Siebel Deployment Planning Guide

Version 8.0, Rev. A

July 2011

ORACLE

Contents

Index

1 What's New in This Release

What's New in Siebel Deployment Planning Guide, Version 8.0, Rev. A

Table 1 lists changes described in this version of the documentation to support release 8.0 of Siebel Business Applications for Oracle.

Table 1. New Product Features in Siebel Deployment Planning Guide, Version 8.0, Rev. A

Topic	Description
"About Siebel Server Load Balancing" on page 19	Revised topic. Removed content applicable only to earlier versions of Siebel Business Applications.
"About Siebel Connection Broker" on page 22	Revised topic. Added a description of the parameter `ConnForwardAlgorithm`. This parameter requires that you install Siebel Business Applications version 8.0.0.9 or later.
"How Service Failures Affect the Siebel Deployment" on page 45	Revised topic. Restructured content for clarity.
"About Third-Party Server Clustering Products"	Removed topic. Customers are advised to obtain information about clustering products from the applicable vendors.
Chapter 8, "Application-Level Deployment Planning"	Revised topics in chapter. Removed content applicable only to earlier versions of Siebel Business Applications and content that described products that are no longer provided.

What's New in Siebel Deployment Planning Guide, Version 8.0

Siebel Deployment Planning Guide provides important information for planning deployments of Oracle's Siebel Business Applications software.

No new features have been added to this guide for this release. This guide has been updated to reflect only product name changes.

2 Siebel Architecture Overview

This chapter includes the following topics:

Building Blocks of a Siebel Deployment

Figure 1 on page 10 shows an example of the elements in a Siebel deployment. A brief description of these elements appears in Table 2 on page 11, to help you understand the Siebel architecture.

The current release supports only certain specific database and operating system platforms, as well as certain combinations of them. For a list of all operating systems and RDBMS products supported by this release, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

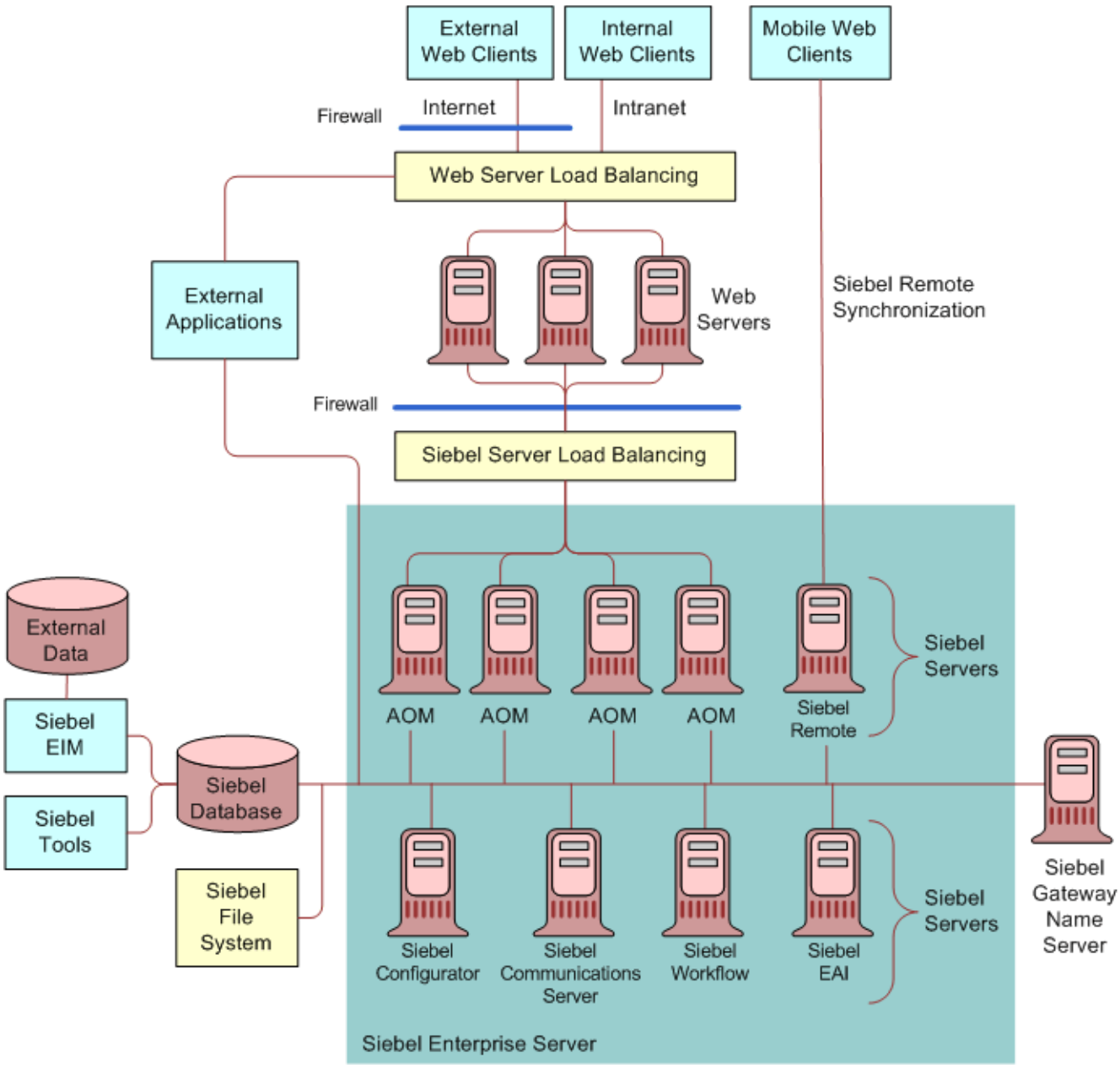

Figure 1. Example of a Siebel Deployment

Table 2. Siebel Deployment Elements

Entity	Description
Siebel Web Clients	Includes the following client types: ■ Siebel Web Client ■ Siebel Mobile Web Client ■ Siebel Developer Web Client (limited support) ■ Siebel Wireless Client ■ Siebel Handheld Client
Siebel Web Server Extension (SWSE)	Software installed on third-party Web server, where virtual directories for Siebel applications are created. SWSE identifies requests for Siebel data and forwards them to the Siebel Servers. Receives data from Siebel Servers and helps format it into Web pages for Siebel clients.
Siebel Server load balancing	The two options for Siebel Server load balancing are: ■ Siebel native load balancing ■ Third-party HTTP load balancers Siebel native load balancing is part of the Siebel Web Server Extension (SWSE). When you configure the SWSE, the wizard prompts you for information about configuring load balancing. Figure 1 on page 10 shows a third-party HTTP load balancer.
Siebel Enterprise Server	A logical grouping of Siebel Servers that connect to one database. Allows management of Siebel Servers as a group.
Siebel Servers	Application servers that provide both user services and batch mode services to Siebel clients or other components.
Siebel Gateway Name Server	Stores Siebel Enterprise Server and Siebel Server configuration and status information.
Siebel Database	Stores database records. Includes third-party RDBMS software and Siebel tables, indexes, and seed data.
Siebel File System	Shared file system directory or directories storing data and physical files used by Siebel clients and Siebel application components.

Table 2. Siebel Deployment Elements

Entity	Description
Siebel deployment	All of the physical and logical elements required to deploy Siebel applications, including: ■ Siebel Database ■ Siebel Gateway Name Server ■ Siebel Enterprise Server ■ Siebel Servers ■ Siebel Web Server Extension (and Web server) ■ Siebel clients and Web browsers ■ Related components such as third-party HTTP load balancers
Siebel Enterprise Integration Management (EIM) Siebel Enterprise Application Integration (EAI)	These modules allow importing and exporting of data from other databases to the Siebel Database, and various other integration services.
Siebel Tools	An integrated development environment for configuring Siebel applications. You use Siebel Tools to modify standard Siebel objects and create new objects to meet your organization's business requirements. For example, you use Siebel Tools to extend the data model, modify business logic, and define the user interface.

About Siebel Web Clients and Siebel Web Server Extension

This topic describes the types of Siebel Web Clients. This term sometimes refers to all the browser-based clients and sometimes to one specific client type. The Siebel Web Server Extension is also discussed.

NOTE: Siebel Web Clients running in high interactivity mode use ActiveX controls and JavaScript, and require the Java Runtime Environment (JRE). These items can be downloaded to the browser automatically. For more information about client and browser requirements, see *Siebel System Administration Guide* and *Siebel System Requirements and Supported Platforms* on Oracle Technology Network. See also "About Standard and High Interactivity Modes" on page 73.

Siebel Web Client

Siebel Web Client runs in a standard browser on the end user's client computer. The browser connects through a Web server to the Siebel Server, which executes business logic and accesses data from the Siebel Database. Only the user interface layer of the Siebel Business Applications architecture resides on the user's computer.

Other considerations about the Siebel Web Client are as follows:

- **Installed software.** No additional application software is required on the client. At minimum, the client requires only a Web browser.
- **Application connection.** This is the connection through a Web server to the Siebel Server. Applications run on the Siebel Server and forward pages to the client. Applications display in a standard Web browser on the end user's client computer, such as a connected laptop or desktop.
- **Database connection.** This is the connection through the Siebel Server to the Siebel Database. No Siebel Database or database client is installed on the client.

Siebel Mobile Web Client

Siebel Mobile Web Client includes the following:

- **Installed software.** Windows-based software containing Siebel applications and related services is installed on each client. The client also requires a Web browser.
- **Application connection.** Applications run on each client. Applications display in a standard Web browser on the end user's client computer, such as a laptop.
- **Database connection.** A local database and local Siebel File System are installed on each client. Applications access the local database.

  Users periodically synchronize the local database and Siebel File System with a remote Siebel Database and Siebel File System. Users synchronize data using Siebel Remote components of the Siebel Server. Mobile users synchronize with the remote Siebel Database and Siebel File System without going through the Web server or any other Siebel Server component.

Siebel Developer Web Client

NOTE: This client type is supported for limited administration and troubleshooting purposes only.

Siebel Developer Web Client includes the following:

- **Installed software.** Windows-based software containing Siebel applications and related services is installed on each client. The client also requires a Web browser.
- **Application connection.** Applications run on each client. Applications display in a standard Web browser on the end user's client computer, such as a connected laptop or desktop.
- **Database connection.** A direct connection is required to the Siebel Database. Appropriate database client software must be installed on the client. This client can connect directly to the Siebel File System or connect through the File System Manager server component.

Siebel Wireless Client

The Siebel Wireless Client is a modified Siebel Web Client that runs on a mobile device. Users can view, edit, and create records in the Siebel Database through a wireless connection between a mobile device and a Web server. An Internet-enabled mobile phone, personal digital assistant, or other wireless device communicates with a wireless gateway server, which translates messages for display in a browser on the device.

Several different types of wireless browsers are supported. For a list of Siebel Business Applications that support wireless access, see *Siebel Wireless Administration Guide*. For a list of supported browsers, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

Siebel Handheld Client

The Siebel Handheld Client is a streamlined version of the Mobile Web Client that runs on a handheld device. It includes only the functionality required by field technician end users. The Siebel Handheld Client supports the same data relationships, the same configuration in Siebel Tools, and much of the same functionality as the Siebel Mobile Web Client. Siebel Handheld runs on devices that support the Windows CE operating system.

Siebel Web Server Extension (SWSE)

The Siebel Web Server Extension (SWSE) is a plug-in for supported third-party Web servers. It identifies requests for Siebel application data coming from Web clients and flags these requests for routing to a Siebel Server. When information is sent from the Siebel Server back to the Web client, the SWSE helps complete the composition of the Web page for forwarding to the client.

The SWSE includes the Siebel native load balancing module. This module provides round-robin load balancing for routing requests to Application Object Manager components running on Siebel Servers.

You can install and deploy multiple Language Packs on a single SWSE instance. The Siebel Server and the Web server do not need to be operated in the same language. However, the Siebel Server, the Web server, and all other server components must use the same character set. For details, see the *Siebel Installation Guide* for the operating system you are using and see *Siebel Global Deployment Guide*.

About Siebel Enterprise Server and Siebel Server

This topic describes the Siebel Enterprise Server and Siebel Server, and also Application Object Manager components of Siebel Server.

For information on applicable installation and configuration tasks, see the *Siebel Installation Guide* for the operating system you are using.

Siebel Enterprise Server

The Siebel Enterprise Server is a logical grouping of one or more Siebel Servers that connect to one Siebel Database. You can configure some server parameters at the Enterprise level. Such parameters are inherited by individual Siebel Servers and applicable components. Some parameters may be overridden at the server level or component level.

After initial configuration of the Siebel Enterprise and each Siebel Server using wizards, some subsequent configuration and administration tasks may be performed by one or more administrators using Siebel Server Manager. Server Manager supports both a command-line UI and a GUI.

You use the Siebel Enterprise Server installer for installing Siebel Gateway Name Server, Siebel Server, Siebel Database Configuration Utilities, and Siebel EAI Connectors.

Siebel Server

Each Siebel Server functions as an application server and is composed of server components. Each server component performs a defined function.

Server components or groups of components determine what applications and services a Siebel Server supports. Components run in one of several modes:

- **Interactive mode.** Interactive mode components start tasks automatically in response to user requests. Interactive tasks run until the user ends the session. Examples of interactive components include the Application Object Managers (AOMs) and the Synchronization Manager.
- **Background mode.** Background mode components handle background processing tasks. Typically, background tasks are called by interactive tasks. Background tasks run until they are explicitly shut down. Examples of background components include Transaction Router and Workflow Monitor Agent.
- **Batch mode.** Batch mode components handle processing of asynchronous work requests. When the task is complete, the component exits. Examples of batch components are Database Extract and Enterprise Integration Manager (EIM).

Many of the Siebel Server components can operate on multiple Siebel Servers simultaneously, allowing Siebel applications to scale across many server machines to support large numbers of users.

Other Siebel Server components provide additional functionality, including the following:

- Siebel Mobile Web Client synchronization
- Integration with legacy or third-party data
- Automatic assignment of new accounts, opportunities, service requests, and other records
- Workflow management
- Document generation

Siebel Connection Broker (SCBroker)

The Siebel Connection Broker component provides load balancing of connection requests to multiple Application Object Manager (AOM) threads or processes running on the same Siebel Server.

Siebel Server Implementation

The Siebel Server runs as a system service under Windows and as a process under UNIX. This system service or process monitors and controls the state of all server components on that Siebel Server. Each Siebel Server is one instantiation of the Siebel Server system service or process within the current Siebel Enterprise Server.

Interactive and batch components can be configured to run as multiple processes or in some cases as multithreaded processes. Application Object Manager (AOM) components (which are interactive) can run as both multiple processes and multiple threads for each process. Background mode components can run as multiple processes only.

For information on administering the Siebel Server system service or process, see *Siebel System Administration Guide*.

Language Pack Installation

It is strongly recommended to install the same set of languages on each physical server in your Siebel Enterprise. However, in the Siebel Server Configuration Wizard you can deploy different languages on different Siebel Servers, as needed. For details, see the *Siebel Installation Guide* for the operating system you are using. See also *Siebel Global Deployment Guide*.

Application Object Manager

One of the most important types of server components is the Application Object Manager (AOM). These server components always run in interactive mode. They process user requests and are application- or service-specific. For example, the Siebel Call Center component group contains the Call Center Object Manager, one for each language deployed on the Siebel Server. This AOM provides the session environment in which this application runs.

Internally, each AOM also contains a data manager and the Siebel Web Engine. When an AOM receives a user request to start an application, the AOM follows this procedure:

- The business object layer starts an application user session, processes any required business logic, and sends a data request to the data manager.
- The data manager creates an SQL query and forwards it the Siebel Database.
- The data manager receives the data from the database and forwards it to the business object layer for additional processing.
- The business object layer forwards the result to the Siebel Web Engine, which helps create the UI for the data. The Siebel Web Engine then forwards the Web pages to the Siebel Web Server Extension on the Web server.

Application Object Manager Implementation

An Application Object Manager (AOM) server component is implemented as a multithreaded process on the Siebel Server. At run time, a parent process starts one or more AOMs as multithreaded processes, according to the AOM configuration. The terms multithreaded server or MT server are alternative terms for the multithreaded process, which may also be called an AOM process.

Each thread in an AOM hosts tasks that are typically linked to one user session. These threads may be dedicated to particular user sessions, or they may serve as a pool that can be shared by user sessions. For each AOM, a few threads are dedicated to housekeeping functions.

Each AOM task communicates with the Siebel Database, the Web server (through the SWSE), or other components, as follows:

- Communication with the Siebel Database uses ODBC database connections. You can manage and tune database connections for optimal performance. You can optionally configure connection sharing for database connections.
- Communication with the Siebel Web Server Extension uses SISNAPI (Siebel Internet Session API), a Siebel messaging format that runs on top of the TCP/IP protocol. You can configure SISNAPI connections to use encryption and authentication based on Secure Sockets Layer (SSL).
- Communication with other Siebel Enterprise Server components (including other Siebel Servers) also uses SISNAPI.
- The Siebel Connection Broker (SCBroker) on each Siebel Server listens on a static, configurable TCP port for requests coming from the Web server. SCBroker forwards these requests to AOM processes.

For more information about the operation of multithreaded processes for AOM components, see *Siebel System Administration Guide* and *Siebel Performance Tuning Guide*.

About Siebel Gateway Name Server

The Siebel Gateway Name Server serves as the dynamic address registry for Siebel Servers and components. At startup, a Siebel Server within the Siebel Enterprise Server stores its network address in the Gateway Name Server's nonpersistent address registry.

Siebel Enterprise Server components query the Gateway Name Server address registry for Siebel Server availability and address information. When a Siebel Server shuts down, this information is cleared from the address registry.

