for this method.

Table 107. Arguments for Get Work Item Attributes Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Attribute Name (type: String)	AttrName	Input	N	Name of the attribute
Attribute Value (type: String)	AttrValue	Output	N	Value for the attribute
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Get Work Item Info Method

Table 108 lists the arguments for the Get Work Item Info method.

Table 108. Arguments for Get Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Channel Type (type: String)	MediaType	Output	Y	The language-independent channel type
Channel Type Locale (type: String)	MediaTypeLocale	Output	Y	The language-dependent channel type locale
Description (type: String)	Description	Output	Y	Work item description
Driver Work Tracking ID (type: String)	DriverWorkTrackID	Output	Y	The tracking ID of this work item in driver scope
Icon File (type: String)	IconFile	Output	Y	Icon file of driver profile that owns this work item
Is Active State (type: String)	IsActiveState	Output	Y	Is this work item active?

Table 108. Arguments for Get Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Is Data Contained Inside (type: Number)	HasWorkData	Output	N	Is data contained inside?
Is Inbound Item (type: String)	IsInboundItem	Output	Y	Is this work item inbound work item?
Parent Work Item ID (type: String)	ParentWorkItemID	Output	Y	ID of parent work item
Profile ID (type: String)	ProfileID	Output	Y	ID of driver profile that owns this work item
Profile Name (type: String)	ProfileName	Output	Y	Name of driver profile which owns this work item
Time In Queue (type: String)	TimeInQueue	Output	Y	Time this work item stays in the queue
UQ Work Item ID (type: String)	UQWorkItemID	Output	Y	The original work item ID when received from UQ
View Bookmark (type: String)	ViewBookmark	Output	Y	View bookmark
Work Duration (type: String)	WorkDuration	Output	Y	Duration of work item
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID
Work Item Key (type: String)	WorkItemKey	Output	Y	Work item key
Work Item Not Exist (type: Number)	WorkItemNotExist	Output	N	Work item does not exist
Work Object ID (type: String)	WorkObjectID	Output	Y	Object ID
Work Start Time (type: String)	WorkStartTime	Output	Y	Time work item started

Table 108. Arguments for Get Work Item Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work State (type: String)	WorkState	Output	Y	Work item state
Work Tracking Object Business Component (type: String)	WorkTrackObjBusComp	Output	Y	Name of business component for after-work tracking
Work Tracking Object Business Object (type: String)	WorkTrackObjBusObj	Output	Y	Name of business object for after-work tracking

Arguments for Get Work Item Track Info Method

Table 109 lists the arguments for the Get Work Item Track Info method.

Table 109. Arguments for Get Work Item Track Info Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
After Work Tracking Object Business Component (type: String)	AfcBusComp	Output	Y	Name of business component for after-work tracking
After Work Tracking Object Business Object (type: String)	AfcBusObj	Output	Y	Name of business object for after-work tracking
After Work Tracking Object Row ID (type: String)	AfcRowID	Output	Y	Record ID for after-work tracking
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Invoke Command Method

Table 110 lists the arguments for the Invoke Command method.

Table 110. Arguments for Invoke Command Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Command Tracking ID (type: String)	CommandTrackingID	Input	Y	Command ID
Data Set (type: Hierarchy)	DataSet	Input	Y	Data parameters for driver
Device Command (type: String)	DeviceCommand	Input	Y	Command to be invoked in driver
Driver Profile ID (type: String)	DriverProfileID	Input	Y	Row ID of driver profile which will be invoked
Profile Name (type: String)	ProfileName	Input	Y	Name of driver profile which will be invoked
String Parameter (type: String)	StringParam	Input	Y	Single string data

Arguments for Invoke Extended Command Method

Table 111 lists the arguments for the Invoke Extended Command method.

Table 111. Arguments for Invoke Extended Command Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Command Tracking ID (type: String)	CommandTrackingID	Input	Y	Command ID
Command Type (type: Number)	CommandType	Input	Y	Command type
Data Set (type: Hierarchy)	DataSet	Input	Y	Data parameters for driver

Table 111. Arguments for Invoke Extended Command Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Driver Profile ID (type: String)	DriverProfileID	Input	Y	Row ID of driver profile which will be invoked
Profile Name (type: String)	ProfileName	Input	Y	Name of driver profile which will be invoked

Arguments for Notify Event Handling Finished Method

Table 112 lists the arguments for the Notify Event Handling Finished method.

Table 112. Arguments for Notify Event Handling Finished Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Result (type: Number)	Result	Input	Y	Result code
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Release Work Item Method

Table 113 lists the arguments for the Release Work Item method.

Table 113. Arguments for Release Work Item Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Release Work Item by Activity ID Method

Table 114 lists the arguments for the Release Work Item by Activity ID method.

Table 114. Arguments for Release Work Item By Activity ID Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Activity ID (type: String)	ActivityID	Input	Y	Row ID of activity record

Arguments for Resume Work Item Method

Table 115 lists the arguments for the Resume Work Item method.

Table 115. Arguments for Resume Work Item Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Set Work Item Attributes Method

Table 116 lists the arguments for the Set Work Item Attributes method.

NOTE: For this method, you can set values for one or more attributes for the work item, using key-value pairs rather than arguments with fixed names. (By contrast, the methods Get Inbound Work Item Attributes and Get Work Item Attributes use the Attribute Name and Attribute Value arguments to store the name of a single attribute to retrieve, and its value, respectively.)

Table 116. Arguments for Set Work Item Attributes Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Suspend Work Item Method

Table 117 lists the arguments for the Suspend Work Item method.

Table 117. Arguments for Suspend Work Item Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Work Item Released Method

Table 118 lists the arguments for the Work Item Released method.

Table 118. Arguments for Work Item Released Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Stop Time (type: Number)	StopTime	Input	Y	Time that work item is released
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Work Item Resumed Method

Table 119 lists the arguments for the Work Item Resumed method.

Table 119. Arguments for Work Item Resumed Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Work Item Started Method

Table 120 lists the arguments for the Work Item Started method.

Table 120. Arguments for Work Item Started Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Channel Profile (type: Number)	MediaProfile	Input	Y	Name of driver profile that owns this work item
Channel Type (type: Number)	MediaType	Input	Y	Language-independent value of channel string
Description (type: String)	Description	Input	Y	Work item description
Object ID (type: String)	ObjectID	Input	Y	Object ID of work item
Old Work Item ID (type: String)	OldWorkItemID	Input	Y	Old work item ID
Start Time (type: Number)	StartTime	Input	Y	Time work item is started
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Arguments for Work Item Suspended Method

Table 121 lists the arguments for the Work Item Suspended method.

Table 121. Arguments for Work Item Suspended Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Work Item ID (type: String)	WorkItemID	Input	Y	Work item ID

Outbound Communications Manager Methods

This section describes the methods and method arguments of the business service Outbound Communications Manager.

For more information about the Siebel application features for creating and submitting communication requests, see Chapter 10, "Defining Outbound Communication Requests."

This section primarily uses the display names for each method (Method Display Name). The internal names, which should be used in any scripts that invoke these methods, are also shown (Method Name).

All the methods use profiles for a communications driver such as Internet SMTP/POP3 Server.

Table 122 on page 449 lists the methods for the Outbound Communications Manager business service.

Table 122. Outbound Communications Manager Methods

Method Display Name / Method Name	Comment
Create and Submit Request (CreateRequest)	Creates a record for a communication request, then submits it to the Communications Outbound Manager server component. This method uses the Action business component to create recipient activity records, for any template that is specified to create activities.
Send Message (SendMessage)	Sends a message to recipients specified by explicit semicolon-delimited lists of addresses (To list, CC list, and BCC list). This method accepts only literal subject and message body text and does not support field substitution. File attachments are supported. Argument values are provided directly, such as from a workflow process or a script.

Table 122. Outbound Communications Manager Methods

Method Display Name / Method Name	Comment
Send SMTP Message (SendSmtpMessage)	Sends an SMTP message to recipients specified by explicit semicolon-delimited lists of addresses (To list, CC list, and BCC list). This method accepts only literal subject and message body text and does not support field substitution or file attachments. Argument values are provided directly, such as from a workflow process or a script. Unlike the Send Message method, this method does not require that a communications profile be specified. The profile Default SMTP Profile is used.
Submit Request (SubmitRequest)	Submits a communication request, using the name or ID of a request previously created. Submits the request to the Communications Outbound Manager server component. This method uses the Action business component to create recipient activity records, for any template that is specified to create activities.

Arguments for Outbound Communications Manager Methods

This section lists the arguments for each method of the business service Outbound Communications Manager.

NOTE: In Siebel Tools, other arguments than these may also be listed. Disregard any arguments that are flagged as Hidden or Inactive.

Arguments for Create and Submit Request Method

Table 123 lists the arguments for the Create and Submit Request method.

Table 123. Arguments for Create and Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
# of Recipients (type: String)	NumRecipients	Input	N	Estimated total number of recipients. Used as hint for auto-adjustment of number of subtasks. (Not available from the user interface for creating outbound requests.)
# of Tasks (type: String)	NumTasks	Input	N	The number of subtasks to run, based on each recipient's last name.
Child Recipient Search Spec (type: String)	ChildRecipSearchSpec	Input	N	Search specification to filter iteration child business component, in the form *child_BC_name*: *child_recip_search_spec*.
Comm Profile Override (type: String)	CommProfileOverride	Input	N	Communications profile to be used for all communications templates in this request.
Comm Template Name List (type: String)	PackageNameList	Input	Y	Semicolon-delimited list of communications template names.
Comments (type: String)	Comments	Input	N	Comments for the request. Corresponds to Comments field for the request.
Create Only (type: String)	CreateOnly	Input	N	Create the communication request without any associated source, recipient, or template records specified. Used when creating a parent request.
Default Preference (type: String)	RequestDefaultMedium	Input	N	Communications channel to use if Only Send Preference is checked for request, and channel preference is not specified for a recipient. Corresponds to Default Preference field for the request.