The Gateway Name Server also includes a persistent file (siebns.dat) containing Siebel Server configuration information, including:

- Definitions and assignments of component groups and components
- Operational parameters
- Connectivity information

As this configuration information changes, such as during the configuration of a Siebel Enterprise or Siebel Server, it is written to the siebns.dat file on the Gateway Name Server.

There can be only one Gateway Name Server installed for each environment in which you have created a Siebel Enterprise. Further, do not share the same Gateway Name Server across development, test, and production environments.

The Gateway Name Server is usually a candidate for high availability using a cluster configuration. If the primary node in the cluster fails, the second machine in the Gateway Name Server cluster takes over, minimizing potential downtime.

For information on installation and configuration tasks associated with Siebel Gateway Name Server, see the *Siebel Installation Guide* for the operating system you are using.

Language Pack Installation

It is strongly recommended to install the same set of languages on each physical server in your Siebel Enterprise. For details, see the *Siebel Installation Guide* for the operating system you are using. See also *Siebel Global Deployment Guide*.

About the Siebel File System

The Siebel File System is a shared file system directory or set of directories. The Siebel File System stores document files, Siebel Configurator models, Web template definitions, and other files not appropriate for database storage. System user preferences are also stored in the userpref subdirectory of the Siebel File System.

For information about setting up and maintaining the Siebel File System, see the *Siebel Installation Guide* for the operating system you are using and *Siebel System Administration Guide*.

Siebel File System Best Practices

The following list provides suggested best practices when dealing with the Siebel File System:

- All Siebel Servers must have direct access to the Siebel File System. The only exception to this rule is when a server is running a Siebel Document Server only. If the File System Manager (FSM) is disabled on the Document Server, it accesses the Siebel File System through the FSM on another Siebel Server. In all other cases, the Siebel Server must be able to directly access the Siebel File System.
- The Siebel File System can be configured to use multiple directories that may exist on separate devices or partitions. Before you configure the Siebel Enterprise, at least one file system directory must exist that you can designate for use by the Siebel File System.
- There are few strict rules for deploying the Siebel File System. Theoretically, the file system can be hosted on any of the servers in the Siebel Enterprise or on servers in an independent file server farm. For small-to-medium deployments, a common scenario may be to place the Siebel File System on the Siebel Gateway Name Server. Implementations with a large number of attachments may need a dedicated file system server. Consider using a high-speed RAID disk storage system to increase file system throughput.
- When using a large number of files in the Siebel File System (300,000 or more) in a Windows NTFS folder, disable short filename generation. For more information, see the Microsoft technical document about how NTFS works.
- During normal operation of Siebel Business Applications software, it is likely orphaned files will be stored in the Siebel File System and that orphaned records will exist in the Siebel Database. Periodically run the SFSCLEANUP utility to remove orphaned files from the Siebel File System. This utility is located in the bin subdirectory within the Siebel Server root directory.

About Siebel Server Load Balancing

Load balancing distributes workload across multiple Siebel Servers. Each server runs an instance of the service you want to load balance. Load balancing also provides failover. If one server fails, then requests are automatically routed to the remaining servers.

You can use load balancing when the Siebel Enterprise Server has two or more Siebel Servers that are not clustered. Load balancing is the preferred method for providing high availability for the following server components:

- Application Object Managers (AOMs)
- Siebel Configurator (uses own load balancing method)
- Siebel EAI (whenever possible)

See also "Choosing a Load Balancing Method" on page 33.

Two methods are available for implementing Siebel Server load balancing for AOMs:

- **Siebel native load balancing.** With Siebel native load balancing, a load balancing module is built into the Siebel Web Server Extension (SWSE). This module provides software-based load balancing for Siebel Servers. You can use Siebel native load balancing instead of third-party HTTP load balancers.

  (On each Siebel Server, Siebel Connection Broker (SCBroker) provides intraserver load balancing. SCBroker distributes connection requests across multiple instances of Application Object Manager processes running on the same server. For details, see "About Siebel Connection Broker" on page 22.)

- **Third-party HTTP load balancing.** Oracle has validated third-party, hardware-based HTTP load balancers for use in a Siebel deployment.

  Although Siebel applications are designed to work with standard, third-party HTTP applications, customers should perform compatibility testing before using a nonvalidated load balancer.

NOTE: For help with configuring Siebel software to support load balancing, create a service request (SR) on My Oracle Support. Alternatively, you can phone Global Customer Support directly to create a service request or get a status update on your current SR. Support phone numbers are listed on My Oracle Support. For configuration and troubleshooting information for third-party HTTP load balancers, contact the third-party vendor directly. See also 477835.1 (Article ID) on My Oracle Support. This document was previously published as Siebel Technical Note 540.

Figure 2 on page 20 shows an example of third-party HTTP load balancing.

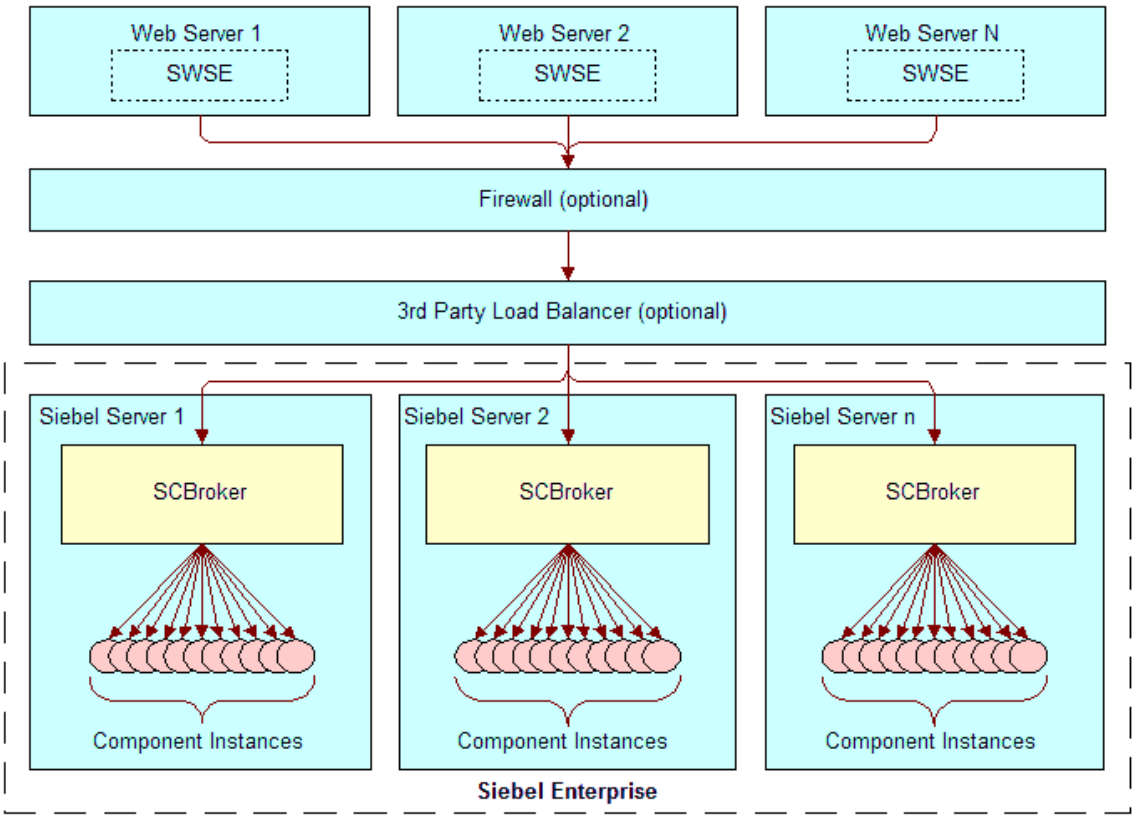

Figure 2. Example of Third-Party HTTP Load Balancing

About Siebel Internet Session Network API

Siebel Internet Session Network API (SISNAPI) is a Siebel proprietary message-body format running on top of TCP/IP. SISNAPI is used for communications between the Web server, Siebel Gateway Name Server, and Siebel Servers.

When a client request comes to the Web server, the Siebel Web Server Extension (SWSE) intercepts the request and forwards it in SISNAPI format. The SISNAPI message-body format has the following parts:

- HTTP header
- Object Manager method name
- Method arguments as key-value pairs

HTTP Header

When the Siebel Web Server Extension (SWSE) requests a new connection, the initial packets of the first SISNAPI message contain an HTTP header. This header includes a Uniform Resource Locator (URL) that provides routing information to the Siebel Enterprise Server, Siebel Server, and server component. Third-party HTTP load balancers use routing rules to parse the URL and route the message to the correct Siebel Server.

Connection Multiplexing

SISNAPI TCP/IP connections are specific to an Application Object Manager on one Siebel Server. Before opening new connections, the system checks to see if an existing connection is available. If so, the system uses the existing connection. Once the connection is established, it remains open for use by subsequent messages in the session or to be reused by other sessions. For more information about connection multiplexing, see *Siebel Performance Tuning Guide*.

Secure Sockets Layer (SSL)

Secure Sockets Layer (SSL) encryption and authentication can be configured for SISNAPI connections. For more information, see *Siebel Security Guide*. See also the *Siebel Installation Guide* for the operating system you are using.

User Request Types

The Siebel Web Server Extension (SWSE) generates three types of user requests. Each request type creates a new connection to a Siebel Server through the load balancer: initial request, retry request, and reconnect request. The Siebel native load balancing module in the SWSE recognizes these request types and automatically routes them correctly. If you use a third-party HTTP load balancer, you must configure routing rules to handle these requests.

- **Initial request.** The SWSE generates this request to start a new user session as follows:
  - The SWSE receives the request to start a user session.
  - The SWSE creates the SISNAPI message. The HTTP header specifies the Siebel Enterprise Server and the desired server component. The message does not specify a Siebel Server name. The SWSE forwards the message to a third-party HTTP load balancer, if installed.
  - The load balancer parses the URL and compares it to routing rules that have been entered in the load balancer.
  - The load balancer uses these routing rules to route the message to a Siebel Server specified in the routing rule. If no SISNAPI connection exists to the Siebel Server, a new one is created.
  - The Siebel Server receives the message and creates a new user session. The Siebel Server forwards address information back to the Web server.
  - The Web server creates a cookie containing the address information. The Web server receives the cookie information in subsequent session requests. SWSE includes this information in the SISNAPI HTTP header.
  - The load balancer receives subsequent messages and forwards them directly to the specified Siebel Server and server component through the open SISNAPI connection.

- **Retry request.** If a server rejects an initial request, the request is routed back to the SWSE and the following occurs:
  - The SWSE modifies the URL contained in the HTTP header by appending the letters *RR* to it.
  - The SWSE forwards the message to the load balancer, if installed.
  - The load balancer applies the routing rule that has been entered for RR messages. Typically, this is a round-robin routing rule that forwards the message to another Siebel Server.
- **Reconnect request.** The SWSE generates a reconnect request when it receives a user request for an existing user session that does not have a SISNAPI connection. The SWSE uses the session cookie information to include the server address in the SISNAPI HTTP header.

  The reconnect request opens a new SISNAPI connection. Reconnect requests can occur for several reasons:
  - The SISNAPI connection was opened by Web Server 1, but a Web server load balancer routes subsequent session messages to Web Server 2, which does not have an existing connection.
  - The SISNAPI connection timeout is exceeded and the connection is closed.
  - The network environment closes the connection, for example due to a firewall time-out.

About Siebel Connection Broker

The Siebel Connection Broker (SCBroker) server component provides intraserver load balancing. SCBroker distributes server requests across multiple Application Object Manager (AOM) processes running on a Siebel Server.

SCBroker listens on a configurable, static port for new requests. When a new request is received, it forwards the request to the AOM process with the least number of running tasks, or forwards the request to another AOM process in round-robin fashion. A user session is created on this AOM. Thereafter, the requests that apply to this session are directly sent to the AOM process hosting the session.

SCBroker is enabled by default and has several parameters:

- **PortNumber.** Sets the port number on which SCBroker listens. The default is 2321, but you can change the port number.
- **DfltTasks.** Sets the default number of processes for SCBroker. The recommended value is 2.
- **MaxTasks.** Sets the maximum number of processes for SCBroker. The recommended value is 2. This value cannot be less than DfltTasks.
- **AutoRestart.** Default is On. If SCBroker terminates abnormally, this setting allows it to restart automatically. Setting this parameter to Off or False is not recommended.

- **ConnForwardAlgorithm.** The connection forwarding algorithm is the routing scheme for SCBroker to use when routing intraserver requests to AOM processes. Possible values are LL (least-loaded) and RR (round-robin). SCBroker uses the least-loaded algorithm by default.
  - The least-loaded scheme routes each request to the AOM process with the fewest number of tasks.
  - The round-robin scheme routes each request to the next AOM process, cycling through all AOM processes.

  NOTE: Support for the ConnForwardAlgorithm parameter requires Siebel Fix Pack version 8.0.0.9 or later.

  The round-robin algorithm does not take the number of running tasks for each AOM process into account. It simply forwards each request to the next AOM process among the available processes. The least-loaded algorithm considers only the current number of tasks for each AOM process that is running.

  In many circumstances, how an individual connection request is forwarded may be the same for either forwarding algorithm. Actual forwarding behavior depends on these factors:

  - The current number of AOM processes.
  - The current number of tasks for the AOM component and for each AOM process.
  - Patterns of requests for new tasks and patterns of threads being freed up through logouts.
  - AOM parameter settings controlling the maximum number of tasks and the minimum and maximum number of AOM processes. The applicable parameters are:
    - Maximum Tasks (MaxTasks)
    - Minimum MT Servers (MinMTServers)
    - Maximum MT Servers (MaxMTServers)

    For detailed information about calculating the settings for MaxTasks, MaxMTServers and MinMTServers, see *Siebel Performance Tuning Guide*.

  NOTE: Using the round-robin routing scheme instead of the least-loaded scheme can enhance availability in some cases where multiple AOM processes are running and one process is nonresponsive. Eventually, the nonresponsive process will have the least number of tasks. With the least-loaded scheme, all new tasks would be routed to this unresponsive process. With the round-robin scheme, however, one request would be routed to this process for each “round” of requests routed to all participating AOM processes.

  Sometimes an individual connection request is not forwarded to an existing AOM process but instead causes a new AOM process to start. This can occur with either forwarding algorithm, when each existing AOM process has reached its theoretical maximum number of tasks, based on dividing the maximum number of tasks by the maximum number of AOM processes. A new process starts only if the maximum number of tasks and the maximum number of AOM processes have not yet been reached.

  For more information about applicable parameters, see also *Siebel System Administration Guide* and *Siebel Performance Tuning Guide*.

- **ConnForwardTimeout.** The connection forward time-out determines how long SCBroker will wait for an AOM process to accept a request. The default is 500 milliseconds. This time-out minimizes wait time when SCBroker forwards a connection request to an AOM process, and the request cannot be accepted.

  If a time-out occurs, SCBroker reports an error back to the Web server. The SWSE then modifies the request by appending *RR* to the Siebel application URL. The SWSE then retries the request.

  - If you are using Siebel native load balancing, the SWSE forwards the request to an AOM process on another Siebel Server. The Siebel Server is selected using a round-robin algorithm.
  - If you are using a third-party HTTP load balancer, the SWSE forwards the request to the load balancer. The load balancer forwards the request to an AOM process on another Siebel Server. The Siebel Server is selected using a round-robin algorithm.

- **ConnRequestTimeout.** The connection request time-out determines how long SCBroker will wait for all the packets in an incoming new request. The default is 500 milliseconds. This time-out minimizes SCBroker wait time when TCP/IP requests are incomplete.

  If a time-out occurs, the request is sent back to the Web server in the same fashion as a connection forward time-out.

About Server Request Broker

The Server Request Broker (SRBroker) server component processes both synchronous and asynchronous server requests.

- Synchronous server requests are requests that must be run immediately, and for which the calling process waits for completion.
- Asynchronous server requests are requests for which the calling process does not wait for completion.

SRBroker can run server requests on any server in the Siebel Enterprise. For example, if SRBroker is unable to run a server request on the local Siebel Server because the required component is not enabled, SRBroker finds another Siebel Server that is hosting the required component and runs it there. SRBroker runs by default on all Siebel Servers.

SRBroker decides where to run a server request using the following criteria:

- If the required component is available locally, then SRBroker runs the task locally.
- If the required component is not available locally, then SRBroker identifies any Siebel Servers in the same Enterprise that have the component online. Server requests are submitted to each of these Siebel Servers in turn (that is, using a round-robin algorithm).
- If the required component is not available anywhere in the Enterprise, then the server request will fail.

The SRBroker component helps provide resilient processing. As long as the required component is running on a Siebel Server somewhere in the Enterprise, then the server request can be processed. For more information on resilient processing, see “About Resilient Processing” on page 60.

About Server Request Processor

The Server Request Processor (SRProc) server component processes asynchronous, server-initiated requests. These are requests that are submitted for later execution and that do not require the calling process to wait for the request to complete.

SRProc runs by default on all Siebel Servers. When asynchronous requests are submitted, they are stored in the Siebel Database in the S_SRM_REQUEST table. SRProc periodically checks this table for any requests that are eligible to be run. For a request to be eligible, it must meet all the following criteria:

- The request must be in the correct state (Queued)
- Its start time must have passed
- The target Siebel Server must not be specified or must be the Siebel Server where the requested component is running

If a request is eligible, SRProc will invoke Server Request Broker (SRBroker) to run the request. Therefore, as long as a target Siebel Server is not specified, asynchronous requests will be read by any SRProc task on any Siebel Server.

The SRProc component helps provide resilient processing for server-initiated tasks. As long as an SRProc task is running somewhere in the Siebel Enterprise, the request will be processed. For more information on resilient processing, see "About Resilient Processing" on page 60.

About Siebel Enterprise Application Integration

Siebel Enterprise Application Integration (EAI) provides components for integrating Siebel Business Applications with external applications and technologies. It is designed to work with third-party solutions such as those from IBM, CrossWorlds, TIBCO, Vitria, SeeBeyond, webMethods, and others.

Siebel EAI provides bidirectional real-time and batch solutions for integrating Siebel applications with other applications.

Siebel EAI is designed as a set of interfaces that interact with each other and with other components within the Siebel application. These interfaces are compatible with IBM MQSeries; Microsoft MSMQ; Java and J2EE; XML; HTTP; and many other standards.

Siebel EAI interfaces do the following:

- Allow a flexible service-based architecture built on top of configurable messages using XML and other formats
- Expose internal Siebel objects to external applications
- Take advantage of prebuilt adapters and enterprise connectors, and are compatible with third-party adapters and connectors
- Allow for data transformation
- Integrate external data through virtual business components (VBCs)

- Provide a graphical business process designer, programmatic interfaces, and a high-volume batch interface

For more information about Siebel EAI, see *Overview: Siebel Enterprise Application Integration*.

About Siebel Enterprise Integration Manager

Siebel Enterprise Integration Manager (EIM) manages the bidirectional exchange of data between the Siebel Database and other corporate databases. This exchange is accomplished through intermediary tables called EIM tables. (In earlier releases, these tables were known as interface tables.) The EIM tables act as a staging area between the Siebel Database and other databases.

You must use Siebel EIM to perform bulk imports, exports, updates, and deletes. Using native SQL to load data directly into Siebel base tables (the tables targeted to receive the data) is not supported.

For more information about Siebel EIM, see *Siebel Enterprise Integration Manager Administration Guide*. See also *Siebel Performance Tuning Guide*.

About Siebel Tools

Siebel Tools is a Windows-based, integrated environment for configuring Siebel applications. You use Siebel Tools to modify standard Siebel objects and create new objects to meet your organization's business requirements. For example, you use Siebel Tools to extend the data model, modify business logic, and define the user interface. Siebel Tools is also a way to integrate programs written using Siebel scripting languages.

A standard Siebel application provides a core set of object definitions that you can use as a basis for your own tailored application. Siebel Tools object definitions are grouped into four layers, each with a different purpose:

- **Physical user interface (UI) layer.** Templates and tags that render the UI in the client.
- **Logical user interface objects layer.** Presentation of data (UI).
- **Business objects layer.** Objects that extract defined information from the database or provide a defined service.
- **Data objects layer.** Database interface objects and table definitions.

Object types in a given layer depend on definitions in the next lower layer, and are insulated from other layers in the structure. You can make certain kinds of changes to a Siebel application without changing the underlying database structure. Similarly, you can extend the Siebel Database schema without affecting the Siebel application. In many cases, configuration changes are made in concert across multiple layers in order to achieve the desired business functionality.

For additional information about Siebel Tools, see *Configuring Siebel Business Applications* and *Using Siebel Tools*. For Siebel Tools installation instructions, see the *Siebel Installation Guide* for the operating system you are using.

Example of User Request Flow in a Siebel Deployment

Figure 3 on page 27 illustrates how a user request is processed within the Siebel Business Applications architecture. In the diagram, there are two types of load balancing:

- **Web server load balancing.** Web client requests are forwarded through a load balancer to multiple Web servers.
- **Siebel Server load balancing.** Web servers forward user requests to a third-party HTTP load balancer for distribution to Siebel Servers. If Siebel native load balancing is used, a request is forwarded directly to a Siebel Server.

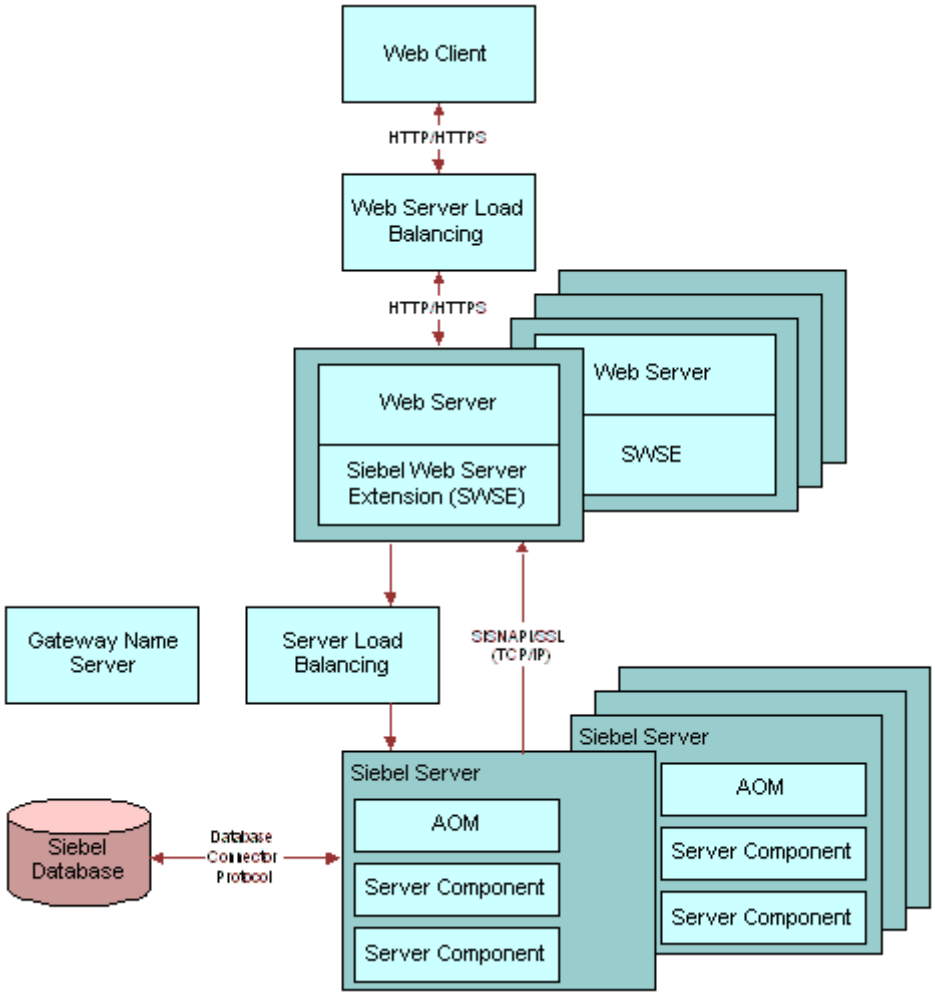

Figure 3. Generic User Request Flow in Siebel Business Applications

A typical Siebel client request flows from the user's Siebel Web Client through the system and back again, following the general flow outlined below:

1. A user performs an action that initiates a request. For example, the user clicks a link in the Site Map to navigate to a particular view. The request is generated by the Web browser and Siebel Web Client framework.
2. The request goes through the network, using an existing or new HTTP connection. The request may go through a network router, proxy server, cache engine, or other mechanism.
3. If present, Web server load balancing software evaluates the request, determines the best Web server to receive the request, and then forwards the request to a Web server.
4. The Web server receives the HTTP request, determines that it is a Siebel application request, and forwards the request to the Siebel Web Server Extension (SWSE) installed on the Web server.
5. The Siebel Web Server Extension parses the HTTP message and generates a SISNAPI message, based on the content of the HTTP message. SWSE also parses the incoming cookie or URL to obtain the user session ID. For Siebel Server load balancing:
   - If you are using Siebel native load balancing, SWSE forwards the request to a Siebel Server in round-robin fashion.
   - If you are using a third-party HTTP load balancer, SWSE forwards the request to the load balancer. The load balancer uses user-configured routing rules to forward the request to a Siebel Server.
6. On the Siebel Server, an AOM receives and processes the SISNAPI message. If a database query is needed to retrieve the information, the AOM formulates the SQL statement and sends the request to the Siebel Database over a database connection.

   The database request uses a protocol format that is specific to the database connector.
7. The database executes the SQL statement and returns data back to the AOM. The AOM forwards the message to the Web server that originated it. If you are using a third-party HTTP load balancer, the message may go through the load balancer before reaching the Web server.
8. The SWSE on the Web server receives the SISNAPI message and translates it back to HTTP. It then forwards the HTTP message to the Web server. The message is now in the form of Web page content.
9. The Web server load balancer, if present, then forwards the Web page content through the original HTTP connection to the end user's Web browser.
10. The Web browser and the Siebel Web Client framework process the return message and display it to the end user.

3 Siebel Infrastructure Planning

This chapter explains how to plan the infrastructure of your Siebel deployment. It includes the following topics:

Process of Infrastructure Planning

The tasks in this process will help you determine the Siebel infrastructure requirements for a production environment. Along with a production environment, you should also plan for a development environment and a test environment.

Use the following steps to plan your Siebel deployment infrastructure:

Determining How the System Will Be Used

This topic is a step in "Process of Infrastructure Planning" on page 29.

This infrastructure planning step identifies what tasks users will perform when using the system.

Examples are completing a customer order, adding a contact, and creating a quote. Later in the planning process, you will map these tasks to specific Siebel applications and functions.

To determine how the system will be used

1 Define user types.

   For each business location, identify user types. Organize this list by the functional areas that participate in key business processes.

   For example, you have a call center in Denver. One of your key business processes is order creation. Two of the functional areas that participate in this business process are call center agents and product line administrators. These are two user types.

   Include application developers and integrators, system administrators, and application administrators in your list of user types.

2 Identify tasks by user type.

   For each user type, identify all the tasks the users will perform using the system. Start with each key business process and map its steps to tasks. Doing this step helps you verify that your business processes are being correctly automated.

3 Identify background tasks.

   If your business operation includes background tasks, list these as well. Background tasks are those that the system performs, rather than users. These include batch processing of business data and automated workflow processes.

4 Estimate transaction volumes.

   For each user task, estimate average and maximum daily transaction volumes. For example, in your Denver call center there are 25 call center agents. Transaction records indicate that each agent will complete an average of 12 customer orders per day and a maximum of 20 per day. Below is an example of how you would list transaction volumes for the Denver call center.

User Type	Number	Task	Average Volume per Day	Maximum Volume per Day
Call Center Agents	25	Inbound customer order	300	500

Defining Data Flows and Integration Requirements

This topic is a step in "Process of Infrastructure Planning" on page 29.

This infrastructure planning step identifies how data will flow to and from the Siebel deployment.

An example of a key data flow would be customer contact updates that originate at several call centers and flow to the master customer contact database at a headquarters location.

This step identifies where the master copy of data records will reside. It also identifies the data interchange requirements for applications.

To identify data flows and transaction volumes

1. Identify business data.

   List the types of business data that will flow through the system. Examples of business data are orders, customer contacts, product line information, and quotes.

2. Identify business data sources.

   For each type of business data, list the user types or business activities that can originate or update the business data. Group user types or business activities by business location.

3. Analyze data requirements of legacy applications.

   Identify all existing applications that will send or receive data from the Siebel deployment. Determine data volumes and group them by location.

4. Identify data formats and transformations.

   For each legacy application that will send or receive data from the Siebel application, identify the required data formats. Specify in detail all data transformation requirements.

5. Map the data flows.

   Create a model that shows all major business data flows. The model should include all data sources, repositories, and key business applications.

Figure 4 on page 32 shows an example of a model of a data flow. The example shows a call center running the Siebel Communications application. The company maintains an ERP database and a phone number database separately from the Siebel Database, which contains customer information.

Siebel Communications sends XML messages containing customer orders to the order fulfillment application, and receives order fulfillment status through an inbound HTTP adapter. Siebel Communications also queries the phone number database for available phone numbers in real time. The phone number database then receives assigned phone numbers from the Siebel Database using Siebel EIM.

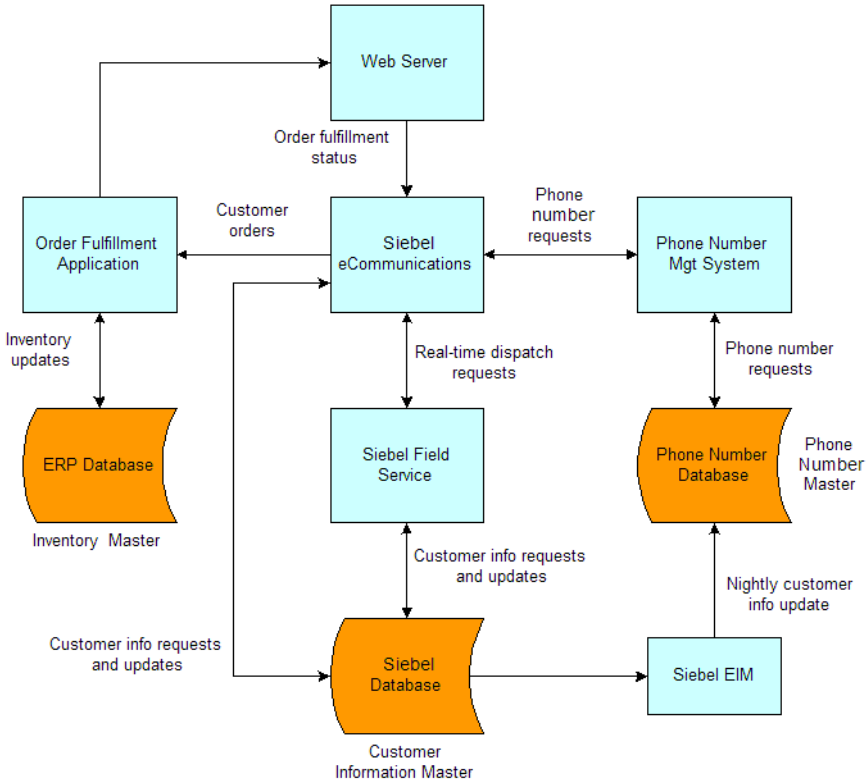

Figure 4. Example of a Data Flow Model

Determining Database Requirements

This topic is a step in "Process of Infrastructure Planning" on page 29.

This infrastructure planning step identifies database requirements for the Siebel deployment.

You should have already identified the types of data that will be stored in the Siebel Database. This step maps that data to key database characteristics. Doing this step helps you estimate database size requirements and expected growth. Begin by defining general requirements:

- What are the types of records that will be stored? What specific fields will each record contain?
- What is the volume of each record? How many records of each type will be processed each hour? Each day? Each year? Group this information by business location.
- Determine how record volumes map to specific Siebel tables. For help with mapping records to Siebel tables, contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.
- How much space will database indexes occupy? Typically, indexes require as much space as the data. For example, 50 GB of data will require about 50 GB of indexes.
- What is the expected annual growth rate of the database by record type and location?

Include the following information in your analysis of records:

- Number of addresses that will be assigned to each customer account
- Number of employees that will be assigned to each account
- Number of contacts that will be assigned to each account
- Number of attachments that will be assigned to a record
- Number of activities that will be associated with each account
- Whether or not opportunities, quotes, or orders will be stored
- Whether or not product data will be stored
- Whether or not Siebel Remote will be used

Include temporary table space, log files, and space required for loading data.

Mapping Business Requirements to Siebel Server Components

This topic is a step in "Process of Infrastructure Planning" on page 29.

This infrastructure planning step identifies the Siebel Server components needed to meet your business requirements.

Begin by listing the Siebel applications that users will run. For each application, identify the associated Application Object Manager (AOM). If you are deploying internationally, list the language-specific AOMs you will need.

Many AOMs require additional server components such as Workflow Manager. Users typically do not interact with these second-tier components directly. The role of these components is to support the function of AOMs and of the Siebel Server.

All second-tier component requirements must be correctly identified in your planning and review phases. Work closely with your implementation team to identify these components.

After you have identified all required server components, group them by business location. Then, for each location, determine the anticipated workload volumes for all components. Consider both average and peak workloads. This information is key for deciding how to distribute AOMs and other components across Siebel Servers.

Choosing a Load Balancing Method

This topic is a step in "Process of Infrastructure Planning" on page 29.

Siebel Business Applications support two methods of load balancing requests from a Web server to multiple Siebel Servers: Siebel native load balancing and third-party HTTP load balancers. For a description of these two methods, see "About Siebel Server Load Balancing" on page 19.

Siebel native load balancing and third-party HTTP load balancers provide similar features. Table 3 on page 34 compares key characteristics of these load balancing options. In the table, SISNAPI is the Siebel protocol used to communicate with Siebel Servers.

Table 3. Load Balancing Method Comparison

Feature Area	Siebel Native Load Balancing	Third-Party HTTP Load Balancer
Product form	Part of the Siebel Web Server Extension software.	Can be a dedicated device or part of an intelligent network switch. If it is software-based, it is usually installed on an available server. Considered part of customer's networking infrastructure environment.
Installation	Part of the Siebel installation process.	Varies by vendor. Hardware-based load balancers must be physically installed on the network. May have network topology restrictions.
Configuration	Supports SISNAPI protocol.	Must define server rules to support routing of SISNAPI connections. Hardware-based load balancers are typically administered using a Web browser. Software-based load balancers provide administration software.
Load balancing scheme	Round-robin only.	Response-time-based, resources-based, or round-robin.
Scalability	No application-imposed hard limit.	Varies by vendor. Typical limiting factors are network traffic throughput and number of servers for each load balancing pool.
Server health checks	Connection success or failure is monitored through SWSE stat page. No active checks.	Supports ICMP, TCP, and HTTP health-checks. HTTP health-checks are recommended.
Security and network access	Web server must directly connect to the Siebel Server.	Generally supports NAT, VIPs, VPorts. Also supports packet inspection and filtering.
Administration and configuration	Configured using text file. Administered through Siebel Server administration methods.	Generally configured and administered through Web interface and command line tools.
Deployment limitations	All load-balanced servers should have same configuration and equal load capacity.	No limitations on load balancer except network topology requirements.