Table 123. Arguments for Create and Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Defined Component (type: String)	DefinedComponent	Input	N	Name of the defined Siebel Server component, based on CommOutboundMgr, that is to receive this request. Corresponds to Component Name field for the request.
Message Reply Address List (type: String)	MsgReplyAddressList	Input	N	Semicolon-delimited list of reply email addresses to be used by recipients of email messages generated by the request.
Only Send Preference Flag (type: String)	RequestSendFlag	Input	N	(Currently this is set only from the user interface for creating outbound requests.) Corresponds to Only Send Preference field for the request.
Recipient Group (type: String)	RecipientGroup	Input	Y	Recipient group: the source business object and recipient business component, as selected from the Recipient Group picklist. Corresponds to Recipient Group field for the request.
Recipient Search Spec (type: String)	RecipSearchSpec	Input	N	Search specification to filter the recipients returned by the recipient business component.
Request End Date (type: String)	EndDate	Output	N	Date/time the request finished. Corresponds to End Time field for the request.
Request ID (type: String)	CommRequestId	Output	N	ID for this communication request. A request must use either Request ID *or* Request Name. Corresponds to Request # field for the request.

Table 123. Arguments for Create and Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Request Language Code (type: String)	RequestLanguageCode	Input	N	The language applicable to this request. The value for this argument is derived, in priority order, from: ■ A value explicitly specified for this parameter, such as in a workflow process, or ■ The language specified for the first template for the request, or ■ The language applicable to the current Application Object Manager. It is assumed that all recipients for a request share a common language and locale.
Request Locale Code (type: String)	RequestLocaleCode	Input	N	The locale applicable to this request. The value for this argument is derived, in priority order, from: ■ A value explicitly specified for this parameter, such as in a workflow process, or ■ The locale specified for the first template for the request, or ■ The locale applicable to the current Application Object Manager. It is assumed that all recipients for a request share a common language and locale.
Request Name (type: String)	RequestName	Input	N	Name for this communication request. A request must use either Request ID *or* Request Name. Corresponds to Description field for the request.

Table 123. Arguments for Create and Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Request Parent ID (type: String)	CommRequestParentId	Input	N	ID for the parent request for this communication request (when current request is a subrequest). Corresponds to Request # field for the request.
Request Start Date (type: String)	StartDate	Output	N	Date/time the request started. Corresponds to Start Time field for the request.
Source ID List (type: String)	SourceIdList	Input	Y	Semicolon-delimited list of IDs for recipient source records.
SRM Request ID (type: String)	SRMRequestId	Output	N	The ID for the server request for this communication request, as known to the Server Request Broker.
Status (type: String)	Status	Output	N	Status of the request. Corresponds to Status field for the request.
Status Message (type: String)	StatusMessage	Output	N	Request status message. Corresponds to Status Message field for the request.
Task Recipient Minimum (type: String)	TaskRecipMin	Input	N	Preferred minimum number of recipients for a subtask; warning issued if not met. (Not available from the user interface for creating outbound requests.)

Table 123. Arguments for Create and Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Test Address (type: String)	TestAddress	Input	N	Recipient address (for the To line) to use for testing, which overrides all others. If set to Null (case insensitive), communications drivers will not be loaded and messages will not be sent.
Web Server (type: String)	WebServer	Input	N	The Web server to be used for Siebel bookmarks (URLs). This argument corresponds to the WebServer parameter for Communications Outbound Manager. For details, see "Configuring Communications Outbound Manager" on page 262. See also the description for the Create Bookmark check box for advanced templates, in "Fields for Templates" on page 280.

Arguments for Send Message Method

Table 124 lists the arguments for the Send Message method.

Table 124. Arguments for Send Message Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
AttachFileList (type: String)	AttachFileList	Input	N	List of files to be attached to the message. Alternatively, a list of paths that corresponds to the list of files specified using AttachNameList. When you specify file names, the full path to the file must be specified. **NOTE:** All paths must be valid on the machine where Communications Outbound Manager is running. (It may be helpful to specify the Siebel Server explicitly, through the associated profile.) You specify multiple files or paths by delimiting each entry with an asterisk (*). For example, if you are specifying file names as well as paths: c:\temp\a.txt*d:\temp\b.gif For example, if you are specifying paths only (where files are specified using AttachNameList): c:\temp*d:\temp\
AttachNameList (type: String)	AttachNameList	Input	N	List of files to be attached to the message. The list of files corresponds to the list of paths specified using AttachFileList. You specify multiple files by delimiting each entry with an asterisk (*). For example: a.txt*b.gif

Table 124. Arguments for Send Message Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Bcc Address List (type: String)	MsgBccList	Input	N	Semicolon-delimited list of email addresses for the BCC line.
Cc Address List (type: String)	MsgCcList	Input	N	Semicolon-delimited list of email addresses for the CC line.
Communication Profile (type: String)	CommProfile	Input	Y	Name of the communications profile used to send the message.
Message Body (type: String)	MsgBody	Input	N	Email message body content, which may be specified from a workflow process or script. How the content is processed depends in part on whether the argument type is Expression or Literal. For example, when the type is Expression, you can enter operators such as +, specify field names for substitution, and enclose literal strings in quotes. Carriage returns to be included in output must be enclosed in quotes, such as in this example: "New Quote" + " " + [Quote Number] For more information, see *Siebel Business Process Designer Administration Guide*.
Message Subject (type: String)	MsgSubject	Input	N	Descriptive subject line for the email message.
Reply to Address List (type: String)	MsgReplyAddressList	Input	N	Semicolon-delimited list of email addresses to be used for all recipients.
To Address List (type: String)	MsgToList	Input	N	Semicolon-delimited list of email addresses for the To line.

Arguments for Send SMTP Message Method

Table 125 lists the arguments for the Send SMTP Message method.

Table 125. Arguments for Send SMTP Message Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Bcc Address List (type: String)	MsgBccList	Input	N	Semicolon-delimited list of email addresses for the BCC line.
Cc Address List (type: String)	MsgCcList	Input	N	Semicolon-delimited list of email addresses for the CC line.
From Address (type: String)	FromAddress	Input	Y	The email address of the sender.
Message Body (type: String)	MsgBody	Input	N	Email message body content, which may be specified from a workflow process or script. How the content is processed depends in part on whether the argument type is Expression or Literal. For example, when the type is Expression, you can enter operators such as +, specify field names for substitution, and enclose literal strings in quotes. Carriage returns to be included in output must be enclosed in quotes, such as in this example: "New Quote" + " " + [Quote Number] For more information, see *Siebel Business Process Designer Administration Guide*.
Message Subject (type: String)	MsgSubject	Input	N	Descriptive subject line for the email message.
To Address List (type: String)	MsgToList	Input	N	Semicolon-delimited list of email addresses for the To line.

Arguments for Submit Request Method

Table 126 lists the arguments for the Submit Request method.

Table 126. Arguments for Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
# of Recipients (type: String)	NumRecipients	Input	N	Estimated total number of recipients. Used as hint for auto-adjustment of number of subtasks. (Not available from user interface.)
# of Tasks (type: String)	NumTasks	Input	N	The number of subtasks to run, based on each recipient's last name.
Child Recipient Search Spec (type: String)	ChildRecipSearchSpec	Input	N	Search specification to filter iteration child business component, in the form *child_BC_name*: *child_recip_search_spec*.
Comm Profile Override (type: String)	CommProfileOverride	Input	N	Communications profile to be used for all communications templates in this request.
Defined Component (type: String)	DefinedComponent	Input	N	Name of the defined Siebel Server component, based on CommOutboundMgr, that is to receive this request. Corresponds to Component Name field for the request.
Recipient Search Spec (type: String)	RecipSearchSpec	Input	N	Search specification to filter the recipients returned by the recipient business component.
Request End Date (type: String)	EndDate	Output	N	Date/time the request finished. Corresponds to End Time field for the request.
Request ID (type: String)	CommRequestId	Input	N	ID for this communication request. A request must use either Request ID *or* Request Name. Corresponds to Request # field for the request.

Table 126. Arguments for Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Request Language Code (type: String)	RequestLanguageCode	Input	N	The language applicable to this request. The value for this argument is derived, in priority order, from: ■ A value explicitly specified for this parameter, such as in a workflow process, or ■ The language specified for the first template for the request, or ■ The language applicable to the current Application Object Manager. It is assumed that all recipients for a request share a common language and locale.
Request Locale Code (type: String)	RequestLocaleCode	Input	N	The locale applicable to this request. The value for this argument is derived, in priority order, from: ■ A value explicitly specified for this parameter, such as in a workflow process, or ■ The locale specified for the first template for the request, or ■ The locale applicable to the current Application Object Manager. It is assumed that all recipients for a request share a common language and locale.

Table 126. Arguments for Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Request Name (type: String)	RequestName	Input	N	Name for this communication request. A request must use either Request ID *or* Request Name. Corresponds to Description field for the request.
Request Parent ID (type: String)	CommRequestParentId	Input	N	ID for the parent request for this communication request (for when current request is a subrequest). Corresponds to Request # field for the request.
Request Start Date (type: String)	StartDate	Output	N	Date/time the request started. Corresponds to Start Time field for the request.
Status (type: String)	Status	Output	N	Status of the request. Corresponds to Status field for the request.
Status Message (type: String)	StatusMessage	Output	N	Request status message. Corresponds to Status Message field for the request.
Task Recipient Minimum (type: String)	TaskRecipMin	Input	N	Preferred minimum number of recipients for a subtask; warning issued if not met. (Not available from the user interface for creating outbound requests.)

Table 126. Arguments for Submit Request Method

Argument Display Name / Data Type	Argument Name	Type	Req'd	Comment
Test Address (type: String)	TestAddress	Input	N	Address to use for testing, which overrides all others. If it is set to Null (case insensitive), communications drivers will not be loaded and message will not be sent.
Web Server (type: String)	WebServer	Input	N	The Web server to be used for Siebel bookmarks (URLs). This argument corresponds to the WebServer parameter for Communications Outbound Manager. For details, see "Configuring Communications Outbound Manager" on page 262. See also the description for the Create Bookmark check box for advanced templates, in "Fields for Templates" on page 280.

C Views for Communications Administration

This appendix provides information about views in the Administration - Communications screen. It includes the following topics:

Views for Defining Configurations, Drivers, Profiles, Agents, and Telesets

The following views are used to specify communications configurations, communications drivers and profiles, telesets and extensions, and agents:

- **All Configurations.** Create, modify, copy, or delete a record for a named communications configuration and define or associate related data such as parameters, profiles, agents, commands, and events. See "Creating or Modifying a Communications Configuration" on page 50.
- **Communications Drivers and Profiles.** Specify parameter values for communications drivers, and define profiles to provide access to the driver and to specify override values for selected parameters. See Chapter 3, "Configuring Communications Drivers and Profiles."
- **Agent General Profile.** View or modify information about call or contact center communications users (contact-center agents and supervisors), including their ACD agent logins and passwords and associated ACD queues. Also associate an agent with one or more configurations or telesets. See "Specifying Agents" on page 59.
- **All Telesets.** Specify the telesets in your call center. For each teleset, specify its extensions (according to the switch you are using). You can specify authorized agents for the teleset here or in the Agent General Profile view. For hoteling telesets, specify the hoteling computer. See "Specifying Telesets" on page 62.
- **All Extensions.** View information about all defined extensions, and view or modify each extension's type and associated teleset. For details, see "Specifying Telesets" on page 62.

Views for Defining Events

The following views are used to specify event handlers, event responses, and event logs:

- **All Event Logs.** Create or modify each event log, specify parameters and values, and associate the event log with a configuration.
- **All Event Responses.** Create or modify each event response, specify parameters and values, and associate the event response with a configuration. Also associate the event response with one or more event logs.
- **All Event Handlers.** Create or modify each event handler, specify parameters and parameter values, and associate the event handler with a configuration, and optionally with a profile. Also associate the event handler with an event response.

For details, see "Defining Communications Events" on page 65.

Views for Defining Commands

The following views are used to specify commands and command data definitions:

- **All Command Data.** Create or modify each command data definition, specify parameters and values, and associate the command data definition with a configuration.
- **All Commands.** Create or modify each command, specify parameters and values, and associate the command with a configuration, and optionally with a profile. Also associate the command with a command data definition.

For details, see "Defining Communications Commands" on page 70.

View for Exploring Configurations

The Configuration Explorer view provides an alternative way to view or specify some of the configuration elements. It allows you to view or specify communications configuration data. See "Viewing All Communications Configuration Data" on page 52.

Views for Generating Reports or Reviewing Runtime Status Data

The following views allow you to generate reports about inbound or outbound communications activities, or display runtime status information about agents and communications work items:

- **Reports.** Generate communications-related reports. See "Generating Communications Reports" on page 231.
- **All Active Agent Status.** View information about the communications activities of agents. See "Viewing Communications Status Data" on page 233.
- **All Channel Items.** View information about active work items for each supported channel type. See "Viewing Communications Status Data" on page 233.

View for Defining Templates

The All Templates view is used to specify communications templates and outbound requests. You use it to create or modify outbound communications templates used by Siebel Email Response; by the Send Email, Send Fax, and Send Page commands; by outbound messages sent by Siebel Workflow; and by outbound communication requests for various channels. See Chapter 9, "Configuring Communications Templates."

Views for Defining Outbound Communication Requests

The following views are used to specify outbound communication requests:

- **All Outbound Requests.** Define and submit requests for outbound communications using email, fax, or paging channels. See Chapter 10, "Defining Outbound Communication Requests."
- **Outbound Request Overview.** Define and submit requests for high-volume outbound communications, including specifying recipients and templates. See Chapter 10, "Defining Outbound Communication Requests."

View for Defining Inbound Communications

The following view allows you to configure processing of inbound communications, such as for Siebel Email Response:

- **All Response Groups.** Configure processing of inbound communications. For more information, see *Siebel Email Response Administration Guide*.

View for Monitoring Inbound Communications

The following view allows you to monitor processing of inbound communications for Siebel Email Response:

- **Communications Inbound Events.** Monitor the processing of inbound communications handled by the Communications Inbound Receiver and Communications Inbound Processor server components. For more information, see *Siebel Email Response Administration Guide*.

View for Specifying Message Broadcast Settings

The following view allows you to specify message broadcast settings:

- **Message Broadcasts.** Specify message broadcast settings. For more information, see *Applications Administration Guide*.

D Upgrading from Release 6.x

This appendix provides information about upgrading Siebel implementations from previous releases, particularly release 6.x. It includes the following topics:

About Upgrading from Release 6.x

This appendix describes upgrade issues that apply to the following modules:

- Siebel CTI, release 6.x (or earlier)
- Siebel Communications Server, release 6.x
- Send Email/Send Fax/Send Page commands, release 6.x (or earlier)
- Siebel CTI scripts from release 6.x

For more information about upgrading, see the *Upgrade Guide* for the operating system you are using See also *System Requirements and Supported Platforms* on Siebel SupportWeb.

In order to understand the upgrade issues described here, you must understand how Siebel Communications Server in release 7.x differs from Siebel CTI, Siebel Communications Server, and other communications features in release 6.x.

It is recommended that you closely compare the features documented for the current version with those described in the *Siebel CTI Guide* and *Siebel Communications Server Administration Guide* for previous versions.

For information about upgrading specific modules related to Communications Server, or implementing new modules, also see the following documentation:

- *Siebel Universal Queuing Administration Guide* (new product for 7.*x*)
- *Siebel Email Response Administration Guide*
- *Applications Administration Guide*

Upgrading Siebel CTI from Release 6.x (and Earlier)

This section describes issues in upgrading implementations of Siebel CTI, release 6.x (and earlier). Upgrade issues for Siebel CTI include:

- CTI drivers
- CTI/communications configurations
- Scripting and business service methods
- Siebel CTI Connect
- Inbound call routing

Upgrade Issues for CTI Drivers

In release 6.x (and earlier), Siebel CTI drivers (whether drivers provided by Siebel Systems or custom drivers developed by customers or by other vendors) were written to the Adaptive CTI API, and were based on a client-side CTI architecture.

Due to the architectural changes for Siebel Communications Server, CTI drivers from previous versions will not function in release 7.x and subsequent releases, and cannot be upgraded. Some of the code from an existing driver may, however, be reusable in developing a replacement driver.

In release 7.x, Siebel Systems provides a CTI-specific communications driver that supports Intel NetMerge CTI middleware (formerly Dialogic CT Connect), which is included in Siebel CTI Connect.

The 7.*x* Siebel CTI Connect driver replaces the Dialogic CTI driver provided with previous versions of Siebel CTI Connect. (For versions 7.0.x and 7.5.x, the Siebel CTI Connect driver was named Dialogic CTI.)

(One other new driver is also provided for 7.*x* that enables you to use Siebel Universal Queuing to to support multichannel communications functionality: User-interactive Email.)

Siebel Systems also provides the Adaptive Communications API, which replaces the Adaptive CTI API. The Adaptive Communications API is described in Appendix A, "Developing a Communications Driver." You must use this new API to develop any custom communications drivers intended to support CTI middleware products and other communications functionality.

In previous versions, Adaptive CTI drivers used the COM object model, were loaded on each agent's desktop machine, and could be developed using Visual Basic. In release 7.x, however, Adaptive Communications drivers run in a server environment and the COM model is no longer used. It is recommended that replacement custom drivers be created using C or C++, for best performance.

In release 6.x, a single CTI driver was generally used with your CTI configuration, and profiles were not used with such drivers. To support a multichannel call or contact center, now multiple communications drivers can be used in the same environment, and may be applicable to the same agent.

For each applicable driver that supports a communications channel you require (such as voice or email), a profile must be created and associated with the communications configurations to which the driver's functionality should apply.

The driver and profile model is based on what was supported for communications adapters (now drivers) and profiles for Siebel Communications Server, release 6.x. However, new driver records can now be added in the Communications Drivers and Profiles view, to support using custom drivers. For each driver record:

- The Library Name field specifies the file, usually a library file such as a .dll or .so file, for the driver.
- The Channel String field specifies a string that must match a designated string defined within the driver file. The channel string is used to identify the driver to itself. Each new driver must include this element.

Some of the CTI configuration parameters from previous versions have become driver parameters for the Siebel CTI Connect driver. Several configuration parameters or driver parameters are new or have been renamed.

Drivers and driver parameters are configured, along with profiles and profile parameter overrides, in the Communications Drivers and Profiles view. Configuration parameters are configured in the All Configurations view.

For more information about communications drivers, see:

- Chapter 3, "Configuring Communications Drivers and Profiles"
- Appendix A, "Developing a Communications Driver"

Upgrade Issues for CTI/Communications Configurations

Many issues for upgrading from Siebel CTI release 6.x (and earlier) concern communications configurations (formerly CTI configurations), which play a critical role in defining Siebel application functionality—in particular, using the communications toolbar—for interactive communications channels such as voice.

Some of the steps for upgrading a configuration can be done automatically, if you prepare .DEF and .ini files from your previous version, and then import data from these files into a new communications configuration for the current version. More information on this process is described later in this section.

NOTE: Siebel CTI release 6.x (and earlier) exported CTI configuration data into two different files: .ini for configuration parameters and .DEF for events and commands. In the current version, communications configuration data is exported to only a single file type, .DEF. You can, however, choose which elements to export.

Background for Configuration Upgrade

As noted, additional communications channels are now supported, such as email. You must determine which channels, and which communications-related Siebel products, you require before you complete the upgrade and configuration process and bring Siebel CTI into production in the new version of the Siebel Business Applications.

Siebel Systems provides a replacement communications driver for Dialogic CT Connect CTI middleware (now Intel NetMerge), as described previously. In addition, Siebel Systems provides a replacement predefined communications configuration that is designed to work with this CTI middleware system (and also with email systems).