Load Balancing Guidelines

Third-party HTTP load balancers are a good choice when any of the following is true:

- Hardware load balancers are already in use or are preferred.
- Hardware load balancer provide security features you require.
- A more sophisticated load-balancing scheme is desired.
- The site requires centralized monitoring and management of system hardware and network infrastructure.

Siebel native load balancing distributes user login requests in a round-robin fashion, which works best if all servers are configured equally and have similar capacities. Other considerations include the following:

- Configure all load-balanced Siebel Servers with the same Maximum Tasks setting for an application.
- Allocate all load-balanced Siebel Servers with an equal amount of server resources, such as CPU and memory configuration. For example, you will run Siebel Call Center on two Siebel Servers. One of them also will run some other software or component. On this server, Siebel Call Center must compete for resources with the other software. Allocating unequal loads on your Siebel Servers is not recommended.
- Once you have selected a load balancing method, it is important not to set the maximum number of tasks for an Application Object Manager (AOM) on a server or other load-balanced component higher than the server can reasonably handle. For information on planning and managing server task loads, see *Siebel Performance Tuning Guide*.

Defining High Availability Policies

This topic is a step in "Process of Infrastructure Planning" on page 29.

This infrastructure planning step defines business policies regarding availability of servers.

For more information about high availability, see Chapter 4, "High Availability Deployment Planning."

Siebel Servers

For each business location, assess the impact of losing each server component. Consider the possibility of the component failing, rather than the hosting platform itself. Individual server components that are important to normal application function must be identified in your planning and review phases. Work closely with your implementation team to identify all components that could represent single points of failure.

After you complete this analysis, define high-availability policies for all applications and services. Decide how long your business can tolerate not having access to key applications. Also, decide how long your business can tolerate degraded performance.

For example, a company decides that Siebel Call Center will run 24 hours a day, seven days a week, and the maximum acceptable downtime is 30 minutes. The company also decides the maximum time it can accept degraded performance is one hour.

Finally, at each business location, list all the server components to which each policy applies. This analysis forms the basis for implementing a high-availability strategy as part of hardware planning.

Database Platform and Data Integrity

The server platform that hosts the Siebel Database is crucial to Siebel deployment operations. For this reason, it is important to define high-availability and data integrity policies specifically for the database server. The following policies are recommended:

- Cluster database servers to protect against platform hardware failures.
- Use redundant disk arrays (RAIDs) for disk storage. RAID 1+0 is recommended because it provides maximum performance, and there is no data loss if a disk fails. Do not implement RAID 0 arrays. RAID 0 offers good performance but does not protect data adequately in the event of a disk failure.
- Enable transaction logging.
- Observe the following best-practice guidelines for storing database files:
  - Store data and indexes on separate disk subsystems.
  - Store active log files and archived log files on separate disk subsystems.
  - Store the database and database control files on separate disk subsystems.
- To allow for good OLTP performance, set up four rollback segments (if you choose to use them) for each 20 to 40 users. The size of rollback extents should be 100 KB for Initial extents and 100 KB for Next extents. If you are using Siebel EIM, also create several additional, large rollback segments to support EIM loads.

Siebel Gateway Name Server

The Siebel Gateway Name Server maintains the configuration information for all Siebel Servers in all the Siebel Enterprise Servers it manages. Loss of the Gateway Name Server due to a disk failure could bring your Siebel deployment to a halt while the system is restored.

It is strongly recommended that you install a RAID or some other type of redundant disk configuration on your Gateway Name Server.

Mobile Users

A Siebel Server temporarily stores transaction files that move to and from Siebel Remote mobile users. The loss of these files will result in the need to re-extract the database for all affected mobile users. (Siebel Remote supports synchronization of data between Siebel Mobile Web Clients and the Siebel Database through a dial-up connection.)

It is strongly recommended that you install a RAID or some other type of redundant disk configuration on Siebel Servers that run Siebel Remote.

Mapping Siebel Deployment Elements to Platforms

This topic is a step in "Process of Infrastructure Planning" on page 29.

This infrastructure planning step maps the elements of the Siebel deployment to server platforms.

Criteria

Mapping Siebel deployment elements to platforms must meet the following criteria:

- Guarantees adequate performance and scalability under both average and peak workloads
- Meets high-availability and resiliency goals
- Accommodates infrastructure security requirements

Prerequisites

Review the following information, developed in previous steps:

- Database requirements. See "Determining Database Requirements" on page 32.
- Required Siebel Server components. See "Mapping Business Requirements to Siebel Server Components" on page 33.
- High-availability policies. See "Defining High Availability Policies" on page 35.

To determine server platform requirements

1. Determine the amount of hardware required for Siebel Server components. Consider both average and peak workloads. Also consider background processing workloads.

   On two- or four-CPU platforms, customers typically deploy one Application Object Manager (AOM) on each Siebel Server. Deploying AOMs in this manner is a common practice, but not a requirement. Depending on the number of concurrent users, the amount and complexity of customization and the components used, the distribution of components can vary.

   For detailed information about calculating the settings for MaxTasks, MaxMTServers, and MinMTServers, see *Siebel Performance Tuning Guide*.

   For additional help with sizing Siebel Servers, contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.

2. Identify which Siebel Server components you can collocate. Distribute these across platforms in a way that evenly distributes workload.

   When deciding which server components to collocate, observe the following best practices:

   - Install the document server on a dedicated server. The document server uses Microsoft Word to create documents. Since Word is a single-threaded process, the document server could block other processes running on the same server.

- Consider what time of day a component will be used. For example, a typical scenario is to run EIM batch jobs during off-peak hours. Doing this means EIM can be collocated with a server that is busy during peak hours. You can collocate components running off-peak with on-peak components for server consolidation purposes.
- Collocation architecture decisions are driven by the load placed on each component and the overall size of the deployment. Consider this scenario: an implementation plan involves having 1000 connected users with light Assignment Manager usage, light Workflow usage, heavy off-peak EIM batch jobs, and a moderate Communications Server load. This implementation also requires some degree of high-availability and resiliency. One possible architecture solution would be as follows:
  - Clustered Gateway Name Server. Collocated Siebel File System, Assignment Manager, Workflow Monitor Agent and Communications Server.
  - Multiple, load-balanced Siebel AOMs. The number of servers required depends on the level of configuration and scripting complexity.
- You may choose to engage Oracle's Application Expert Services to perform an architecture and sizing review early in the implementation process. Once key metrics are known (user count, data volume, transaction volume, interfaces, and so on), you can determine the architecture and size the system. Contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.
- Some components are complex to size (for example, Configurator) and require expertise to determine an appropriate architecture and distribution of components. It is always necessary to have a thorough understanding of your customizable products and how they will be used.
- It is common practice in a large implementation (with thousands of users) to dedicate a server (usually a set of servers) to one function. Doing this makes it easier to monitor the performance of each segment of the implementation and makes it easier to scale each subset of the architecture. In a large implementation, AOMs, Workflow, EAI, Assignment Manager and Configurator would all run on dedicated servers. The Siebel Gateway Name Server and the Web servers would also run on dedicated hardware.
- When there are more than a relatively small number of remote users (100 or so), run Siebel Remote on a dedicated server. Planning considerations should include the total number of remote users, the expected data volume transferred between the enterprise database and the remote client, and the quantity of visibility events. Visibility events include adding or removing a position to an account team, adding or removing a user to a position, and so on. When a visibility event occurs, it can cause a lot of Siebel Remote activity.

  It is important to identify those processes with the potential for spikes of resource consumption and to spread the load accordingly.

3 Determine how many additional hardware platforms are needed to comply with high-availability policies.

  For clustered servers, define a failover strategy for components (active-active, active-passive).

  NOTE: In active-active clustering, a process is only active on one node of the cluster and is not active on the other node; that is, two physical servers are running a different clustered process.

4 Identify additional hardware required to comply with security policies. For example, do you need to install additional firewalls or a proxy server? Do you need to install LDAP servers?

5 Use average and peak workload information to determine how many Web servers are needed.

6 Create a diagram of the Siebel deployment that shows all the platforms and the distribution of Siebel Servers. Use the diagram to do the following:

   a Verify that all needed server components are enabled and correctly set up on each platform.

   b Run component and platform failure scenarios. Verify that there are no single points of failure that will cause unacceptable impacts.

   For example, assume you have one Web server. All your inbound customer orders must go through it and then to one HTTP inbound adapter. If the Web server or the inbound adapter fails, customers cannot place orders.

7 Use server naming conventions to identify groups of servers that provide similar functions.

   For example, in an enterprise, Application Object Managers (AOMs) run on one group of machines, workflows run on a second group of machines, and remote user synchronization runs on a third group. Give the AOM servers names starting with APP, the workflow servers names starting with WF, and the Siebel Remote servers names starting with REM.

   Each group will display together in Server Manager, which simplifies server administration.

   NOTE: Additional considerations for server naming are provided in the *Siebel Installation Guide* for the operating system you are using.

Topology Planning Guidelines

Consider the following guidelines for topology planning:

- A single Siebel Gateway Name Server can manage multiple Siebel Enterprise Servers.
- A Siebel Enterprise Server can belong to one and only one Siebel Gateway Name Server.
- A single Siebel Enterprise Server can manage multiple Siebel Servers.
- A Siebel Server can belong to one and only one Siebel Enterprise Server.
- A Siebel Server can manage multiple instances of a single server component or of multiple server components. Components may include multiple Application Object Manager types, each with its own SRF.
- The Siebel Servers in a Siebel Enterprise Server can connect to only one Siebel Database.

Table 4 on page 40 describes possible deployment schemes for Siebel Business Applications.

Table 4. Siebel Deployment Schemes

Deployment Scheme	Recommended?
A single Siebel Gateway Name Server with multiple Siebel Enterprise Servers configured on a single machine or UNIX hardware partition. Each Siebel Enterprise Server has its table owner and Siebel Database.	No. If the Siebel Gateway Name Server fails, it would adversely affect all the Siebel Enterprise Servers. Failure of one UNIX partition could adversely affect all the Siebel Enterprise Servers. If one Enterprise Server requires an upgrade, all Enterprise Servers are affected.
Running multiple Siebel Gateway Name Servers on a single unpartitioned machine or on a single UNIX hardware partition.	No. This scheme is very difficult to set up. It requires manual workarounds. On Windows, it requires manually editing the registry.
Multiple Siebel Enterprise Servers sharing a DBMS table owner (schema).	May be suitable for some deployments. There are many possible scenarios for deploying with multiple Enterprise Servers and many issues to consider. For more information, see 477829.1 (Article ID) on My Oracle Support. This document was previously published as Siebel Technical Note 544.
Multiple Siebel Enterprise Servers, each with their own DBMS table owner and sharing the same instance of the DBMS executable.	No. If the database server goes down, all of the Siebel Enterprise Servers will go down. Too much dependency on one machine.
A single Siebel Enterprise Server hosting multiple Siebel Servers within a single hardware partition.	No. There is no additional scalability, throughput, or performance benefit to this scheme. Managing multiple Siebel Servers within a single hardware partition also requires more Siebel administrator time.
A Siebel Enterprise Server hosting multiple Siebel Servers, with each Siebel Server on its own machine or UNIX hardware partition (multiple partitions on each UNIX server machine).	Yes. This scheme is the most common way to deploy Siebel Business Applications.

Table 4. Siebel Deployment Schemes

Deployment Scheme	Recommended?
A single Siebel Server managing multiple applications, each Application Object Manager type having its own copy of the SRF file.	Yes. This is a common deployment scheme. It allows you to segment functionality for each process. If multiple SRFs are used for the Object Managers in a Siebel Enterprise Server, the SRF used for common components such as Workflow Process Manager, EAI Object Managers, and so on should have the common objects needed for proper processing. It is critical to have all SRF files in sync. The Siebel Repository in the production DBMS must also be in sync with the SRFs.
Installing the Siebel Gateway Name Server, each Siebel Server, and the Siebel Database all on different operating systems.	Yes. This scheme is supported. However, you should try to keep the deployment as simple as possible. In some cases a heterogeneous environment is required. For example, you want to install Siebel Servers running on one operating system, but a third-party product you need only runs on another. For information on supported operating systems for Siebel deployments, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

Determining Network Requirements

This topic is a step in "Process of Infrastructure Planning" on page 29.

The purpose of this infrastructure planning step is to identify network requirements needed to support the Siebel deployment.

To determine network requirements

1. Use the information on average and peak workloads to verify that there is sufficient network bandwidth to handle network traffic to and from, as well as within, the Siebel deployment.
2. Determine whether you will use data encryption. If so, define data encryption policies. Then add data encryption protocols to the deployment plans you outlined using "Mapping Siebel Deployment Elements to Platforms" on page 37.
3. Define firewall requirements. If you are creating a network demilitarized zone (DMZ), also define requirements for proxy servers and other items that will be installed in the DMZ.

   Include network address translation (NAT) and HTTPS requirements.

4 Analyze interactions and dependencies between networking components.

   For example, assume you plan to use HTTP with SSL between browsers and Web servers. You also plan to install a Web server load balancer. Typically, Web server load balancers cannot do HTTP, URL-based load balancing unless the load balancer has an integrated SSL accelerator.

   An SSL accelerator allows SSL connections to be terminated at the Web server load balancer. This allows the load balancer to parse packet information and perform HTTP, URL-based load balancing.

5 Write down the virtual IP (VIP) address used by Web server and Siebel Server load balancers. Make sure the requesting component is set up to access the VIP. Furthermore, make sure you have configured firewalls to allow VIP traffic to go through.

6 Select port numbers for all network components that listen on TCP ports.

   This includes Web server load balancers, Web servers, Siebel Server load balancers, Siebel Servers, and server clusters. For a full description of Siebel Server components that require a port number assignment, see *Siebel System Administration Guide*.

   The default TCP port number for Web server-to-Siebel Server traffic is 2321. This is the port number of the Siebel Connection Broker (SCBroker); however, you can configure the port number. If a third-party HTTP load balancer is deployed, then you must set it up to communicate with Siebel Servers using the SCBroker port. You cannot assign the Siebel Gateway Name Server and Siebel Servers port numbers higher than 32767.

   If Siebel native load balancing is deployed, then the load balancing configuration file must reference this port number. See "Choosing a Load Balancing Method" on page 33.

   Also, verify that firewalls are configured to communicate with the correct TCP ports.

7 Consider any other factors that may affect networking connectivity.

Defining a Test and Transition Plan for the Siebel Deployment

This topic is a step in "Process of Infrastructure Planning" on page 29.

It is important that you define a test plan to verify that the proposed deployment infrastructure functions correctly and is sized correctly. Equally important is defining a plan that transitions the Siebel deployment to production.

Observe the following best-practices guidelines for testing the Siebel deployment and transitioning it to production:

- **Separate production environment.** Keep the development and test environments physically separate from the production environment. Development and test activities should not be conducted on the production Siebel Database or, if possible, the production database server.
- **Server stress testing.** Test Siebel Enterprise Server performance under average and peak workloads.

Oracle's Application Expert Services finds that performance problems at customer sites are frequently caused by the following:

- Servers were tested at much less than average or peak workloads. This prevents configuration and tuning problems from being uncovered.
- Siebel Server components are either incorrectly distributed across servers or are not configured correctly.
- The load balancing strategy is ineffective under typical workloads. This can be caused by stress-testing the servers with a workload that has characteristics different than the production environment.

- **Failover and resiliency testing.** Define a test plan that evaluates the effect of server component failures. Work closely with your implementation team to identify all components that could represent single points of failure, as noted in "Defining High Availability Policies" on page 35.

  Define a server cluster test plan that evaluates failover behaviors. Run the test plan under average and peak workloads. It is particularly important to verify that failover performance under peak workloads is acceptable.

- **Database server testing.** Define a test plan that evaluates the following:
  - **OLTP performance.** Consider OLTP performance under average and peak workloads.
  - **Database server platform failover.** Typically the database server is clustered.
  - **Recovery from database corruption.** The database vendor usually provides recovery mechanisms.
  - **Batch processing support.** Verify that the database server correctly handles batch jobs from servers as well as synchronization requests from Siebel Remote components.
  - **Web Client users.** Verify that batch jobs do not degrade transaction processing performance and are completed in a reasonable time.

4 High Availability Deployment Planning

This chapter includes the following topics:

How Service Failures Affect the Siebel Deployment

This topic describes how major architectural components in a Siebel deployment are affected when a service failure occurs. Such failures may be prevented or mitigated by using high availability deployment options. Services include both hardware platforms and software applications. Subtopics include:

Components Involved in Service Failures

This topic is part of "How Service Failures Affect the Siebel Deployment" on page 45.

This topic identifies major architectural components in a Siebel deployment that may be affected when a service failure occurs.

Siebel Web Clients

Client hardware failure and browser failures are the most common causes of Siebel Web Client failure. Operating system failures can also cause this, but are rare.

When the Siebel Web Client fails, user sessions are lost even though the sessions may continue running on the Siebel Server for a time. The user session is lost because when the Web Client fails, the Siebel session cookie usually is also lost. Without the cookie, the user cannot be routed back to the existing user session on the Siebel Server. Therefore, the user will usually need to log in again and start a new user session.

Web Servers

Web servers may fail because of hardware or software issues. Typically, when the Web server fails, Web Clients cannot access Siebel applications, since requests must go through the Web server first. Existing connections from the Web server to Siebel Servers are also lost.

If Web servers are set up for high availability, for example if there are multiple, load-balanced Web servers, then subsequent requests can be routed to another working Web server. Usually when this occurs, the function of affected Web Client user sessions is not noticeably affected.

Third-Party HTTP Load Balancer for Siebel Servers

Third-party HTTP load balancers handle communication between Web servers and Siebel Servers. Causes of failure differ significantly between hardware-based and software-based solutions. When the load balancer fails, Web Clients and Web servers going through the load balancer cannot communicate with Siebel Servers. Network connections in most cases would also be severed, and user sessions are lost.