CTI configuration files from previous versions can be upgraded whether they were originally used with CTI drivers provided by Siebel Systems or used with custom Adaptive CTI drivers. However, the upgrade process described in this appendix performs specific functions; you must address all other configuration issues.

Communications configurations function differently in the current version than they did in previous versions. Many configuration elements have changed or are new, such as event and command parameters, macros for parameter values, special events and commands, configuration parameters, and driver parameters.

Siebel-provided configurations and drivers are enhanced so that one configuration can support multiple drivers, for multichannel communications.

The relationships between configurations, agents, telesets, extensions, and ACD queues have also changed:

- Agents are now associated with configurations, and can be associated with more than one configuration (although only one is in effect at a time).
- Telesets are no longer directly associated with configurations, only with agents.
- Agents do not need to be associated with telesets at all if they will only use hoteling telesets, or if they will only use channels other than voice.
- ACD queues are now associated with agents, not with telesets. Agents can log in to multiple ACD queues at the same time.
- Multiple extensions can be defined for the same teleset.
- The extension of a hoteling teleset is no longer used to determine the communications configuration for a user logged into a hoteling computer.

In the User Preferences screen, which replaces the Options dialog box, the options from the former CTI tab are now Communications options. In the previous version, agents could specify extensions, while in the current version, agents can specify both telesets and extensions for the next communications session. Agents can also specify which configuration to use for the next communications session, and can selectively log in to ACD queues.

Agents, telesets, extensions, and ACD queues are not part of the configuration files you export or import. However, they are upgraded automatically when you upgrade the Siebel Database, so that they are in the correct relationship in the database tables after upgrade.

For example, an agent who was previously associated with a teleset that was in turn associated with a particular configuration is directly associated with this configuration after you upgrade. The teleset remains associated with the agent, but not with the configuration.

NOTE: When teleset data is upgraded to Siebel 7, each teleset name is concatenated with a "+" followed by its database row ID. After upgrading, you can remove the added characters from the teleset names using a database script you create. Before doing so, make sure that any duplicate telesets or extensions have been resolved. The applicable table name is S_CM_TELESET, and the column name is NAME.

The following sections list new or modified configuration elements from release 6.x to release 7.x. Included are configuration parameters, Siebel CTI Connect driver parameters, Siebel CTI Connect events and commands, event handler parameters, event response parameters, event log parameters, command parameters, command data parameters, special events and commands, and macros.

Configuration Parameters in Release 7.x

Most configuration parameters from release 6.x with the prefixes "Dialing:" and "Setting:" function in the current version as configuration parameters, and do not have a prefix. These parameters are not directly associated with any communications driver. Each release 7.x configuration parameter is listed in Table 127, along with an indication of its name in release 6.x.

These parameters are described in "Specifying Parameters for Communications Configurations" on page 52.

Table 127. Configuration Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
AutoLogin	Setting:AutoLogin	Renamed. (Parameter now supports "UserPreference" value.) **NOTE:** This parameter is automatically renamed if you import configuration parameters from an .ini file exported from release 6.x.
AutoLoginCmd	N/A	New
BackupCommSessionMgr	N/A	New
BackupEnterpriseServer	N/A	New
BackupGatewayAddress	N/A	New
BackupRequestServer	N/A	New
ChannelCleanupTimer	N/A	New
CheckPopupBeforeExecute	N/A	New

Table 127. Configuration Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
CommSessionMgr	N/A	New.
DialingFilter.Rule*N*	Dialing:Filter.Rule*N*	Renamed. **NOTE:** These parameters are automatically renamed if you import configuration parameters from an .ini file exported from release 6.x.
EnterpriseServer	N/A	New.
GatewayAddress	N/A	New.
MaxCommToolbars	N/A	New.
MultiTenancy	Setting:MultiTenancy	Renamed. **NOTE:** This parameter is automatically renamed if you import configuration parameters from an .ini file exported from release 6.x.
PreferenceLoginCmd	N/A	New.
PreferenceLogoutCmd	N/A	New.
RequestServer	N/A	New.
RestoreScreenOnWork Resumed	N/A	New.
UpdateChannelStatusTable	Setting:UpdatePhoneStatus Table	Renamed. **NOTE:** This parameter is automatically renamed if you import configuration parameters from an .ini file exported from release 6.x.
UQConfigurationName	N/A	New.

Siebel CTI Connect Driver Parameters in Release 7.x

Dialogic configuration parameters from release 6.x with the prefix "Driver:" function in the current version as Siebel CTI Connect driver parameters, with either the "Driver:" or "Service:" prefix. Each release 7.x driver parameter is listed in Table 128 on page 473, along with an indication of its name in release 6.x. These parameters are described in "Siebel CTI Connect Driver Parameters" on page 339.

NOTE: No automatic upgrading occurs for these driver parameters if you import Dialogic configuration parameters from an .ini file exported from release 6.x.

Table 128. Siebel CTI Connect Driver Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
Driver:CIMServer	Driver:CIMServer	Changed from configuration parameter to driver parameter.
Driver:CIMTimeDelay	N/A	New.
Driver:CTCServer	Driver:CTCServer	Changed from configuration parameter to driver parameter.
Driver:DriverLogFile	Driver:LogFileName	Changed from configuration parameter to driver parameter, and renamed.
Driver:LogicalID	Driver:LogicalID	Changed from configuration parameter to driver parameter.
Driver:MaxServices	N/A	New.
Driver:NetCallPort	N/A	New.
Driver:NetworkType	Driver:NetworkType	Changed from configuration parameter to driver parameter.
Driver:Simulate	N/A	New.
Driver:SwitchType	Driver:SwitchType	Changed from configuration parameter to driver parameter.
N/A	Driver:TraceLevel	Obsolete.
Service:ACDDNList	N/A	New.
Service:AutoLogout	N/A	New.
Service:DNList	N/A	New.
Service:HandleRouteRequest	N/A	New.
Service:Has*KeyName*	Driver:Has*KeyName*	Changed from configuration parameter to driver parameter, and renamed.
Service:IsQueueRequired	N/A	New.
Service:IsSiemens	N/A	New.

Table 128. Siebel CTI Connect Driver Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
Service:SelectDN	N/A	New.
Service:ServiceLogFile	N/A	New.
Service:Use1StepTransfer	Driver:Use1StepTransfer	Changed from configuration parameter to driver parameter, and renamed.
LogDebug	N/A	New.
MaxLogKB	Driver:MaxLogKB	Changed from configuration parameter to driver parameter, and renamed.
ReleaseLogHandle	N/A	New.

New Siebel CTI Connect Event in Release 7.x

A new Siebel CTI Connect driver event is listed in Table 129. This event is described in "Siebel CTI Connect Events" on page 354.

Table 129. New Dialogic CTI Event in Version 7.x

Release 7.x
EventAnswerCall

New Siebel CTI Connect Commands in Release 7.x

New Siebel CTI Connect driver commands are listed in Table 130. These commands are described in "Siebel CTI Connect Commands" on page 346.

Table 130. New Dialogic CTI Commands in Version 7.x

Release 7.x
ChangeBusyState
MergeCall
RouteCall
SelectDN1... SelectDN5
SimulateCall
ToggleForward

Event Handler Parameters in Release 7.x

Where there are differences between release 6.x and release 7.x, each release 7.x event handler parameter is listed in Table 131, along with an indication of its name or status in release 6.x. These parameters are described in "Event Handler Parameters" on page 105.

Table 131. New or Changed Event Handler Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
DeviceEvent	DeviceEvent	Now specified as a field value (Device Event), not as an event handler parameter.
FilterSpec	N/A	New.
Order	Order	Now specified as a field value (Order), not as an event handler parameter.
Profile	N/A	New. Specified as a field value (Profile), not as an event handler parameter.
Response	Response	Now specified as a linked child record (Event Response), not as an event handler parameter.
ServiceMethod	N/A	New.
ServiceParam	N/A	New.

Event Response Parameters in Release 7.x

Where there are differences between release 6.x and release 7.x, each release 7.x event response parameter is listed in Table 132, along with an indication of its name or status in release 6.x. These parameters are described in "Event Response Parameters" on page 112.

Table 132. New or Changed Event Response Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
AddLog	AddLog	Now specified as a field value (Type) in a child record (Associated Event Logs), not as an event response parameter.
AddRecordApplet	N/A	New.
ContextLog	ContextLog	Now specified as a field value (Type) in a child record (Associated Event Logs), not as an event response parameter.

Table 132. New or Changed Event Response Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
FindLog	FindLog	Now specified as a field value (Type) in a child record (Associated Event Logs), not as an event response parameter.
N/A	InvokeResponseIfNoData2	Obsolete.
Log	Log	Now specified as a field value (Type) in a child record (Associated Event Logs), not as an event response parameter.
MultiLog	MultiLog	Now specified as a field value (Type) in a child record (Associated Event Logs), not as an event response parameter.
OpenView	N/A	New.
SingleField	N/A	New.
SingleLog	SingleLog	Now specified as a field value (Type) in a child record (Associated Event Logs), not as an event response parameter.
WorkObject	CallObject	Renamed. Can be used with multiple types of work items.

Event Log Parameters in Release 7.x

Where there are differences between release 6.x and release 7.x, each release 7.x event log parameter is listed in Table 133, along with an indication of its name and status in release 6.x. These parameters are described in "Event Log Parameters" on page 124.

Table 133. New or Changed Event Log Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
AfterWork	AfterCall	Renamed. **NOTE:** This parameter is automatically renamed if you import events and commands from a .DEF file exported from release 6.x.
WorkTrackingObj	N/A	New.

Command Parameters in Release 7.x

Where there are differences between release 6.x and release 7.x, each release 7.x command parameter is listed in Table 134, along with an indication of its name or status in release 6.x. These parameters are described in "Command Parameters" on page 130.