If there are multiple, clustered load balancers, then the backup load balancer can take over. Some load balancers can failover TCP sessions to a backup load balancer. See the vendor's load balancer documentation for details.

When the backup load balancer takes over, user sessions continue without interruption. However, user sessions are lost if any of the following occurs:

- A Web Client makes a request while the load balancers are failing over.
- TCP sessions are not cleaned up properly on the Web servers.

Siebel Servers

Siebel Servers may fail because of hardware or software issues. If the hardware platform fails, or the Siebel Server software fails, then all Siebel Server components are lost.

In other cases, individual Siebel Server components may fail. In turn, component failure can cause related user sessions or user requests to fail. The major groups of Siebel Server components are as follows:

- **Application Object Managers (AOMs).** When AOM processes terminate unexpectedly, user sessions hosted by the AOM are lost. Users must log in to the Siebel application again.

  If users return to the same Siebel Server, SCBroker tries to route the user request to a running AOM process.

  If there is only one AOM process and it has failed, then the request is directed to a different Siebel Server, unless there is only one Siebel Server.

- **Batch-mode server components going through SRBroker.** Most batch-mode server components receive server requests through SRBroker. An example is Workflow Manager. When a batch-mode component fails, the current server request fails.
- **Synchronous server requests.** An error is returned to the requesting component.

- **Asynchronous server requests.** An error is logged but not returned to the requesting component.

  Subsequent requests for the failed batch-mode component will be attempted against either a different instance of the component on the same Siebel Server, or an instance of it on a different server.

  If no instance of the batch-mode component is available, then the request is logged to the S_SRM_REQUEST table to be processed later.

- **Direct Object Manager requests.** Examples of direct Object Manager requests are those to Siebel Product Configuration Object Manager. Some components, such as Siebel Configurator, have a native failover mechanism.

- **Other server components with location restrictions.** There are specialized server components that do not communicate through SRBroker. Siebel Remote Server is an example. Typically, requests to these components can only be processed by a specific Siebel Server. Therefore, if the server fails, requests to that server will fail, until the server is restarted.

Siebel Database

Access to the Siebel Database can fail due to any of several factors:

- Database server hardware failure
- Database server running out of resources
- Disk failure
- Network failure

The impact on the Siebel deployment will be either temporary or long term. For example, a temporary networking interruption, or a quick database server reboot, would result in a temporary disruption in service. A long-term interruption may occur when there is database corruption or a major server malfunction.

In general, user sessions are lost when there is a Siebel Database service interruption. Users must log in to the system again. Application Object Manager sessions will continue to try to connect to the database and once the database is running (assuming the connection retry count has not been exceeded), the connection will succeed. Users should not notice that there was an outage, unless they were currently working at the time of the database failure. In this case, users get database error messages.

If the interruption is temporary, interactive server components and most of the batch-mode server components try to reconnect with the Siebel Database.

If the interruption is long-term, the Siebel deployment must be shut down and restarted once the database service is restored.

NOTE: It is strongly recommended not to collocate a Siebel Server or other component with the Siebel Database. Such a deployment may affect performance of either component, and may in some cases lead to Siebel Server failure or extend interruptions of availability. For example, where a Siebel Server and the Siebel Database are collocated, if the server machine is rebooted, the database instance may not be up by the time this Siebel Server is up, so a connection could not be made. Siebel Server failure could result. However, if the Siebel Server and the database are on separate servers, as recommended, the database would be unaffected by rebooting the server machine where Siebel Server is installed, and would be available as soon as the Siebel Server was ready to connect.

Impact of Service Failures

This topic is part of "How Service Failures Affect the Siebel Deployment" on page 45.

Table 5 on page 48 summarizes the impact of failure of services in the Siebel deployment. The table includes information on specific services not already covered.

Table 5. How Service Failures Affect the Siebel Deployment

Service Failed	Affected Component	Impact
Siebel Gateway Name Server	Siebel Server components	You cannot start or add any new components. Users can continue to log in and out of Siebel applications. Existing user sessions are not interrupted. Server requests will continue to be processed successfully. Exceptions are listed below.
	Server administration functions	Unavailable.
	Siebel Product Configuration Object Manager	You can still launch product configuration sessions, as long as the connection information has been cached. By default, the connection information is cached when the first connection is made.
	Gateway Name Server database (siebns.dat)	This database maintains server configuration information for the Siebel Enterprise Server. If this database is corrupted or lost, you must reinstall all Siebel Enterprise Server software.

Table 5. How Service Failures Affect the Siebel Deployment

Service Failed	Affected Component	Impact
Siebel Server	AOM components	The Siebel application is unavailable. Siebel Connection Broker (SCBroker) failure: You cannot create new user sessions. If the SISNAPI connection between the Web server and the Object Manager fails, SWSE will retry the connection. If after a certain number of attempts the connection is still not available, the connection will completely fail and the user gets an error message. Existing user sessions are unaffected by SCBroker failures.
	EAI	Interface to external application unavailable.
	Batch components	Loss of functionality (components such as Assignment Manager or Workflow unable to process server requests).
Siebel File System	Attachments	Unavailable.
	Correspondence	Unavailable.
	Shared user preference files	Unavailable.
	Docking transaction files from EIM	Unavailable.
	Email Response	Unable to process inbound messages. Unable to send outbound messages with attachments.
File System Manager (FSM)	Components that access the FSM	Current requests fail.
	Attachments	Unavailable.
Web server	Siebel Web Clients accessing Application Object Managers (AOMs)	The Siebel application is unavailable to Web Clients. Mobile Web Clients are unaffected.
	EAI inbound HTTP adapter	Unavailable.
Siebel Database	Client access, background tasks, batch tasks	Unable to access Siebel Business Applications. Siebel Servers cannot function. Only the Mobile Web Client is not immediately affected by a Siebel Database failure.
	Batch and interactive components	Unavailable.

Specific Failures and Associated Impact

This topic is part of "How Service Failures Affect the Siebel Deployment" on page 45.

Table 6 on page 50 describes potential failure scenarios when running a Siebel deployment, and summarizes benchmark test results and the associated impact on the tested Siebel environment.

NOTE: All tests were conducted in a test lab. Actual results in production may differ due to the complexity of a production environment.

The test environment included the following:

- Multiple Web servers were deployed with load balancing provided by a hardware-based HTTP load balancer.
- Multiple Application Object Manager servers were deployed with load balancing provided by either Siebel native load balancing or third-party HTTP load balancers (usage depending on test scenarios).
- Multiple batch component application servers were deployed. Request distribution was provided by Server Request Broker (SRBroker) and Server Request Processor (SRProc).
- Siebel Product Configuration Object Manager servers were used and load balanced by the Siebel Configurator-provided load balancing scheme.
- A clustered pair of database servers was used.
- A clustered Siebel Gateway Name Server was also deployed.

Table 6. Specific Failures and Associated Impact

Component Tested	Failure Scenario	Observed Behavior
Siebel Database	Observe system behavior while driving server CPU load to 100%.	■ Significant response time impact. ■ No failures were observed.
Application Object Manager (eChannel)	Observe system behavior while driving server CPU load to 100%.	■ Minor response time impact. ■ No failures were observed.
Web Server	Observe system behavior while driving server CPU load to 100%.	■ Negligible response time impact. ■ No failures were observed.
Workflow Server	Observe system behavior while driving server CPU load to 100%.	■ Negligible response time impact. ■ No failures were observed.
Application Object Manager (eChannel)	Observe system behavior while server memory consumption is 100%.	■ Significant response time impact. ■ Increased CPU usage and context switching were observed. ■ A few login failures were seen when attempting to log in additional users.

Table 6. Specific Failures and Associated Impact

Component Tested	Failure Scenario	Observed Behavior
Workflow Server	Observe system behavior while server memory consumption is 100%.	■ Major response time impact. ■ Increased CPU usage and context switching were observed. ■ A few login failures were seen when attempting to log in additional users.
Application Object Manager (eChannel)	Observe system when all available disk space is consumed on the tested server.	■ Minor response time impact in some transactions. ■ Major response time impact when logging in new users. ■ No failures were observed.
Workflow Server	Observe system when all available disk space is consumed on the tested server.	■ Significant response time impact in Workflow transaction response time. ■ Significant response time impact when additional users logged in. ■ Negligible increase in CPU and context switching. ■ No failures were observed.
SCBroker	Simulate a process failure for various task-based server components while the server is handling both synchronous and asynchronous server requests. Also note system recovery after bringing the process back up.	■ SCBroker auto-restarts upon receiving an SEGV signal. ■ No failures were observed. ■ A new SCBroker was started when an SEGV signal was received.
SRBroker	Simulate a process failure for various task-based server components while the server is handling both synchronous and asynchronous server requests. Also note system recovery after bringing the process back up.	■ SRBroker does not auto-restart upon receiving an SEGV signal. ■ When eScripting invokes a workflow, users get a `no server connect string` error message. ■ Failures were seen for the above step.

Table 6. Specific Failures and Associated Impact

Component Tested	Failure Scenario	Observed Behavior
WfProcMgr (Workflow Process Manager)	Simulate a process failure for various task based server components while the server is handling both synchronous and asynchronous server requests. Also note system recovery after bringing the process back up.	■ Shutdown WfProcMgr on one server caused a few failures initially and then stabilized with no further failures. ■ CPU and memory activity increased on the server still running WfProcMgr. ■ When the other WfProcMgr is shut down many failures resulted. ■ Brought up one WfProcMgr and no more failures were observed.
Siebel Gateway Name Server	Simulate the failure of the Siebel Gateway Name Server.	■ Unable to connect to srvrmgr but the transactions were passing. ■ When adding 100 more users, still unable to connect to srvrmgr but no errors were observed. ■ When the Gateway Name Server was restarted, still unable to connect to the Gateway Name Server.
Application Object Manager	Consume all available tasks on an Object Manager component and observe the result.	■ Object Managers components fail over to another Object Manager as expected when MaxTasks is reached. ■ When all Object Managers are out of tasks, the user receives a server busy error message. ■ When some users log out, new users can connect to servers again. ■ For more information about MaxTasks, see *Siebel System Administration Guide* and *Siebel Performance Tuning Guide*.

Table 6. Specific Failures and Associated Impact

Component Tested	Failure Scenario	Observed Behavior
Application Object Manager	Simulate resource leaks while server recycling is enabled, and verify how process recycling works under load.	■ New Object Manager process is created when `MemoryLimit` is reached. ■ Old Object Manager process remains instantiated for a period of time (even when no more users are running on it), but eventually the old Object Manager is recycled. ■ When `MemoryLimitPercent` is reached, then the memory-consuming process restarts. ■ For more information about `MemoryLimit`, `MemoryLimitPercent`, and related parameters, see *Siebel System Administration Guide*.
Application Object Manager	Simulate applying a new SRF (simple SRF and browser script changes) and stopping and restarting each server.	■ Browser script gets updated on a visit to a URL. ■ Browser does not respond after user visits the URL, even after browser scripts get updated.

Application Object Manager	Simulate a thread or process that is not responding.	■ Users can still log in to the AOM with the non-responsive process.
		■ After simulating a non-responsive process, 100 extra users were added. After that, stopping the non-responsive process caused about 40 running users to fail (users on that AOM). The following can be inferred:
		  ■ An AOM with a non-responsive process still receives new connections.
		  ■ You cannot safely stop a non-responsive process unless you set the component group offline and shut down the whole component group or Siebel Server.

About High Availability Deployment Options

High availability means that a user can access key system services even when the underlying hardware or software for those services fails. For example, if a user synchronization session was interrupted by a failure of the server to which it was connected, the user can reconnect to the system and restart the synchronization process without any data loss.

To achieve high availability, the system must automatically replace lost services and distribute loads among services to assure acceptable response times. When the system cannot replace a lost service, this is called a single point of failure. High availability planning and deployment are designed to eliminate these single points of failure.

In a Siebel Business Applications deployment, a service (for the purposes of this discussion) is one of the following:

- Siebel Gateway Name Server
- Siebel Server
- Siebel Database
- Siebel File System
- Web server with the Siebel Web Server Extension (SWSE) installed

To eliminate single points of failure, some form of redundancy is required. Clustered servers are an example. When one service fails, other resources are available to take over for the failed service. To be successful, this process must be:

- Automatic: No operator intervention is necessary
- Transparent: Users do not have to change anything for the services that have failover protection

There are cases where full, automatic failover may not be possible. For example, results of the failure may need to be manually cleaned up. This guide does not cover all scenarios, and customers are advised to review environment-specific requirements before finalizing high availability planning.

The options available for high availability deployment consist of the following techniques:

- Scalable services (load balancing)
- Resilient processing (distributed services)
- Server clusters

Scalable Services (Load Balancing)

Load balancing distributes workload across multiple servers. Each server runs an instance of the service you want to load-balance. Load balancing also provides failover. If one server fails, then requests are automatically routed to the remaining servers.

Application Object Managers (AOMs) are the server components for which load balancing is most frequently provided. Distributing workload across AOMs indirectly distributes workload across the other server components that AOMs call, in a form of indirect load balancing.

Resilient Processing (Distributed Services)

Resilient processing, also called distributed services, is used for tasks initiated by the Siebel Server. (Load balancing is used for tasks initiated by users.) Multiple instances of a component run on the same Siebel Server, or the same component can run on multiple Siebel Servers. If one instance of the component fails, then another instance on the same server or on a different server takes over processing subsequent requests. For more information, see "About Resilient Processing" on page 60.

Server Clusters

Server clusters consist of two or more physical servers linked together so that if one server fails, resources such as physical disks, network addresses, and applications can be switched over to the other server. Server clusters can provide resilience when a particular Siebel operation can only take place on one server, either because of the type of process (such as Siebel Gateway Name Server or Siebel Remote) or because of hardware constraints.

Figure 5 on page 56 illustrates an example of server load balancing and server clustering in a Siebel Enterprise Server.

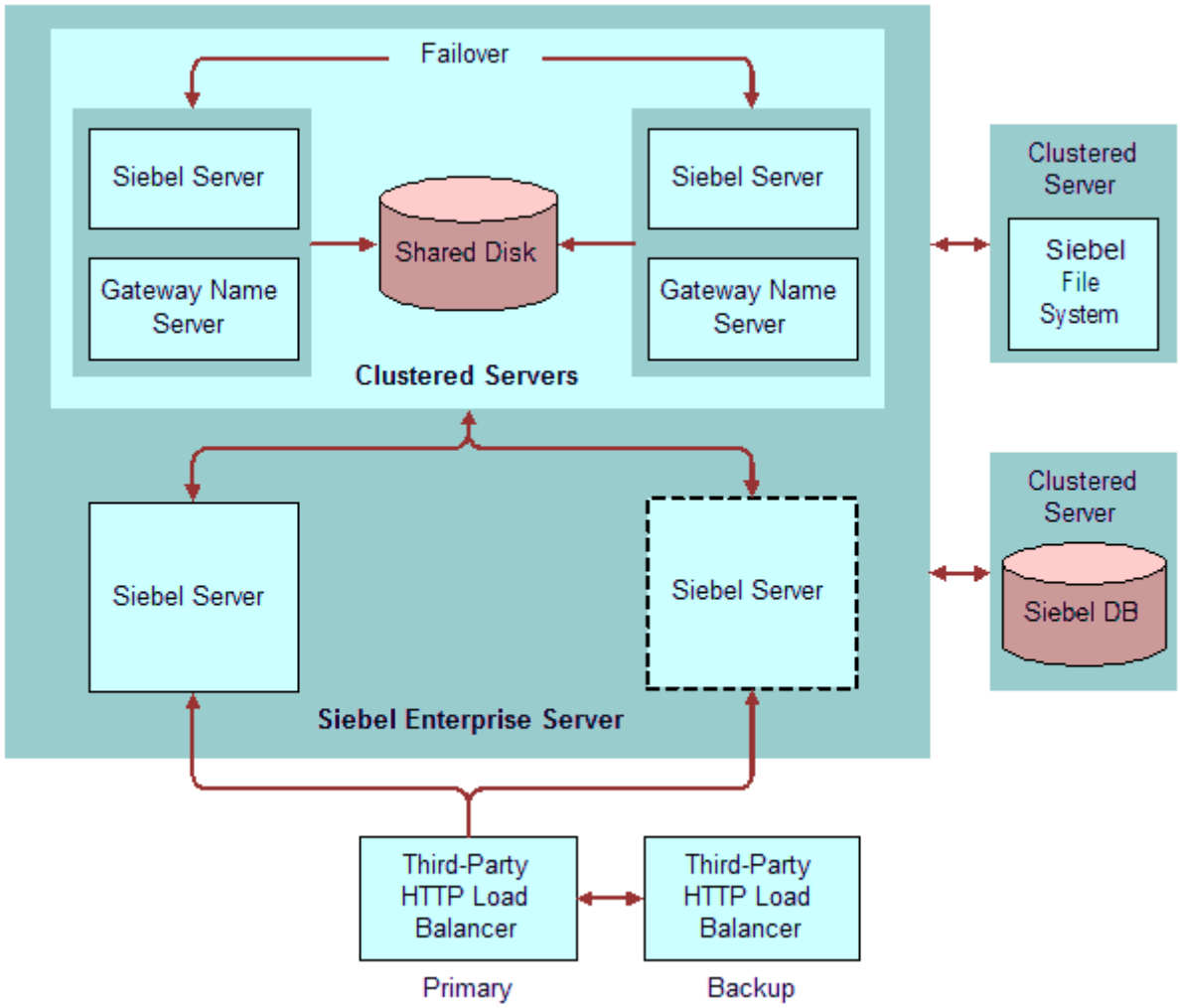

Figure 5. Example of a High Availability Deployment

Recommended High Availability Techniques for Specific Services

The three supported high availability techniques are server clustering, load balancing, and resilient processing. Table 7 on page 57 lists the recommended high availability technique for specific Siebel Enterprise deployment services:

- **Preferred.** Indicates that more than one high availability technique is supported for this function, but this is the preferred technique you should use wherever possible.