Table 134. New or Changed Command Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
AcceptReject	N/A	New.
AROnRejectCmd	N/A	New.
ARTitle	N/A	New.
ARWorkItemID	N/A	New.
CmdChannelOnFocus	N/A	New.
CmdData	CmdData	Now specified as a linked child record (Command Data), not as a command parameter.
ExecuteAll	N/A	New.
ExecUntilOK	N/A	New.
FilterSpec	N/A	New.
HotKeyText	N/A	New.
IndicateActiveCmd	N/A	New.
MenuPosition	Order	Renamed. **NOTE:** This parameter is automatically renamed, for any command that was configured to appear in the CTI menu or the shortcut menu, if you import events and commands from a .DEF file exported from release 6.x.
MultiActiveCmdIcon	N/A	New.
Profile	N/A	New. Specified as a field value (Profile), not as a command parameter.
N/A	ScriptParam	(This is a command data parameter, incorrectly documented as a command parameter for 6.*x*.)
SubCommand_*N*	SubCommand	Renamed. Now specified as a child record (Subcommands), not as a command parameter.

Command Data Parameters in Release 7.x

Several new command data parameters are available in the current version. Each release 7.x command data parameter is listed in Table 135, along with an indication of its name in release 6.x. These parameters are described in "Command Data Parameters" on page 142.

Table 135. New or Changed Command Data Parameters in Release 7.x

Release 7.x	Release 6.x	Comments
ScriptParam	N/A	(This command data parameter was incorrectly documented as a command parameter for 6.*x*.)
SelectQuerySpec	N/A	New.
WorkTrackingObj	N/A	New.

Special Events in Release 7.x

Special events are new in release 7.x. Each release 7.x special event is listed in Table 136. These special events are described in "Special Events for Device Events" on page 85.

Table 136. New Special Events in Version 7.x

Release 7.x
@HandleNonRealtimeWorkItem
@PreIndicateNewWorkItemEvent
@PostIndicateNewWorkItemEvent
@PreWorkItemReleasedEvent
@PostWorkItemReleasedEvent
@PreWorkItemResumedEvent
@PostWorkItemResumedEvent
@PreWorkItemStartedEvent
@PostWorkItemStartedEvent
@PreWorkItemSuspendedEvent
@PostWorkItemSuspendedEvent
@RuntimeEvent

Special Commands in Release 7.x

Where there are differences between release 6.x and release 7.x, each release 7.x special command is listed in Table 137, along with an indication of its name or status in release 6.x. These special commands are described in "Special Commands for Device Commands" on page 89.

Table 137. New or Changed Special Commands in Release 7.x

Release 7.x	Release 6.x	Comments
@CreatePopupFrame	N/A	New
@InvokeSWECommand	N/A	New
@OpenView	N/A	New
@UpdateRecord	N/A	New
@UQAbortWorkItem	N/A	New
@UQAddWorkItem	N/A	New
@UQAcceptWorkItem	N/A	New
@UQAgentAvailable	N/A	New
@UQAgentChangeReadyState	N/A	New
@UQAgentInitAuxWork	N/A	New
@UQAgentLogon	N/A	New
@UQAgentLogout	N/A	New
@UQAgentSignOnOff	N/A	New
@UQBlindTransferWorkItemToAgent	N/A	New
@UQBlockAgentChannel	N/A	New
@UQCancelTransfer	N/A	New
@UQChangeAgentMediaMode	N/A	New
@UQCompleteTransfer	N/A	New
@UQCompleteWorkItem	N/A	New
@UQHoldWorkItem	N/A	New
@UQInitTransfer	N/A	New
@UQRejectWorkItem	N/A	New
@UQTransfer	N/A	New
@UQTransferWorkItemToRoute	N/A	New
@UQUnBlockAgentChannel	N/A	New

Table 137. New or Changed Special Commands in Release 7.x

Release 7.x	Release 6.x	Comments
@UQUnHoldWorkItem	N/A	New
@ViewWorkObject	@ViewCallObject	Renamed

Macros in Release 7.x

Where there are differences between release 6.x and release 7.x, each release 7.x macro is listed in Table 138, along with an indication of its name or status in release 6.x. These macros are described in "Macros for Parameter Values" on page 182.

Table 138. New or Changed Macros in Release 7.x

Release 7.x	Release 6.x	Comments
@ACDDNList	N/A	New
@ClientHostName	N/A	New
@ClientIP	N/A	New
@Configuration	N/A	New
@CountryCode	N/A	New
@DeselectedWorkItem	N/A	New
$DialingRuleMethod	N/A	New
@DNList	N/A	New
$GetCommandStatus	N/A	New
$GetInboundWorkItemAttr	N/A	New
$GetWorkItemAttr	N/A	New
$HotelingPhone	@HotelingPhone	Renamed
@Language	N/A	New
@Now	N/A	New
@PrimaryQueueList	N/A	New
@QueueList	N/A	New
@SelectedDN	N/A	New
@SelectedQueue	N/A	New
@SelectedText	N/A	New
@SelectedWorkItem	N/A	New
@WorkDuration	@CallDuration	Renamed

Table 138. New or Changed Macros in Release 7.x

Release 7.x	Release 6.x	Comments
@WorkObjectID	@CallObjectId	Renamed
@WorkStartTime	@CallStartTime	Renamed

Preparing to Upgrade the Configuration

In the All Configurations view in the Administration - Communications screen, you can import a configuration file that you previously exported from Siebel CTI release 6.x. The configuration data is manipulated so that your existing elements are properly mapped to the way in which events and commands and related elements are structured in the current version. You do this after the general upgrade.

Device Events Prefixed with "Sys_" No Longer Supported

In release 7.x, communications event handlers can no longer use device events prefixed with "Sys_". Such device events were supported, across CTI drivers, in previous versions. Before you complete upgrading your communications configuration, you must make sure to make the following changes:

For users of CTI drivers for Intel NetMerge:

- Change Sys_InboundCall device events to InboundCall
- Change InboundCall device events to TpAnswered

For more information, see "Handling an Inbound Call Received by an Agent" on page 109 and "Using Device Event to Enhance Screen Pop Performance" on page 207.

Users of CTI drivers for other middleware products (that is, drivers supported by other vendors than Siebel Systems) must make comparable changes applicable to their Siebel CTI implementations.

Event Logs Specified in Event Responses

Each event log specified in a single event response must be unique. If multiple parameters in an event response definition specify the same value, you must either delete (or comment out) event log parameters that specify duplicate values, or define new event logs and specify them in order to meet the uniqueness requirement. Otherwise, when you import your .DEF file, errors will be generated.

For more information, see "Event Responses" on page 110.

Version Identifier in 7.x .DEF File

Configuration files exported from release 7.x now contain an entry identifying the version. Configuration files from previous versions that you import for upgrade purposes must *not* contain the following entry, or else no upgrade manipulations will be performed:

```
[Siebel]
    CommServerVersion = "7.0"
```

Event and Command Manipulation During Upgrade

Before you import, you must make sure you do not have (or that you do not require) existing commands or command data definitions with certain names, or those elements will be replaced when you import. Most of these changes are done in order to support the communications toolbar configuration in the current version.

Event-related changes are as follows:

- **Event logs containing AfterCall parameter.** For any event log that includes the parameter AfterCall, this parameter is renamed AfterWork when you import the configuration file.

Command-related changes are as follows:

- **AnswerCall and AcceptWorkGroup commands.** If AnswerCall is found, this command will be renamed to AcceptWorkGroup. If AcceptWorkGroup already exists, it will be replaced by this command, or by a newly created command defined as follows:

```
[Command:AcceptWorkGroup]
    DeviceCommand = "AnswerCall"
    Hidden = "TRUE"
```

- **CancelForwardCall command.** If CancelForwardCall exists, it will be removed.
- **ChangeBusyState and ChangeBusyStateGroup commands.** If ChangeBusyState is found, this command will be renamed to ChangeBusyStateGroup. If ChangeBusyStateGroup already exists, it will be replaced by this command, or by a newly created command defined as follows.

```
[Command:ChangeBusyStateGroup]
    DeviceCommand = "ChangeBusyState"
    Hidden = "TRUE"
```

- **ChangeNotReadyState and NotReadyGroup commands.** If ChangeNotReadyState is found, this command will be renamed to NotReadyGroup. If NotReadyGroup already exists, it will be replaced by this command, or by a newly created command defined as follows:

```
[Command:NotReadyGroup]
    DeviceCommand = "ChangeNotReadyState"
    Hidden = "TRUE"
```

- **ConferenceComplete command.** If ConferenceComplete is found, no change will be made to this command. If ConferenceComplete is not found, a new command will be created, defined as follows:

```
[Command:ConferenceComplete]
    DeviceCommand = "ConferenceComplete"
    Hidden = "TRUE"
```

- **ConferenceTransferGroupInToolbar and ConferenceTransferGroup commands.** If ConferenceTransferGroupInToolbar is found, this command will be renamed to ConferenceTransferGroup. If ConferenceTransferGroupInToolbar is not found, no change will be made. If ConferenceTransferGroup already exists, it will be replaced by a newly created command defined as follows:

```
[Command:ConferenceTransferGroup]
    SubCommand_0 = "ConferenceComplete"
```

- **ConsultativeTransferGroupInToolbar and ConsultativeTransferGroup commands.** If ConsultativeTransferGroupInToolbar is found, this command will be renamed to ConsultativeTransferGroup. If ConsultativeTransferGroupInToolbar is not found, no change will be made. If ConsultativeTransferGroup already exists, it will be replaced by a newly created command defined as follows:

```
[Command:ConsultativeTransferGroup]
    SubCommand_0 = "TransferComplete"
```

- **ForwardCall and ForwardWorkGroup commands.** If ForwardCall is found, this command will be renamed to ForwardWorkGroup. If ForwardWorkGroup already exists, it will be replaced by this command, or by a newly created command defined as follows:

```
[Command:ForwardWorkGroup]
    DeviceCommand = "ToggleForward"
    Hidden = "TRUE"
```

- **HoldCall and SuspendWorkGroup commands and SuspendWorkGroup command data.** If HoldCall is found, this command will be renamed to SuspendWorkGroup. If SuspendWorkGroup already exists, it will be replaced by this command, or by a newly created command and command data, defined as follows:

```
[Command:SuspendWorkGroup]
    DeviceCommand = "HoldCall"
    Hidden = "TRUE"

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

- **Login and SignOnGroup commands and SignOnGroup command data.** If LogIn is found, this command will be renamed to SignOnGroup. If SignOnGroup already exists, it will be replaced by this command, or by a newly created command and command data, defined as follows:

```
[Command:SignOnGroup]
    DeviceCommand = "LogIn"
    CmdData = "SignOnGroup"

[CmdData:SignOnGroup]
    Param.ACDQueue = "{@QueueList}"
    Param.AgentId = "{@AgentId}"
    Param.AgentPin = "{@AgentPin}"
```

- **LogOut and SignOffGroup commands and SignOffGroup command data.** If LogOut is found, this command will be renamed to SignOffGroup. If SignOffGroup already exists, it will be replaced by this command, or by a newly created command and command data, defined as follows:

```
[Command:SignOffGroup]
    DeviceCommand = "LogOut"
    CmdData = "SignOffGroup"

[CmdData:SignOffGroup]
    Param.ACDQueue = "{@QueueList}"
```

- **MakeCallGroupInToolbar, MakeCallGroup, and InitiateWorkGroup commands.** If MakeCallGroupInToolbar is found, this command will be renamed to MakeCallGroup. In this case, if MakeCallGroup already exists, it will be replaced by this command. (If MakeCallGroupInToolbar is not found, this means that the Make Call button—now the Initiate Work Item button—was not previously enabled.)

  In the case that MakeCallGroupInToolbar is found (and renamed to MakeCallGroup), then the InitiateWorkGroup command is created, defined as shown below. If InitiateWorkGroup already exists, it will be replaced by the newly created command, defined as follows:

  ```
  [Command:InitiateWorkGroup]
      SubCommand = "MakeCallGroup"
      Hidden = "TRUE"
  ```

- **MuteTransferGroupInToolbar and BlindTransferGroup commands.** If MuteTransferGroupInToolbar is found, this command will be renamed to BlindTransferGroup. In this case, if BlindTransferGroup already exists, it will be replaced by this command. (If MuteTransferGroupInToolbar is not found, this means that the Mute Transfer button—now the Blind Transfer button—was not previously enabled.)

- **ReleaseCall and ReleaseWorkGroup commands.** If ReleaseCall is found, this command will be renamed to ReleaseWorkGroup. If ReleaseWorkGroup already exists, it will be replaced by this command, or by a newly created command defined as follows:

  ```
  [Command:ReleaseWorkGroup]
      DeviceCommand = "ReleaseCall"
      Hidden = "TRUE"
  ```

- **ResetState and ResetStateGroup commands.** If ResetState is found, this command will be renamed to ResetStateGroup. If ResetStateGroup already exists, it will be replaced by this command, or by a newly created command defined as follows:

  ```
  [Command:ResetStateGroup]
      DeviceCommand = "ResetState"
      Hidden = "TRUE"
  ```

- **RetrieveCall and RetrieveWorkGroup commands.** If RetrieveCall is found, this command will be renamed to RetrieveWorkGroup. If RetrieveWorkGroup already exists, it will be replaced by this command, or by a newly created command defined as follows:

  ```
  [Command:RetrieveWorkGroup]
      DeviceCommand = "RetrieveCall"
      Hidden = "TRUE"
  ```

- **TransferComplete command.** If TransferComplete is found, no change will be made to this command. If TransferComplete is not found, the command will be created and defined as follows:

  ```
  [Command:TransferComplete]
      DeviceCommand = "TransferComplete"
      Hidden = "TRUE"
  ```

- **UnHoldCall and ResumeWorkGroup commands and ResumeWorkGroup command data.** If UnHoldCall is found, this command will be renamed to ResumeWorkGroup, and the command data ResumeWorkGroup will be associated with it. If ResumeWorkGroup already exists, it will be replaced by this command, or by a newly created command and command data, defined as follows:

```
[Command:ResumeWorkGroup]
    DeviceCommand = "UnHoldCall"
    Hidden = "TRUE"
    CmdData = "ResumeWorkGroup"

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

- **Subcommands.** For any command that contains SubCommand parameters to specify subcommands, these subcommands will have the Order field set to 0 (zero) when you import the configuration file from the previous version.

  In .DEF files exported from the current version, subcommand records are represented by parameters with the name in the form "SubCommand_*N*", where *N* represents the value of the Order field for the subcommand record in the All Commands view (of the Administration - Communications screen).

  If, before you import, you manually modify subcommand parameters to use the new form "SubCommand_*N*", the value you use for *N* for each subcommand will be used for the value of the Order field for the subcommand record.

  After importing the configuration data, assign a value in the Order field for each subcommand, as appropriate.

- **Commands for the CTI menu or the shortcut menu.** For any command configured to appear in the CTI menu or the shortcut menu, the Order parameter will be renamed to the MenuPosition parameter.

  For previous versions, a command is enabled for the CTI menu if the parameter Hidden is set to TRUE or is not defined; a command is enabled for the shortcut menu if the parameter LocalMenu is set to TRUE.

  For the current version, the shortcut menu is now the applet-level menu, and the CTI menu is now the Communications submenu of the Tools application-level menu.

Upgrading the Configuration

Follow this general procedure in order to upgrade Siebel CTI and your communications configuration to the current version.

To upgrade your CTI configuration from a previous version

1. In release 6.x (or earlier), export your CTI configuration data to files:
   - Export configuration parameters to an .ini file.

- Export events and commands to a .DEF file.

Prior to release 5.x, CTI configuration data existed *only* in the .DEF and .ini files, and was not stored in the Siebel Database.

2. Follow documented procedures for upgrading the Siebel application and database from your previous version to the current version.
3. Optionally, add any new custom communications driver (to replace your CTI driver used in a previous version) in the Communications Drivers and Profiles view, in the Administration - Communications screen.
4. Optionally, for the Siebel CTI Connect driver, enable logging for Communications Server activity for the driver (Driver:DriverLogFile and Driver:LogDebug parameters), in order to record all of the upgrade actions that are performed on the configuration files.
5. Create a profile for any custom communications driver, or for the Siebel-provided CTI driver you use in the current version.
6. Associate the driver profile with the upgraded version of the same configuration you used in the earlier release.
7. Optionally, import your .ini file into the upgraded version of the same configuration from which you exported the configuration parameters in the previous version.

   Because only a small number of parameters are imported and upgraded, you may choose to re-create all necessary configuration and driver parameters rather than import. For more information, see "Configuration Parameters in Release 7.x" on page 471 and "Siebel CTI Connect Driver Parameters in Release 7.x" on page 473.
8. Import your .DEF file into the upgraded version of the same configuration from which you exported the events and commands in the previous version.
9. For all configuration parameters from the previous release that have become driver parameters in the current release, create records for the driver parameters, as applicable, and specify the appropriate values for them.

   Some driver parameter values may best be specified in the form of profile parameter overrides.
10. Optionally, add or modify any other elements in your configuration, as necessary.

    For example, you may choose to adapt elements from the Siebel-provided communications configurations for use in your upgraded configuration. This may be especially important if you support a multichannel environment.

    Also review the changes described in "Upgrade Issues for CTI/Communications Configurations" on page 469 and following subsections, and alter your configuration as appropriate.
11. Test your upgraded environment, then enable communications for your agents.

    For information about enabling communications, see "Enabling Session Communications and Simulation" on page 240.

Upgrade Issues for Communications Toolbar and Menu Commands

Several changes affect the communications toolbar and communications menu commands. For more information about configuring toolbars and menus, see *Configuring Siebel Business Applications* and *Siebel Developer's Reference*. Also consult SupportWeb on the Siebel Systems Web site for relevant information—such as, for example, how CTI toolbar configuration changed between release 5.x and release 6.x.

CTI/Communications Toolbar Configuration

This section provides upgrade information for the communications toolbar:

- The CTI toolbar in previous versions is now the communications toolbar.
- In Siebel Tools, "Communication Toolbar" is the name of the object definition for the communications toolbar.
- The business service methods and Siebel Tools commands invoked by each toolbar button have changed.

  When you upgrade your communications configuration as described earlier, communications command names that invoke toolbar buttons are changed to match the business service methods that the toolbar buttons are now configured to invoke.
- The Siebel Tools command names and bitmaps for toolbar buttons have changed.
- Some functions that used two CTI toolbar buttons in release 6.x now use only a single button each. These functions include Consultative Transfer, Consultative Conference, and Forward Calls/ Cancel Forward.
- Image files for toolbar buttons are now .gif files, instead of .bmp files.
- Drag-and-drop toolbar customization is no longer supported.
- The floating toolbar display mode is no longer supported.

General Steps for Upgrading CTI/Communications Toolbar

If you have any custom toolbar buttons, you must perform the following general steps:

1. Convert all .bmp files used for toolbar button icons into .gif files, or create the icon files you require, in .gif format. Files must be 18 x 18 pixels in dimension. The files must be located in the correct place in the Siebel installation.
2. For each icon file used for a toolbar button, create a new bitmap object definition in Siebel Tools.
3. For each custom toolbar button, create a new command object definition.

For more information, see "About Communications Toolbar Configuration" on page 151.

CTI/Communications Menu Configuration

This section provides upgrade information for communications menu commands:

- The CTI menu in previous versions is now the Communications submenu of the Tools application-level menu. (In earlier releases than release 7.x, this submenu was located in the View application-level menu.)
- The shortcut ("local") menu in previous versions is now the applet-level menu.

For more information, see "Configuring Communications Menu Commands" on page 160.

Upgrade Issues for Scripting and Business Service Methods

Any references to the "CTI Client" business service in Siebel VB or Siebel eScript scripts from previous versions must be changed to the "Communications Client" business service in the current version.