- **Supported.** Indicates a high availability technique is supported for this function. You can use this technique if local conditions prevent using the preferred technique.
- **Not Applicable (N/A).** The high availability technique in this column is not applicable for this component.

Table 7. High Availability Support Matrix for Siebel Components

Component	Clustering	Load Balancing	Resilient Processing
Siebel Gateway Name Server database (siebns.dat)	Preferred	N/A	N/A
Application Object Managers	Supported	Preferred	N/A
Communications Session Manager	Supported	N/A	Preferred
Siebel Configurator	Supported	Preferred. Uses its own load balancing method	N/A
Document Server	Supported[1]	N/A	Preferred
Pricer	Supported	N/A	Preferred
EAI (adapters and connectors)	Supported	Preferred, whenever possible[2]	Supported
EAI Object Manager	Supported	Preferred	N/A
Field Service (non-AOM components such as Appointment Booking System and Scheduling)	Supported	N/A	Preferred
File System Manager	Supported	N/A	Preferred
Interactive Assignment	Supported	N/A	Preferred
MQ Series Receiver	Preferred	N/A	N/A
Replication Agent	Preferred	N/A	N/A
SAP BAPI Integration	Preferred	N/A	N/A
SAP IDOC Receiver	Preferred	N/A	N/A
SAP IDOC Receiver for MQ	Preferred	N/A	N/A
Siebel File System	Supported	N/A	N/A
Siebel Marketing	Supported	N/A	Preferred
Siebel Remote	Preferred	N/A	N/A

Table 7. High Availability Support Matrix for Siebel Components

Component	Clustering	Load Balancing	Resilient Processing
Workflow Monitor Agent	Preferred	N/A	N/A
Workflow Process Manager	Supported	N/A	Preferred

1. Document Server clustering is supported where Microsoft Office has been installed on all clustered nodes. This approach may be particularly helpful in smaller deployments.

2. There are so many different types of EAI deployments that providing a single, standardized recommendation is not practical. For help determining the best approach for your deployment, contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.

Best Practices for High Availability Deployments

Use the best practices below as a starting point for planning a high availability infrastructure.

Profile 1: Uninterrupted Global Deployment

This deployment has several hundred to tens of thousands of users worldwide requiring the Siebel application to be available 24 hours a day, seven days a week.

- **Siebel Server load balancers.** A dedicated third-party HTTP load balancer is recommended for this type of deployment. If using hardware load balancers, set up redundant load balancers. Verify that if a load balancer fails, the remaining load balancers can provide acceptable performance under high workloads.
- **Siebel Gateway Name Server.** This should reside on a dedicated, clustered server pair. It can also reside on Siebel Servers in an existing cluster. Sharing the clustered servers will have minimal performance impact.
- **Siebel File System.** Consider deploying fault-tolerant and resilient file systems to host the files. Clustering the server that hosts the Siebel File System is also an appropriate strategy. The File System is restricted to one for each Siebel Enterprise Server; therefore, you cannot use load balancing.
- **Web servers.** Set up load balancing using one of the standard HTTP load balancers certified by Oracle. Set up the Web server load balancer to allow user requests to failover to other Web servers. Verify that if a load balancer fails, the remaining load balancers can provide acceptable performance under high workloads.

- **Siebel Servers hosting an AOM.** Multiple Siebel Servers hosting an Application Object Manager (AOM) should be load-balanced. You can use either a third-party HTTP load balancer or Siebel native load balancing. Third-party HTTP load balancers typically offer more management capabilities, while Siebel native load balancing is less complex to set up and maintain.

  Consider AOM or server failure when doing capacity planning. For example, if each Siebel Server can handle 500 users, and you typically have 1500 concurrent users, consider providing four Siebel Servers to handle this load. If one server fails, the other three can still support user loads.

  The Siebel Product Configuration Object Manager is an exception to standard load-balancing configuration. It includes an internal load balancing mechanism.

- **Siebel Servers hosting other types of components.** Enable batch components on multiple Siebel Servers. Server Request Broker will route requests to these components, thus providing resilient processing for batch requests.

  Some components can be hosted on only one Siebel Server, for example Siebel Remote. If user loads permit, you set up high availability as follows:

  - For the AOM and related components, use load balancing.
  - For the components that can be installed on only one server, use server clustering.

- **Siebel Database.** Deploy high availability clustered services provided or supported by the vendor of your RDBMS. To guarantee data availability and integrity, use data replication techniques such as mirroring and disk arrays to keep the backup instance of the database in sync with the primary instance. Also consider fault-tolerant file systems to host database files.

Profile 2: Large Domestic Deployment

This deployment has several hundred to several thousand users in an Enterprise deployment that is operational during standard business hours only.

- **Load balancers.** If using hardware-based third-party HTTP load balancers, set up redundant load balancers. Verify that if a load balancer fails, the remaining load balancers can provide acceptable performance under high workloads.
- **Web server.** Set up at least two load-balanced Web servers for high availability.
- **Siebel Servers hosting an AOM.** Same as Profile 1.
- **Siebel Servers hosting other types of components.** Same as Profile 1.
- **Siebel Gateway Name Server.** This should reside on a dedicated, clustered server pair and can also reside on Siebel Servers in an existing cluster. Sharing the clustered servers will have minimal performance impact.
- **Siebel File System.** Deploy a clustering technology that has been certified by Oracle. At a minimum, use a RAID 5 disk array for your file system. In addition, make regular backups of your data.
- **Siebel Database.** Same as Profile 1.

Profile 3: Limited Resources Deployment

This deployment has 500 users or less and operates during standard business hours with limited hardware resources.

Consider collocating multiple Siebel Servers and Web servers on a single computer. Use load balancing for each server type to achieve high availability at minimal cost. Siebel native load balancing for the Siebel Servers will work well in this configuration.

To establish high availability, consider putting the Siebel deployment in a two-system cluster. At minimum, make sure that the Siebel Gateway Name Server, Siebel Database, and Siebel File System are clustered.

Profile 4: Application Integration Deployment

This deployment uses third-party application servers to access the Siebel application. There are multiple integration points between Siebel applications and other, third-party applications. This profile may use Siebel EAI extensively.

There are no unique high availability requirements for this profile. See the previous discussions of the other profiles.

Make sure that the third-party applications are highly available by reviewing the specifications published by those vendors.

If you use multiple load-balanced Siebel Servers, review the recommendations for Profile 2.

About Resilient Processing

Resilient processing, also called distributed services, distributes server requests to multiple instances of batch-mode server components. The server requests for these components are typically message-based, so any instance of the component can process the request. If one instance of a component fails, another can perform the task, thus providing resiliency. Multiple instances of the components can run on the same Siebel Server or on several Siebel Servers.

Load balancing is about distributing workloads. Resilient processing is about providing redundancy. Resiliency also provides round-robin distribution of workloads to multiple instances of server components.

Resilient processing makes more efficient use of hardware resources than server clustering. In addition, resilient processing does not require third-party clustering software. Where possible, use resilient processing instead of server clustering, or use them in combination.

Resilient processing is the preferred method for providing high availability for the following server components:

- Communications Session Manager
- Document Server
- Pricer
- Field Service

- File System Manager
- Interactive Assignment
- Siebel Marketing
- Workflow Process Manager

Resilient processing uses two server components:

- **Server Request Broker.** This component handles all server requests. See "About Server Request Broker" on page 24.
- **Server Request Processor.** This component handles asynchronous server requests. See "About Server Request Processor" on page 25.

5 Server Clustering Planning

This chapter includes the following topics:

About Server Clustering

A server cluster is a group of two or more servers that are configured so that if one server fails, another server can take over application processing. The servers in a cluster are called nodes. Typically, these servers store data on a common disk or disk array.

Clustering software monitors the active nodes in a server cluster. When a node fails, the clustering software manages the transition of the failed server's workload to the secondary node.

When a clustered Siebel Server fails, all the applications and services on the server stop. Application users must reconnect and log in to the server that takes over. For example, if the Siebel Server that failed was hosting Siebel Communications Server, the communications toolbar is disabled, and users must reconnect and log in to the new server.

Cluster vendors can validate their third-party server cluster products to provide server clustering for deployments of Siebel Business Applications. For validation assistance, contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services. For recommendations and help on the use of cluster products with Siebel Business Applications, customers should contact the cluster vendor of their choice.

Active-Passive Configuration

An active-passive server cluster contains a minimum of two servers. One server actively runs applications and services. The other is idle. If the active server fails, its workload is switched to the idle server, which then takes over application processing.

Because the standby server is idle, active-passive server clusters require additional hardware without providing additional active capacity. The benefit of active-passive clusters is that, after a failover, the same level of hardware resources is available for each application, thereby eliminating any performance impact on users. This benefit is particularly important for performance-critical areas such as the database. The most common use of active-passive clusters is for database servers.

Active-Active Configuration

An active-active server cluster contains a minimum of two servers. Both actively run applications and services. Each may host different applications or may host instances of the same application. If one server fails, its processing load is transferred to the other.

Active-active configuration is the most common server clustering strategy for servers other than the database server.

NOTE: Configuring the Siebel Database (database server) and a Siebel Server to failover to each other is supported, but not recommended.

Potential Port Conflicts

Some Siebel Server components, such as Siebel Connection Broker (SCBroker), Siebel Gateway Name Server, Synchronization Manager (Siebel Remote), and Siebel Handheld synchronization listen on a configurable static port. When these components run in an active-active cluster, you must plan your port usage so there is no port conflict after failover.

For example, an active-active server cluster contains two platforms, each running a Siebel Server. If one platform fails, the other will host two Siebel Servers. Siebel Servers include several services, such as Siebel Connection Broker, that use a dedicated port. If this port number was the same on both platforms, there will be a port conflict after failover.

Capacity Planning

Active-active clusters use all the server platforms continuously. Consequently, they take better advantage of computing resources than active-passive clusters. When doing capacity planning, make sure that clustered servers have sufficient capacity to handle a failover. Because failovers are usually infrequent and normally last only a short time, some performance degradation is often acceptable.

Where to Use Server Clustering

Siebel Business Applications support server clustering for the following parts of a Siebel deployment:

- Siebel Gateway Name Server
- Siebel Servers

  Individual server components can be clustered, load-balanced, or both. Some Application Object Managers do not support or require clustering.
- Siebel File System
- Siebel Database

  Subject to limitations of third-party RDBMS software.
- Web server on which the Siebel Web Server Extension (SWSE) is installed.

In addition, server clustering is the preferred method for providing high availability for the following Siebel Server components or component groups:

- Workflow Monitor Agent
- Siebel Remote

  The DockString parameter in the Mobile Web Client configuration file must reference the virtual server name for remote synchronization to work after failover.
- Replication Agent

- MQSeries Server Receiver
- SAP BAPI integration
- SAP IDOC Receiver
- SAP IDOC Receiver for MQ

Server Clustering for Some Components Is Not Supported

Siebel Business Applications do not support server clustering for the following components:

- LDAP or ADSI directory server. Vendor may provide built-in replication.
- Siebel Document Server (Microsoft server-side integration)
- CTI hardware/switch
- Siebel Charts Server

Server Clustering and Load Balancing Can Be Used Together

You can set up server clustering and load balancing on the same Siebel Server. For example, you have three Application Object Managers running on a Siebel Server: AOM1, AOM2, and AOM3. You can set up server clustering for AOM1 and AOM2, and set up load balancing for AOM3.

Best Practices for Server Clustering

The best practices below help promote failover protection for your system. However, these practices are neither exhaustive nor all-inclusive.

- If you have multiple Siebel Servers running that are not clustered, these should be load-balanced.
- Make clustering the Siebel Database a high priority because it is a single point of failure. When clustering the Siebel Database, have it already installed and running. The Siebel Database should be the first server to be clustered.
- Install and configure clustering software on each node to detect failure of that node and to recover and manage all servers as a single system.
- Make sure all hardware used is certified for server clustering by the hardware vendor.
- The Siebel Enterprise Server installer allows you to install on a single machine all server modules for which you have a license. If you will be operating the Siebel Gateway Name Server and Siebel Servers as part of a cluster, you must install and configure the Siebel Gateway Name Server and the Siebel Server individually as separate cluster services.
- On the copy you made of the cluster deployment worksheet (located in an appendix of the *Siebel Installation Guide* for the operating system you are using), fill out the section related to server clustering, and refer to it during installation.

6 Data Integrity and Capacity Planning

This chapter includes the following topics:

Sizing the Database for a Siebel Deployment

As with most client-server applications, the overall performance of Siebel Business Applications is largely dependent on the I/O performance of the database server. To promote optimal I/O performance, you must arrange the tables and indexes in the database across available disk devices in a way that evenly distributes the I/O load.

The mechanism for distributing database objects varies by RDBMS, depending on the way storage space is allocated. Most databases can force a given object to be created on a specific disk.

To verify which RDBMS products, versions, and patch levels are supported, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

In your planning, you need to allocate space for multiple purposes, including system storage space, undo or rollback space, temporary table space, and space for logs and system files, as well as space for Siebel data and indexes. If you allocate too little space for your system, you reduce performance; if you allocate too much, you waste disk space.

The space the RDBMS needs varies primarily based on the total number and types of users supported, as well as the transaction mix and rate. Consult the RDBMS vendor's documentation for more information on these requirements.

The space required for Siebel data and indexes varies depending on what Siebel Business Applications functionality you will implement and the amount and nature of data supporting that functionality.

NOTE: The Siebel Servers in a Siebel Enterprise Server can connect to only one database. For example, you cannot configure both an Oracle and an IBM DB2 database for use by the same Siebel Enterprise Server.

To determine the size of the database required for a Siebel deployment

1. Determine the total number and types of users of Siebel Business Applications (for example, 500 sales representatives and 75 sales managers).
2. Determine the Siebel Business Applications functionality that you will implement and the entities required to support them. Usually, the largest entities are as follows:
   - Accounts
   - Activities
   - Contacts
   - Forecasts
   - Opportunities
   - Service requests
3. Estimate the average number of entities for each user (for example, 100 accounts for each sales representative) and calculate an estimated total number of records for each entity for your total user base.
4. Using standard sizing procedures for your specific database, calculate the average record size for each entity and multiply by the total number of records.

   Typically, these entities span multiple physical tables, all of which you must include in the row size calculation in order to determine the estimated data size for the largest entities.
5. Add additional space for the storage of other Siebel data. A rough guideline for this additional amount would be half the storage required for these key entities.
   - Indexes typically require approximately the same amount of space as data.
   - Factor growth rates into your total size calculation.
   - Factor a margin of error into your total size calculation.

Database Table Planning

In most implementations, the Siebel tables listed in Table 8 on page 69 and their corresponding indexes are either the most commonly used, or they can be large in some enterprise deployments.

For example, the tables S_EVT_ACT, S_CONTACT, and S_ORG_EXT are large in all enterprise-level deployments of Siebel Business Applications. These tables and indexes should be separated across devices.

As a general rule, indexes should be in a different table space and, if possible, on different physical devices from the tables on which they are created.

Siebel table spaces on an IBM DB2 database should be database-managed table spaces (DMS) rather than system-managed table spaces (SMS).

Table 8. Frequently Used and Largest Tables, Enterprise Customers

Table Names	Table Names	Table Names
S_ACCNT_CHRCTR	S_DOCK_TXN_LOGT	S_OPTY_POSTN
S_ACCNT_CO_MSTR	S_DOCK_TXN_SET	S_OPTY_PROD
S_ACCNT_POSTN	S_DOCK_TXN_SETT	S_OPTY_TERR
S_ADDR_ORG	S_ESCL_ACTN_REQ	S_OPTY_POSTN
S_ADDR_PER	S_ESCL_LOG	S_ORG_EXT
S_ASSET	S_ESCL_REQ	S_ORG_TERR
S_CALL_LST_CON	S_EVT_ACT	S_PARTY
S_CON_CHRCTR	S_EXP_ITEM	S_PARTY_PER
S_CON_TERR	S_EXP_RPT	S_PARTY_REL
S_ACCNT_CHRCTR	S_EXP_RPT_APPR	S_PARTY_RPT_REL
S_CRSE_TSTRUN	S_IC_CALC	S_POSTN_CON
S_CRSE_TSTRUN_A	S_IC_CALC_IT	S_PROC_REQ
S_CS_RUN	S_IC_CMPNT_EARN	S_PROD_BASELINE
S_CS_RUN_ANSWR	S_IC_TXN	S_PROD_CONSUME
S_CTLGCAT_PATH	S_IC_TXN_IT	S_PROD_SHIPMENT
S_CYC_CNT_ASSET	S_IC_TXN_POSTN	S_PROD_TARGET
S_DNB_CON_MRC	S_INVC_ITM_DTL	S_QUOTE_ITEM
S_DNB_ORG	S_INVLOC_ROLLUP	S_SRM_REPLY
S_DNB_ORG_SIC	S_INVOICE	S_SRM_REQUEST
S_DNB_UPDATE	S_INVOICE_ITEM	S_SRM_REQ_PARAM
S_DOCK_INIT_ITEM	S_INV_LGR_ENTRY	S_SRV_REQ
S_DOCK_TXN_LOG		

Database Recovery Planning

Follow the RDBMS vendor's recommendations on configuring the database for recovery in case of data corruption, hardware failure, or disaster.

Oracle Database Recovery Planning

Many companies today use RAID storage systems that make Oracle Database online redo log mirroring unnecessary.

If your organization does not use RAID storage systems, you should, at a minimum, mirror the redo log, as this is essential when a database goes through failure recovery.