For more information, see the following sections:

- "Using Business Services with Communications Server" on page 216
- "Integrating with Siebel Scripting Languages" on page 222
- "Using Siebel Scripts from Release 6.x" on page 492
- Appendix B, "Communications Server Business Services"

Upgrade Issues for Siebel CTI Connect

New versions of Intel NetMerge middleware (formerly Dialogic) software products are applicable to Siebel CTI Connect for the current version. Customers should uninstall and reinstall their Intel NetMerge Call Processing Software and Call Information Manager software.

For more information about Siebel CTI Connect, see "About Siebel CTI Connect" on page 337.

For more information about support for third-party products, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

Upgrade Issues for Inbound Call Routing

In release 6.x (and earlier), inbound call routing functionality was offered through the use of database stored procedures that were invoked by the CTI Inbound Call Router server component. In release 7.x, this component is no longer available.

In release 7.x, inbound call routing capability can be implemented using Siebel Universal Queuing, which can support a multichannel environment by using a single routing engine to route inbound work items of various channel types, including voice calls or email requests. For more information, see *Siebel Universal Queuing Administration Guide*.

As in previous versions, inbound routing of voice calls can still be implemented using routing capabilities of an ACD (Automatic Call Distributor) system. ACD routing for voice calls can be implemented alongside of a Siebel Universal Queuing implementation for routing work items of other channel types.

Upgrading Communications Server from Release 6.x

This section describes issues in upgrading Siebel implementations using Communications Server, release 6.x. Upgrade issues for Siebel Communications Server include:

- Communications drivers and profiles
- Communications templates
- Siebel Server component name
- Business service methods

Upgrade Issues for Communications Drivers and Profiles

In release 6.x, communications drivers for Siebel Communications Server were called communications adapters.

Existing profiles you created in release 6.x for the Internet SMTP/POP3 Server driver/adapter will still work in the current version, but you can now take advantage of additional supported driver parameters and other driver settings. The From Address driver parameter, which was previously optional for outbound communications, is now required in all cases.

NOTE: Other drivers/adapters for email and fax from release 6.x (Extended MAPI, Microsoft Exchange Server, and Microsoft SMTP Service) have *not* been updated by Siebel Systems and are now unsupported. Use the Internet SMTP/POP3 Server driver instead.

For information about supported email servers, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

For more information, see Chapter 3, "Configuring Communications Drivers and Profiles."

Upgrade Issues for Communications Templates

In release 6.x, communications templates for Siebel Communications Server were called communications packages. Similarly, template items were called package items.

Existing templates (packages) you created in release 6.x will still work in the current version, but you can now take advantage of enhanced template functionality, and you can use templates in a wider range of contexts.

Communications templates are created and managed in the My Templates view (Communications screen) or All Templates view (Administration - Communications screen). For more information, see Chapter 9, "Configuring Communications Templates."

Response templates previously used for Siebel Email Response in release 6.x are upgraded, and can be accessed with the other communications templates in release 7.x. After upgrade, the name of each template has been appended with the template type, in parentheses.

For example, a template named "Dear John" of type Body will now be named "Dear John (Body)". You may choose to adjust the values of other template fields after upgrade, to take advantage of current template features. Use the Simple tab in the appropriate templates view to modify such templates.

Templates used for the Send Email, Send Fax, or Send Page commands in release 6.x will not work in the current version. Such templates will need to be re-created as communications templates in the manner described in this book. Use the Simple tab in the appropriate templates view to re-create such templates.

In templates you re-create for the Send commands, notations for indicating fields to be substituted with data from the Siebel Database must change. Such fields must be enclosed with brackets ([and]) instead of with % symbols. In addition, the business component name must precede each field name, followed by a period. For example, "%SR Number%" must change to "[Service Request.SR Number]".

Communications templates are now used for:

- Outbound communication requests (for all communications channels)
- Siebel Email Response outbound replies
- Send Email command
- Send Fax command
- Send Page command

The same template infrastructure is applicable to all of the features identified above. However, different features will use different template settings, and some advanced template features do not apply in certain contexts of use.

Upgrade Issues for Communications Manager Server Component

The Siebel Server component for sending outbound communications has been renamed. In release 6.x, the server component name was Communications Manager (CommMgr). In the current version, this component is named Communications Outbound Manager (CommOutboundMgr).

Upgrade Issues for Outbound Communications Manager Business Service

Workflow processes (or scripts) that invoke business service methods (Outbound Communications Manager business service) that support outbound communications server functionality will still work in release 7.x, but you can now take advantage of additional supported method arguments. In addition, a new method is available, Send SMTP Message.

For more information, see the following sections:

- "Using Business Services with Communications Server" on page 216
- "Integrating with Siebel Scripting Languages" on page 222
- Appendix B, "Communications Server Business Services"

Upgrading Send Email/Fax/Page from Release 6.x (and Earlier)

This section describes issues in upgrading the Send Email, Send Fax, and Send Page commands from release 6.x (and earlier). Upgrade issues for these commands include:

- Migration to server-side integration using Siebel Communications Server infrastructure for Send Email and Send Fax commands, or migration to client-side integration with Lotus Notes or Microsoft Outlook (for Send Email)
- New communications template infrastructure
- Migration to release 7.x Siebel Web Client architecture
- New user interface configuration, including communications toolbar

For information about using these commands, see "Sending Email, Fax, and Page Messages" on page 324.

Migration to Siebel Communications Server

Send Email and Send Fax commands in release 6.x integrated with email application client software on each user's machine, such as Microsoft Outlook.

In release 7.x, these commands use the Communications Outbound Manager server component to submit the email and fax messages to an email server, which actually sends the messages. Applicable business service methods are used to invoke the server component functionality.

NOTE: In version 7.5.*x* and later, the Send Email command can optionally be configured to work with Microsoft Outlook or Lotus Notes. For more information, see "Configuring Client-Side Integration for Send Email" on page 383. The rest of this section applies to the default Send Email configuration that requires using Communications Outbound Manager, and applies to both Send Email and Send Fax (Send Fax cannot use Microsoft Outlook or Lotus Notes).

To support the Send Email and Send Fax commands, an administrator must configure and run one or more instances of the Communications Outbound Manager server component, and must configure profiles for the Internet SMTP/POP3 Server driver (or another applicable communications driver).

Each end user may also need to create one or more profiles for personal use, such as to specify the user's email address as the sender (From Address parameter) for outbound messages. The My Profiles view in the Communications screen is provided for this purpose.

NOTE: Most end user responsibilities do not have access to the My Profiles view. If you need to provide this view to your users, an administrator must add it to the end user responsibilities for your Siebel implementation.

For more information, see "Upgrade Issues for Communications Drivers and Profiles" on page 489.

Communications Templates

Custom templates may need to be re-created. For more information, see "Upgrade Issues for Communications Templates" on page 489.

Migration to Siebel Web Client and Communications Toolbar

The configuration of the Send Email, Send Fax, and Send Page commands has changed. The commands are available to end users in two places: the File application-level menu and the communications toolbar. In addition, keyboard shortcuts are still available.

These commands are configured through a series of object definitions in Siebel Tools and, if the communications toolbar is used, through configuring commands and command data definitions in the applicable communications configurations.

For more information, see "About Communications Toolbar Configuration" on page 151 and "Configuring Communications Menu Commands" on page 160.

Using Siebel Scripts from Release 6.x

Siebel VB and Siebel eScript scripts that you used with Siebel CTI release 6.x can still be used with the current version of Siebel Communications Server to extend your CTI-enabled Siebel applications using the Siebel client.

NOTE: Scripts that support the objects and methods described in this chapter should not also attempt to support new release 7.x Communications Server functionality. In general, it is recommended to use release 7.x scripting capabilities, such as those described in "Integrating with Siebel Scripting Languages" on page 222.

Such scripts can be used in the following ways:

- Communications events or commands can invoke Siebel VB or Siebel eScript scripts.

  - Communications events can invoke Siebel VB or Siebel eScript scripts, for instance on the arrival of an incoming call or upon disconnection.
  - Communications commands can invoke Siebel VB or Siebel eScript scripts.
- Siebel VB or Siebel eScript scripts can invoke communications commands and access call data fields:
  - Siebel VB or Siebel eScript scripts can invoke communications commands. Commands such as transferring a call can be invoked from Siebel VB–based menu items, buttons, or toolbar buttons.
  - Siebel VB or Siebel eScript scripts can access call data fields such as ANI, DNIS, and digits collected from an IVR system.

The integration with Siebel VB and Siebel eScript described in this section allows customers to write call handlers and other advanced communications commands in Siebel VB or Siebel eScript.

All release 6.x Siebel CTI methods are exposed and can be called with any applicable parameters. All release 6.x data fields (such as ANI, DNIS, and so on) are accessible. A Siebel VB or Siebel eScript script can be invoked on any communications event for a CTI driver (call incoming, disconnected, and so on).

NOTE: For scripts invoked to handle events: When a CTI parameter is passed to a script, the receiving function should include input arguments in the function declaration.

The CTI-related objects and interfaces are supported only by Siebel VB or Siebel eScript. The main purpose of such interfaces is to allow communications event handling (such as screen-pop logic). But the interface can also be used to take complete control of the CTI technology. CTI objects can be used to obtain information about the current call, to intercept communications events and perform the necessary handling, or to control calls (make calls, transfer calls, and so on).

NOTE: Define scripts that work with Siebel CTI in the general section of the application object, where they will be compiled into the repository (.srf) file for the Siebel Application Object Manager (for the Siebel Web Client). Siebel CTI data objects and methods are for the active call object.

For more information, see the following:

- *Siebel System Administration Guide*
- *Configuring Siebel Business Applications*
- *Using Siebel Tools*
- *Siebel Developer's Reference*

CTIService Object Type

The methods of the CTIService object type are described in this section.

For descriptions of the CTIData object type, see "CTIData Object Type" on page 495.

Scripting examples using Siebel VB and Siebel eScript follow.