Also, when redo logs are mirrored at the RAID storage system level, usually RAID 1 or RAID 0+1, there is usually no need to mirror them at the Oracle level, since the RAID controller assures that these volumes can always be recovered. Mirroring at the RAID level usually improves database performance, which is especially beneficial for read operations.

If you have the resources, you should mirror the Oracle control files as well. Otherwise, you can put the Oracle control files into a RAID 5 device, as it is not heavily accessed and disk performance is not a concern. The information it records, though, is very critical for the Oracle database. Any updates to the control file, such as the current System Change Number (SCN) or transaction tables, ripple across all members of the control file specification.

IBM DB2 Recovery Planning

Mirror the transaction log to guarantee database recovery if a single device fails. You must mirror the instance home directory, if resources are available. Hardware or operating system mirroring generally provides the best performance.

Database Physical Device Planning

To make sure that your database performs well, create at least one container for each available logical or physical disk device. You can use table spaces to place objects on multiple physical containers to promote parallel I/O. Spreading the data and index information across several containers (physical devices) can improve the performance of queries.

IBM DB2 Physical Device Planning

Data and log devices should reside on different disk spindles to reduce contention between random and serial I/O. All IBM DB2 devices should reside on different disk spindles to minimize I/O contention. When this approach is not possible, spread devices containing database objects that are often used together across different spindles. These objects include tables, their indexes, and commonly joined tables.

If you are using a high performance disk subsystem, you might choose a different physical device layout. Consult your DBA and the disk subsystem vendor for the optimal setup.

Physical Device Planning for UNIX Deployments

For UNIX database servers, all containers should reside on raw UNIX disk partitions, except the containers used for `LONG VARCHAR` data. Containers for `LONG VARCHAR` data should reside on the UNIX file system to take advantage of the operating system's buffering capabilities. To make sure that your database performs well, create one container for each available logical or physical disk device.

Data and log devices should reside on different disk spindles to reduce contention between random and serial I/O.

Microsoft SQL Server Physical Device Planning

Use filegroups for assigning database objects to one or more files within a filegroup for maximum performance of the Siebel Database. When you group objects, you have the ability to distribute a filegroup across multiple disks, thereby causing less resource contention.

If your enterprise does not require very high performance, based on the number of concurrent users, for example, using RAID devices and Microsoft's default setting may suffice. A database administrator must do the necessary sizing calculations to assess the performance requirements during the planning process.

Database RAID Array Planning

A database RAID array (redundant array of independent drives) can provide large amounts of I/O throughput and capacity, while appearing to the operating system and RDBMS as a single large disk (or multiple disks, as desired, for manageability). The use of RAIDs can greatly simplify the database layout process by providing an abstraction layer above the physical disks, while promoting high performance.

Performance of the RAID feature provided by the operating system may not be satisfactory. To obtain the best RAID performance, use the RAID support provided by your RAID vendor.

If a RAID Array Is Not Used

If a RAID device is not in use, even if space is at a premium, you must separate indexes with names ending in _P1 from the tables on which they are created. These tables are heavily used in joins.

If you will make frequent use of Siebel Enterprise Integration Manager (EIM), you may want to put the EIM tables and indexes (names starting with EIM_) on different devices from the Siebel base tables. Both tables are accessed simultaneously during EIM operations.

Microsoft SQL Server RAID Array Planning

Table 9 on page 71 describes a sample disk layout for a server dedicated to Microsoft SQL Server, where the database uses a single filegroup residing on a disk array. The use of a single RAID array for the database devices provides satisfactory performance in many cases without the administrative overhead of using individual filegroups.

Table 9. Microsoft SQL Server Recommended Disk Layout

Disk	Objects	Comments
Single mirrored	Windows OS	N/A
Single disk	Windows pagefile	Segregate for maximum performance.
Single mirrored	SQL Server logfile	Segregate sequential I/O for database performance.
3 to 5 disks (minimum) in a RAID configuration	Siebel Database data and indexes	Add as many spindles as required for performance and storage capacity.

If your enterprise requires the highest performance standards, you should place heavily used tables and their corresponding indexes, such as those listed under "Sizing the Database for a Siebel Deployment" on page 67, in a specific SQL Server filegroup within your database. By creating a filegroup on a specific disk or on multiple disks, you can control where tables and indexes in your database are physically located. For details, see "Database Physical Device Planning" on page 70.

When separating database objects into filegroups, you can avoid complex calculations by using Microsoft's recommended RAID disk layouts.

Your choice to use RAID devices or multiple filegroups to distribute database objects depends solely on how great your performance needs are. It is recommended that you work with your hardware vendor to determine the optimal RAID configuration for your specific requirements.

7 Siebel Client Deployment Planning

This chapter includes the following topics:

About Standard and High Interactivity Modes

In the Web browser, Siebel applications run in one of two modes: standard interactivity or high interactivity.

High interactivity is designed to provide Siebel applications with a user experience similar to that of traditional GUI-based Windows applications. High interactivity reduces the number of page refreshes (compared to standard interactivity) when a user interacts with the application, browses through records, and so on. This degree of interactivity is made possible by requesting data-only updates from the server. The application thus makes optimal use of network bandwidth.

For example, a high interactivity client does not require a page refresh for creating a new record. A user creates a new record by clicking New. A new row is created in a list dynamically, without a page refresh. The user enters the relevant data, then clicks outside of the record to implicitly commit the change, again without a page refresh.

Some of the features of the high interactivity framework are:

- Fewer page refreshes.
- Support for client-side scripting.
- Support for implicit commit. This feature enables automatic saving when a user steps off of a new or modified record.
- Other usability features, including:
  - Lists displayed in special applets
  - Drag-and-drop column reordering
  - Drag-and-drop file attachments
  - Keyboard shortcuts
  - Smart controls for calendar, calculator and currency
  - Applet scrollbars

The high interactivity framework requires the Microsoft Internet Explorer browser running in a Windows environment and uses both ActiveX controls and Java Runtime Environment (JRE). Deploying Siebel applications in high interactivity mode requires adhering to strict guidelines regarding the deployed operating system, Internet Explorer version and settings, and JRE.

For information about supported browser versions and about which Siebel applications use high interactivity mode, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

For detailed browser configuration information, see *Siebel System Administration Guide*.

High Interactivity Application Deployment Planning

High interactivity applications should be deployed in a way that maximizes local browser performance. For an overview of high interactivity mode, see "About Standard and High Interactivity Modes" on page 73.

Several factors influence this performance:

- **Default column display.** Users can select which columns to display in list applets. Deploy applications with the minimum number of columns set to display. Then allow users to select which columns to add. Displaying fewer columns improves performance by minimizing the number of Web templates and amount of data required to build views in response to user requests.
- **View layout caching.** Administrators can determine the number of views to be cached by the user's browser. When a view is cached, subsequent visits will cause a user request for only a data update. The Web templates required to build the view are retrieved from the browser cache. The Siebel Server does not rebuild them again. Caching views improves response time. Set the number of views to be cached as high as practical.
- **User login.** The first login is generally the most time-consuming. The client infrastructure caches the main components of the application on first login. Subsequent logins require far fewer resources. Cached objects remain on the client computer until the cache is cleared or a new version of the application configuration is available.
- **Browser version.** Use the latest supported version of Microsoft Internet Explorer in your application development, testing, and deployment environments. Besides fixes, the latest versions frequently include performance enhancements and improved ActiveX and Java Runtime Environment (JRE) support.

Standard Interactivity Application Deployment Planning

Standard interactivity applications do not allow users to select which columns to display in a view. They also do not support data-only user requests. All user requests require that the Siebel Server load page templates and rebuild each page for display.

For an overview of standard interactivity mode, see "About Standard and High Interactivity Modes" on page 73.

To maximize system performance, set the number of columns of data displayed in each view to the minimum needed. Also, when creating or modifying Web page templates, keep template designs as simple as possible.

Standard interactivity applications can be run on several types of Web browsers. Test browser performance for all the types of Siebel clients you will deploy. Investigate all browser settings that affect page display, cookie management, or page caching.

To reduce deployment administration costs, standardize on particular browsers for all users, where possible. Use the same browser types for development, testing, and production environments.

For information about supported browser versions and about which Siebel applications use standard interactivity mode, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

For detailed browser configuration information, see *Siebel System Administration Guide*.

8 Application-Level Deployment Planning

Some Siebel applications or product modules can be deployed in multiple ways. This chapter provides an overview of deployment options for these applications. Use this chapter to help make decisions about how to deploy applications across Siebel Servers.

For more information about application deployment planning, see *Siebel Performance Tuning Guide*, *Siebel System Administration Guide*, *Siebel Applications Administration Guide*, and other documentation on the *Siebel Bookshelf*.

This chapter includes the following topics:

Session Communications Server Components

Session communications refers to using Siebel Communications Server components to enable contact center agents or other users to handle interactive communications work items. For example, Siebel CTI supports this capability, enabling agents to handle voice calls using the communications toolbar.

Siebel Communications Server provides an application environment to support several kinds of communications activities for Siebel application users, including session communications (such as voice calls).

Key Siebel Server Components

Session communications are supported in the Siebel Server environment primarily by the following components:

- **Communications Session Manager (CommSessionMgr).** This server component manages interactive communications work items such as voice calls.
- **Application Object Manager (AOM).** This server component manages application sessions for end users who use the Siebel Web Client, including users who handle communications work items (agents). Interactive communication requests from agents typically go through AOM.
- **Server Request Broker (SRBroker).** This server component handles communications between the AOM and certain other Siebel Server components, including CommSessionMgr.

  For example, when a Siebel CTI agent makes a call through the communications toolbar, the request goes from AOM to CommSessionMgr by way of SRBroker.

  SRBroker is used whether CommSessionMgr runs on the same machine as the AOM, or on a different machine.

Additional Siebel Server Components

You may also be using the following Siebel Server components to manage session communications:

- **Communications Configuration Manager (CommConfigMgr).** Optionally, you may use this server component to cache communications configuration data.

Session Communications Performance Factors

Depending on your deployment, your agents may be handling phone calls (Siebel CTI) or work items of other communications channels, or some combination of these. Use the following factors to analyze system performance:

- **Inbound calls processed per hour.** This is the number of inbound calls (or other types of work items) processed per hour (or some other time period) by your communications infrastructure.
- **Outbound calls processed per hour.** This is the number of outbound calls processed per hour (or some other time period) by your communications infrastructure. (For outbound predictive dialer calls, only the calls that are answered and processed by Communications Server are relevant here.)

- **Number of user communications actions per minute (load).** This is the average number of communications-related user actions per minute, and the average think time between such user actions. Communications-related actions typically refers to actions performed using the communications toolbar.

  Longer think times mean less load on the Siebel Database and Siebel Server. Think time is an important factor in overall system load. Estimations should approximate actual user usage.

- **Number of concurrent communications users (agents).** This is the number of concurrent users of session communications features, typically contact center agents. This figure will be some percentage of the total number of concurrent users on the AOM.

- **Number of work items.** This represents the average number of inbound and outbound work items for each agent. This figure usually relates to your organization's service goals and is another important factor influencing performance. Some agents receive a large number of work items from ACD queues, or initiate a large number of work items. Supervisors or other users may be defined as agents but may receive only escalated work items, for example.

- **Volume of customer data.** This is the total volume of customer data. Data volume affects how quickly data can be retrieved for various purposes, such as to perform lookups for pop-up windows, route work items, or populate the customer dashboard. In many cases, data volume directly affects agents' response times. The volume of data should be realistic and the database needs to be tuned to reflect real-world conditions.

Third-Party Product Considerations

Review information presented in applicable third-party documentation for any requirements that affect your deployment. For example:

- Some CTI middleware software may place limitations on the number of agents that may be served at a single contact center site.
- Integration with ACD queues, predictive dialers, or other modules may affect your configurations, affect network traffic, or have other impacts.
- The capacity of your telephony link (between the ACD switch and the CTI middleware) may affect performance.

Session Communications Deployment Planning

Generally, you should run Siebel Communications Server components for session communications, such as CommSessionMgr, on the same Siebel Server machines as those running AOMs. In some cases, however, you must run CommSessionMgr on a different machine than the AOMs. These options are described in detail below.

CTI middleware generally runs on servers located at each contact center facility.

Running CommSessionMgr on AOM Machines

Generally, you should run Siebel Communications Server components for session communications on the same Siebel Server machines as those running AOMs. Such a topology allows the AOM load balancing mechanism to indirectly balance Communications Server load. CommSessionMgr loads are fairly light and do not, in themselves, present a reason to run this component on dedicated machines.

Set the Enable Communication parameter to TRUE for all AOMs to which your agents will connect. If you use load balancing, then configure all AOMs to which requests are distributed in the same way.

Running CommSessionMgr on Dedicated Machines

Sometimes you must run CommSessionMgr on a different machine than the AOM components.

CommSessionMgr must run on the same machine where the communications driver for your CTI middleware is running. If your driver requires a particular operating system, then you must install Siebel Server and run CommSessionMgr on a machine with that operating system. Communications drivers are required to be able to run on one of the supported Siebel Server platforms, as described in *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

Siebel Email Response Server Components

Siebel Email Response uses Communications Server components to enable contact center agents to read and respond to inbound email messages.

Key Server Components

Siebel Email Response is supported in the Siebel Server environment primarily by the following server components:

- **Communications Inbound Receiver (CommInboundRcvr).** This server component receives inbound work items and queues them for processing by Communications Inbound Processor. Work items may include email messages (for Siebel Email Response).
  - 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 email messages for some deployments of Siebel Email Response, Communications Inbound Receiver processes work items it has received. Communications Inbound Processor is not used.
- **Communications Inbound Processor (CommInboundProcessor).** This server component processes inbound work items that were queued by Communications Inbound Receiver.
- **Communications Outbound Manager (CommOutboundMgr).** This server component sends outbound email.

- **Siebel File System Manager.** This server component writes to and reads from the Siebel File System. It stores inbound messages prior to processing and stores attachments to inbound and outbound email messages.

Other Siebel Product Modules

In addition to Siebel Email Response, you may be using the following Siebel module:

Siebel Assignment Manager. You may use this module for routing email messages to agents.

Third-Party Email Server

Siebel Email Response works in conjunction with your third-party email server. Review information presented in documentation for your email server for any requirements that affect your deployment. For information about supported email servers, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

Siebel Email Response Performance Factors

The key factors that influence performance for Siebel Email Response deployments are as follows:

- **Inbound email messages processed per hour.** This is the number of inbound email messages processed per hour (or some other time period) by your communications infrastructure.

  Requirements for processing outbound messages are relatively minor and are tied to inbound message volume. However, you must also consider other usage of the CommOutboundMgr component or of the email system. For example, you may configure the Send Email command to send email through CommOutboundMgr.

- **Volume of customer data.** This is the total volume of customer data, including templates or categories, literature items, and so on. Template format (HTML or plain text) is a related factor.

Other factors include the size and complexity of inbound email messages and outbound replies.

Also relevant are user settings in the Outbound Communications section of the User Preferences screen, such as whether a reply contains the original message (Include Original Message in Reply setting), or whether HTML or plain text is an agent's default message format (Default Message Format setting).

Siebel Email Response coverage in this chapter focuses on inbound and outbound email processing. In a multichannel environment, session communications performance issues also apply.

Siebel Email Response Deployment Planning

Processing inbound email messages makes more demands on server resources, particularly CPU usage levels, than processing outbound messages.

A single machine must handle processing of inbound messages associated with a single response group.

If inbound message volume warrants it and if multiple server machines are available to run CommInboundRcvr and related components, then you should consider running CommInboundRcvr on a separate machine (or machines) from other Communications Server components.

Combining processing of messages for multiple email accounts in a single response group can make processing of inbound messages more efficient. However, if message volume is expected to grow, then limiting the number of email accounts processed by each response group will give you more flexibility to distribute processing across multiple servers, avoiding processing bottlenecks.

Siebel Configurator Server Components

Siebel Configurator allows users to interactively configure customizable products when ordering or generating a quote. Siebel Configurator uses a constraint-based solution engine that resides on the Siebel Server. This engine evaluates customer choices and generates product configurations that conform to business rules.

For more information about tuning Siebel Configurator, see:

- *Siebel Performance Tuning Guide*
- *Siebel Product Administration Guide*
- *Siebel System Administration Guide*

Remaining topics about Siebel Configurator in this chapter include:

- "Siebel Configurator Performance Factors" on page 84
- "Siebel Configurator Deployment Planning" on page 86

Siebel Configurator Components

Siebel Configurator is supported in the Siebel Server environment by the following components:

- **Application Object Manager (AOM).** The Siebel Configurator engine for product configuration is available within an AOM, such as Call Center Object Manager (SCCObjMgr_*lang*, such as SCCObjMgr_fra for French) for Siebel Call Center.
- **Siebel Product Configuration Object Manager (alias eProdCfgObjMgr_*locale*).** This is a special-purpose Object Manager component suitable for some Siebel Configurator deployments. For convenience, the component alias is generally referred to in this guide as eProdCfgObjMgr. This component contains the Siebel Configurator solution engine. It can be deployed on a separate Siebel Server from where Siebel Configurator sessions are invoked. The Configurator server in this type of deployment is also referred to as a remote or dedicated Configurator, which works in coordination with the invoking AOM.

  See later in this topic for locale-related requirements for a remote Configurator component.
- **Siebel File System.** This component stores cached object definitions for customizable product definitions in the CFGCache directory on the Siebel File System.

Siebel Configurator Architecture

Because product configuration may sometimes be computationally expensive, the configuration infrastructure provides flexible deployment options to suit different business needs. Topics in this section discuss considerations for choosing among different deployment options.

Before discussing Configurator parameters and where to set them, a brief overview will be presented of the Siebel Configurator architecture and of the various services in a Configurator deployment.

Figure 6 on page 83 shows detailed Siebel Configurator architecture and the interaction of various services with each other during run time.