The CTIService object type provides the features of the Siebel CTI integration. This object is created when the Siebel application is running with the enabled CTI option. This object type has methods that allow call control, CTI event handing, and access to CTI information.

The CTI service provides access to the communications commands by name. The communications commands and their names are specific to the particular CTI integration. For Siebel CTI Connect, commands are described in Chapter 12, "Using Siebel CTI Connect." Also refer to documentation from the middleware vendor.

GetCTIService

```
Application.GetCTIService -> CTIService
```

The GetCTIService method returns the CTIService object if the Siebel application is running with CTI enabled. If CTI is not enabled, then it returns Nothing.

IsCommandEnabled

```
CTIService.IsCommandEnabled command_name -> return_value
```

IsCommandEnabled returns 0 if the specified communications command is currently enabled and returns 1 if the command is disabled. For example, the AnswerCall command is enabled only when the phone is ringing.

IsCommandChecked

```
CTIService.IsCommandChecked command_name -> return_value
```

IsCommandChecked returns 1 if the state of the specified communications command is triggered; otherwise it returns 0. For example, when an agent is in the Not Ready state, the command ChangeNotReadyState is triggered and IsCommandChecked returns 1.

InvokeCommand

```
CTIService.InvokeCommand command_name [, parameter]
```

Invokes the device command (for a communications driver) by name, with an optional parameter.

InvokeCommandWithData

```
CTIService.InvokeCommandWithData command_name [, ctiDataObject]
```

Invokes the device command by name, with the optional ctiDataObject parameter. The ctiDataObject parameter can be used to attach user-defined data to the current call, for instance when you are transferring a call or initiating a call. It can also be used to pass multiple parameters to the command when this is required.

GetCurrentCallData

```
CTIService.GetCurrentCallData -> CTIData
```

Returns a read-only copy of the CTIData object for the current call. The returned object can be used to retrieve information about the current call, such as the CTI fields ANI, DNIS and any other switch, middleware, or user-defined data that is associated with the call.

CreateData

```
CTIService.CreateData -> CTIData
```

Creates and returns the new CTIData object. This object can be used to specify parameters for the InvokeCommandWithData method. Use this method if you need to change fields on the active call object.

NotifyEventHandlerBlocking

```
CTIService.NotifyEventHandlerBlocking
```

This method is called before beginning a lengthy operation or interaction with the user through the user interface of the Siebel application. This will prompt Siebel CTI to complete some unfinished operations.

For example, suppose there is a Siebel VB or Siebel eScript script associated with the Answer Call button in the communications toolbar. When the phone rings and the agent clicks the Answer Call button, Siebel CTI will call the Siebel VB or Siebel eScript handler first and answer the phone only after that. Then a Service Request window can be invoked showing the current service request for the caller.

If the Siebel VB or Siebel eScript code internally performs some lengthy operation, then the agent may want to pick up the phone before the script ends. Calling the NotifyEventHandlerBlocking method accomplishes this and immediately connects the agent with the caller.

CTIData Object Type

The methods of the CTIData object type are described in this section. For descriptions of the CTIService object type, see "CTIService Object Type" on page 493.

Scripting examples using Siebel VB and Siebel eScript follow.

The CTIData object type contains a field/value set and can be used for two purposes:

- To retrieve the information about the current call (ANI, DNIS, and so on)
- To pass the parameters and user data to the call-control commands

This object is conceptually similar to Dictionaries for Perl, or Maps for MFC.

Field names are specific to the CTI middleware you are using. The event data fields are specific to the particular CTI integration and are fully described in the command tables for your supported CTI middleware found in Chapter 12, "Using Siebel CTI Connect." Also refer to documentation from the middleware vendor.

Clear

```
CTIData.Clear
```

Empties the CTIData object by deleting all fields.

GetFieldValue

```
CTIData.GetFieldValue field_name -> field_value
```

Retrieves the value of the specified field.

SetFieldValue

```
CTIData.SetFieldValue field_name, field_value
```

Adds the field/value pair to the read-only copy of the CTIData object. If a field with the same name is already present, the previous value is overwritten.

GetCount

```
CTIData.GetCount -> return_value
```

Returns the number of the fields in the CTIData object.

GetFieldAt

```
CTIData.GetFieldAt index -> field_name
```

Returns the name of a field by its index location. The GetCount and GetFieldAt functions can be used to enumerate all fields contained currently in the CTIData object.

RemoveField

```
CTIData.RemoveField field_name
```

Removes (deletes) the field by name.

Script Example: Making a Call

This section shows example Siebel VB or Siebel eScript scripts that use CTI objects to make a call and shows how to attach such a script to a communications command.

For more information about defining communications events and commands, see Chapter 4, "Configuring Session Communications," and Chapter 5, "Configuring Events and Commands."

Siebel VB Script Example

The following is an example of a Siebel VB script that uses CTI objects to make a call.

```
Function NewMakeCall(dim Phonenum as string) as Integer

   dim CTI as CTIService
   dim Data as CTIData
   dim Phonenum as string

   set CTI = TheApplication.GetCTIService
   set Data = CTI.GetCurrentCallData
   Phonenum = Data.GetFieldValue("PhoneNumber")

   CTI.InvokeCommand "MakeCall", Phonenum

NewMakeCall = OperationComplete

End Function
```

Siebel eScript Script Example

The following is an example of a Siebel eScript script that uses CTI objects to make a call.

```
function NewMakeCall()
{
   var CTI;
   var Data;
   var Number;

   CTI = TheApplication().GetCTIService();
   Data = CTI.GetCurrentCallData();
   Number = Data.GetFieldValue("PhoneNumber");

   CTI.InvokeCommand("MakeCall", Number);

   return ContinueOperation;
}
```

Defining Communications Command to Invoke the Script

The example NewMakeCall script, whether Siebel VB or Siebel eScript, can be invoked when the user clicks the Make Call button in the communications toolbar. To specify this behavior, you attach the script to a particular command you define in the Administration - Communications screen.

Define the command and its associated command data definition to be similar to the example command MakeCalltoPhone for Siebel CTI Connect, shown in Table 139 and Table 140 on page 498.

Table 139. Command: MakeCalltoPhone

Parameter Name	Parameter Value
Description	Make Call to "{@Phone}"
Script	NewMakeCall

Table 140. Command Data: MakeCalltoPhone

Parameter Name	Parameter Value
AttachContext	TRUE
Param.CallNotifyText	Call from {@UserName}...
Param.PhoneNumber	{@Phone:PhoneTypeLookup}
RequiredField.@Phone	?*
ScriptParam.PhoneNumber	{@Phone:PhoneTypeLookup}

You specify the script name (NewMakeCall in this example) as the value for the parameter called Script in the command entry. Then you define parameters for the script in the associated command data entry, using the form ScriptParam followed by a period (.) and the parameter name (PhoneNumber in this example).

In this example, when an agent clicks the Make Call button in the communications toolbar, the command MakeCalltoPhone is executed, which in turn invokes the script NewMakeCall. A value is passed to the script for its PhoneNumber parameter, based on the value yielded by macro expansion. Each script parameter is passed to the script as a key/value pair.

The example script displays a message box indicating that a script is being invoked and that a call is being initiated to a particular number; it then dials the number in the edit field in the communications toolbar.

For more information about macro expansion and about the @Phone macro, see "Using Macro Expansion for Character Fields" on page 181 and "Macros for Parameter Values" on page 182.

NOTE: A script invoked from a communications command overrides and does not invoke the device command defined within the command.

Script Example: Handling an Incoming Call

This section shows example Siebel VB or Siebel eScript scripts that use CTI objects to handle an incoming call and shows how to attach such a script to a communications event.

For more information about defining communications events and commands, see Chapter 4, "Configuring Session Communications," and Chapter 5, "Configuring Events and Commands."

Siebel VB Script Example

The following is an example Siebel VB script that uses CTI objects to handle an incoming call.

```
Function HandleCallUsingANI(arg1 As String, arg2 As String) As Integer

TheApplication.TraceOn "c:\trace.log", "Allocation", "All"

Dim CTI As CTIService
Dim Data As CTIData
Dim ANI As String

Set CTI = TheApplication.GetCTIService
Set Data = CTI.GetCurrentCallData

ANI = Data.GetFieldValue("ANI")

CTI.NotifyEventHandlerBlocking

TheApplication.Trace "ANI = " & ANI
TheApplication.Trace "arg1 = " & arg1
TheApplication.Trace "arg2 = " & arg2

TheApplication.TraceOff

HandleCallUsingANI = ContinueOperation
End Function
```

Siebel eScript Script Example

The following is an example Siebel eScript script that uses CTI objects to handle an incoming call.

```
function HandleCallUsingANI(arg1, arg2)
{
   TheApplication().TraceOn("c:\\trace.log", "Allocation", "All");

   var CTI = TheApplication().GetCTIService();
   var Data = CTI.GetCurrentCallData();
   var ANI = Data.GetFieldValue("ANI");

   TheApplication().Trace("ANI = " + ANI)
   TheApplication().Trace("arg1 = " + arg1)
   TheApplication().Trace("arg2 = " + arg2)

   TheApplication().TraceOff()

   return ContinueOperation;
}
```

Defining the CTI Event to Invoke the Script

The example HandleCallUsingANI script, whether Siebel VB or Siebel eScript, can be invoked when the user clicks the Answer Call button. To specify this behavior, you attach the script to a particular event response you define in the Administration - Communications screen.

Define the event response to be similar to the example InvokeVBOnAnswer for Siebel CTI Connect, shown in Table 141. The example event handler that invokes this event response has the Device Event field set to the appropriate device event for the driver, such as TpAnswered for Siebel CTI Connect.

Table 141. Event Response: InvokeVBOnAnswer

Parameter Name	Parameter Value
UseCtxData	TRUE
Script	HandleCallUsingANI
ScriptParam.arg1	test1
ScriptParam.arg2	test2

Using the same techniques, you can write a script that will query the Siebel Database using the information about the current call and display the data by navigating to the corresponding Siebel view.

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W