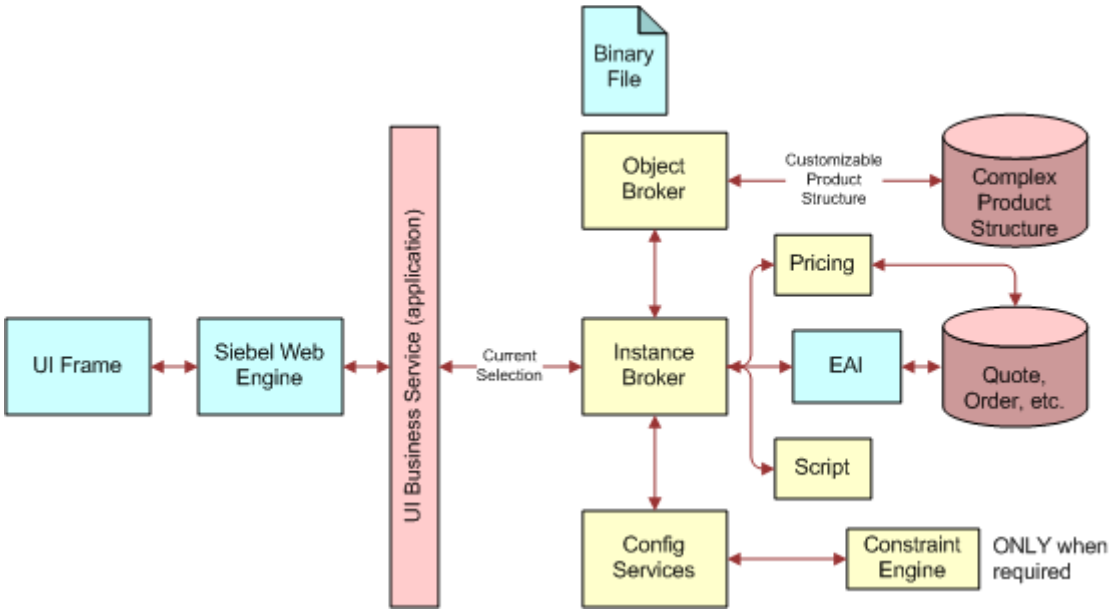

Figure 6. Detailed Siebel Configurator Architecture

The important services depicted in Figure 6 on page 83 are as follows:

- **UI.** The UI business service is a service that the Configurator uses to render the user interface by binding the customizable product structure with the templates and submitting it to the Siebel Web Engine for rendering to the client browser. The UI business service is the way the user interacts with the Configurator. A unique instance of this service is required for each user.
- **Instance Broker.** The Instance Broker is a service that interacts with the UI service and maintains all information about the current configuration of the customizable product that the user is configuring. This service interacts with other services in response to user requests during configuration, receives their responses, and serves as backup to the user through the UI service. The Instance Broker is accessed through a proxy service: either the Complex Object Instance Service business service or the Remote Complex Object Instance Service business service.
- **Object Broker.** The Object Broker is a service (Cfg Object Broker business service) that extracts the customizable product definition from the database for use by other configuration services.
- **Config Services.** This is a configuration service that consists of factories (defined below).

- **Factory.** The factory is a service that represents a translation of the customizable product definition that is retrieved by the Object Broker into a format a worker (defined below) can understand.
- **Constraint Engine or Worker.** The constraint engine, also called a worker, is a service that enforces all the rules associated with the customizable product. It validates all selections (interactive or batch) as they are made to ensure a valid configuration. A worker of a factory can be shared among different requests originating from the same AOM process.

For more information about elements of Siebel Configurator's internal architecture, including the Instance Broker and the Object Broker, see *Siebel Product Administration Guide*.

Locale-Related Requirements for Remote Configurator Components

The three-letter extension to the alias of the Siebel Product Configuration Object Manager component (*jpn* in the example of eProdCfgObjMgr_jpn) corresponds to the value for the Locale Code parameter (alias LocaleCode) associated with the invoking AOM. For a remote Configurator component you intend to invoke, the name of the component must follow this pattern.

The reason for this requirement is that data passed between the invoking AOM and the remote Configurator component is in a locale-specific format.

If Locale Code on the invoking AOM is set to a value that does not correspond to a language supported for Siebel applications, you must either change the Locale Code on the invoking AOM or create a new remote Configurator component with the required name. For example, assume the invoking AOM is SCCObjMgr_enu, but Locale Code is set to ENG rather than ENU. In this case, you must do *one* of the following:

- Change Locale Code to ENU in order to work with a remote Configurator component named eProdCfgObjMgr_enu.
- Create a remote Configurator component named eProdCfgObjMgr_eng.

In addition, note that the Language and Locale Code parameter settings for the remote Configurator component must match the parameter settings for the invoking AOM component.

For more information about the Language and Locale Code parameters, see *Siebel Global Deployment Guide*. For more information about creating and configuring server components, see *Siebel System Administration Guide*.

Siebel Configurator Performance Factors

For an overview of Siebel Configurator server components and other architecture elements, see "Siebel Configurator Server Components" on page 82.

Siebel Configurator performance contexts to consider include response times for:

- **Loading customizable products.** The time elapsed from the moment a user clicks Customize in a quote or order until the Configurator user interface for the customizable product is loaded and displayed to the user.

- **Responding to user selections.** The time elapsed from the moment a user makes a selection until Siebel Configurator returns a response, such as an update to the customizable product or a conflict message.

The performance factors below, particularly customizable product size and complexity, are relevant in both of these contexts:

- **Number of concurrent configuration users.** The number of concurrent users who access customizable product definitions. This figure will be some percentage of the total number of concurrent users on all applicable AOMs.

  Specifically, you would be concerned with the total number of configuration sessions per hour, and the average length of those sessions.

- **Size and complexity of customizable products.** The total size and complexity of each customizable product definition, particularly where multiple hierarchical levels, many constraints, and a complex user interface are defined.

  A major potential performance factor is custom scripting attached to update events on applicable business components, such as Quote, Quote Item, Quote Item Attribute, Order, Order Item, and Order Item Attribute.

- **Number of customizable products.** The number of customizable products accessed by users. It is assumed that each user accesses no more than one customizable product at one time. A given group of concurrent users may access multiple customizable products, however, each of which must have a separately cached factory. (An existing worker can be shared if one is available. Otherwise, internal performance mechanisms will generate a new worker.)

- **Use of SnapShot mode caching.** A feature that caches customizable products in memory, significantly reducing the amount of time required to load customizable products for each new user. SnapShot mode is particularly useful for improving performance when a product line has a small number of large, complex customizable products. For more information on SnapShot mode caching, see *Siebel Product Administration Guide*.

The response time of loading customizable products depends in part on the time taken to extract the customizable product definition from the database, and the time needed to instantiate all the services required to create the session. At the highest level, the response time for loading a customizable product would be best under the following circumstances:

- The customizable product definition is cached in memory and does not have to be extracted from the database.
- All of the services that are required are already available and do not have to be instantiated.

Caching of objects and services in memory can lead to significant improvement in load time performance for a configuration session. Ideally, everything would be cached, which would give the best possible load time performance. However, that is not possible because the RAM available on the server is limited. For this reason, a caching strategy must be devised for each deployment. To enable administrators to implement the caching strategy that is best suited for their deployment, several switches in the form of server parameters have been provided.

Siebel Configurator Deployment Planning

Siebel Configurator deployment planning must take into account the considerations detailed in the following topics:

About Deployment Topology for Siebel Configurator

This topic is part of "Siebel Configurator Deployment Planning" on page 86.

There are two major topology approaches to deploying Siebel Configurator:

- Running Siebel Configurator in an Application Object Manager (AOM) component.
- Running Siebel Configurator on one or more dedicated Siebel Servers.

For details, see "Siebel Configurator Deployment Topology Options" on page 90.

Running Siebel Configurator in an AOM Component

You can run Siebel Configurator in the AOM component, such as a language-specific Call Center Object Manager component.

If a small number of concurrent users require configuration sessions, or there is a small number of customizable product definitions, then this deployment option may yield reasonable performance and make the most effective use of your hardware resources.

Running Siebel Configurator on Dedicated Servers

You can run Siebel Configurator on one or more dedicated Siebel Server machines using a server component called the Siebel Product Configuration Object Manager (eProdCfgObjMgr).

Such servers are sometimes referred to as remote servers, because they are remote to the machine on which the AOM is running. In general, this guide uses the term dedicated servers.

If a large number of concurrent users require configuration sessions, or there are a large number of customizable product definitions, then using one or more dedicated Configurator servers may yield the best performance and make the most effective use of your hardware resources.

Possible variations on this deployment strategy include:

- Running one eProdCfgObjMgr component with one AOM component
- Running multiple eProdCfgObjMgr components with one AOM component
- Running one eProdCfgObjMgr component with multiple AOM components
- Running multiple eProdCfgObjMgr components with multiple AOM components

About Siebel Configurator Caching

This topic is part of "Siebel Configurator Deployment Planning" on page 86.

Object Broker

The Object Broker is a service that extracts the customizable product definition from the database for use by other configuration services. To improve performance, the Object Broker also maintains a cache of objects in the memory to minimize interaction with the database. You can configure the number of objects that are cached by setting the eProdCfgNumOfCachedObjects server parameter. Normally, the size of the cache is quite small. Different users can share the same object cache.

Factory

The factory is a service that creates a translation of the customizable product definition that is retrieved by the Object Broker into a format the worker (detailed below) can understand. Each factory can serve multiple users at run time. Factories are customizable product-specific, meaning that each customizable product requires its own unique factory. Factories can be cached in memory. You can configure the number of factories that are cached by setting the eProdCfgNumbOfCachedFactories server parameter.

Worker

The worker is shared: in other words, a session only locks a worker during a single configuration request. In between requests, the worker can serve additional configuration sessions.

While the relative number of workers now required to support a user base depends on the time between clicks of particular scenarios and the number of clicks that require worker interaction, the general guideline is that a given worker can now support two to three concurrent users for the same customizable product.

SnapShot Mode

SnapShot mode is a server setting that allows the Configurator to create and execute using cached objects, factories, and workers. If this setting is not chosen, each user configuration session would be associated with the following:

- Extraction of the customizable product definition and all objects associated with it from the database

- Creation of a factory for the customizable product
- Creation of a worker for the user session

If SnapShot mode is chosen, depending upon the parameter values set up, objects, factories, and workers would be cached in memory and would be first examined if they can be used to initiate a user session. Only if the existing cache cannot support the user session will new objects be extracted or factories or workers created.

Considerations About SnapShot Mode

Here are some things to remember about SnapShot mode:

- SnapShot mode determines the upper bound of caching.
- The cache is created as one goes along. The first user request results in the first set of cache data being created. The second user may end up using the same cache because the user wants to configure the same customizable product that the first user configured. Meanwhile, the third user, who wants to configure a different product, creates a new cache, and so on.
- Re-use of cache data occurs only for requests originating from the same Application Object Manager process, for any Configurator deployment.

Determining Factory and Worker Size

This topic is part of "Siebel Configurator Deployment Planning" on page 86.

Factory Size

As a rule, you can assume that the size of the factory at run time is 75% of the incremental memory used when instantiating the customizable product.

For example, factory size equals 75% of (Y minus X), which equals .75 times (40 minus 20), which equals 15 MB.

Worker Size

Worker size varies during run time. Generally, the worker size increases as selections are made. To size the worker, take the maximum memory observed and subtract the factory size from it.

For example, worker size equals (Z minus X) minus factory size, which equals (50 minus 20) minus 5, which equals 25 MB.

Object Cache Size

Because this is normally quite small (for example, 500 KB), you can ignore it in your calculations.

Example of Sizing the Cache with SnapShot Mode

This topic is part of "Siebel Configurator Deployment Planning" on page 86.

Assumptions

The requirement is to support 5000 concurrent Siebel Call Center users. Among them, at any time, 100 users use Siebel Configurator. This means:

- The enterprise needs to support 5000 concurrent Call Center users.
- Of these 5000 Call Center users, 100 are expected to use the Configurator concurrently.
- There is only one customizable product in the product portfolio.

Sizing

Because all caching and services are specific to the AOM process on a Siebel Server, first you must estimate the size of the Call Center deployment. (The numbers below are used for example only, and are not indicative of Call Center sizing.)

- Assume you are supporting the 5000 Call Center users on eight application servers (each a Pentium 4 machine with 4 CPUs and 4 GB of memory), with each server handling 625 users.
- Each Siebel Server runs with 25 AOMs, with each AOM supporting 25 users.

To support cached objects, factories, and workers for all 100 users, the following conclusions can be drawn:

- At least one factory needs to be cached for every Object Manager process. This means you must cache 25 factories for each server or one for each AOM.
- To support all 100 concurrent users to get a cached worker, you must cache, at a minimum, 100 workers across the Enterprise. At the same time, at least one worker should be cached for each AOM process. This means you must cache 25 workers for each server or one for each AOM.

In the example above, the cache size in this case for each AOM equals the size of the factory cache plus the size of the worker cache. Expressed as a formula, it looks like this: 5 plus 25 equals 30 MB for each AOM. Therefore, the Configurator cache requires a total of 30 times 25, which equals 750 MB for each server.

The server parameters would be set as follows for each Siebel Server (AOM):

- eProdCfgSnapshotFlg: True
- eProdCfgNumOfCachedObjects: 1000
- eProdCfgNumbOfCachedFactories: 1
- eProdCfgNumbOfCachedWorkers: 1

Observations About Sizing

From the sizing exercise above, it is clear that the Configurator cache needs to be actively managed for best performance using appropriate resources. It is extremely important to go through the exercise of sizing the cache. In some cases, the cache requirements may be such that they require additional application servers to fully support all users with good response times for load time.

In addition to the preceding calculation, in some situations it is appropriate to set the number of workers according to how much memory is available once enough memory has been allocated to the factory cache and application overhead. The details of this calculation are specific to an individual implementation's average factory size, average worker size, and average AOM process size. The average AOM process size depends on the number of AOM processes, the total memory available, and the maximum process size for the operating system being used.

For additional assistance in this area, contact your Oracle sales representative for Oracle Advanced Customer Services to request assistance from Oracle's Application Expert Services.

Siebel Configurator Deployment Topology Options

This topic is part of "Siebel Configurator Deployment Planning" on page 86.

As mentioned earlier, Siebel Configurator offers the flexibility of different deployment topology options. These options are as follows:

- Deploy Configurator to run on the same machine as the base application server, as shown in Figure 7 on page 90.

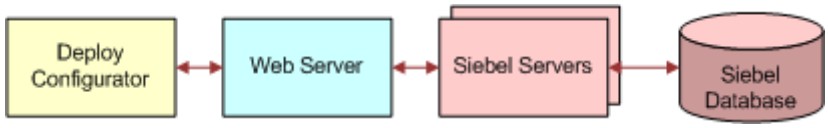

Figure 7. Run Configurator on base application server

- Deploy Configurator to run on a different machine than the base application server, as shown in Figure 8 on page 90.

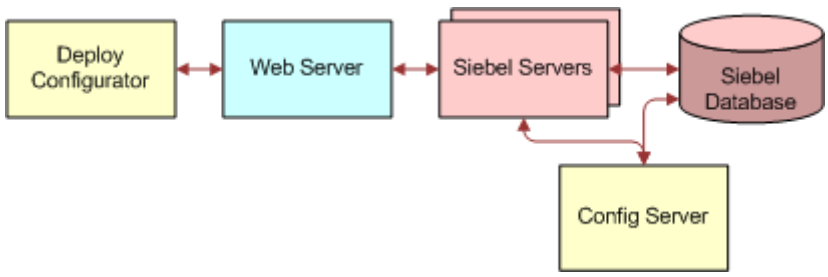

Figure 8. Run Configurator on dedicated application server

- Deploy multiple instances of Configurator on multiple dedicated application server machines, as shown in Figure 9 on page 91.

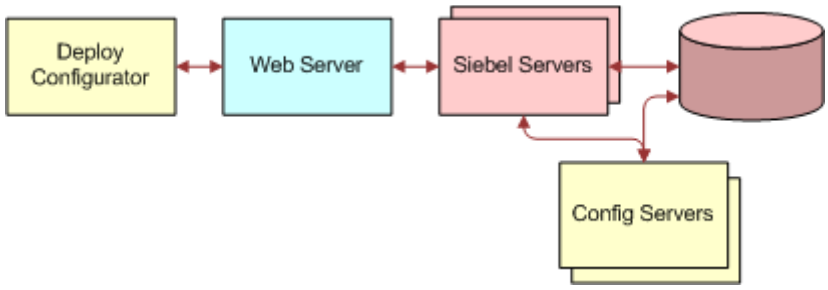

Figure 9. Run multiple Configurator instances on multiple dedicated application servers

In many cases, the option of deploying Configurator on a different application server machine may result in a much better use of resources due to pooling effects. Considering the example examined and sized from the preceding topic, the result was a sizing of one factory and one worker to be cached for each AOM on each server. The implications of this across the whole enterprise are as follows:

- The number of AOMs on each server was 25, so each server will cache 25 factories and 25 workers.
- Across the whole enterprise of eight application servers, this translates to 25 times 8, which equals 200 factories and 200 workers to be cached.
- In memory terms:
  - 200 times 5, which equals 1000 MB for caching factories, and
  - 200 times 25, which equals 5000 MB for caching workers

  This results in a total memory usage of 6000 MB across the enterprise.

Because the requirement is to support 100 concurrent users, many of these factories and at least 100 of these workers are idling at any time. The large amount of cache in this case is because there is no way to know in advance which AOM process the user who is configuring is connected to. For this reason, caching must be done across all AOMs.

There are other problems with this scenario. For instance, what happens when two users are connected to the same AOM process and both want to configure? In this case, the second user has to create a worker, causing performance issues for that user. Also, if there are multiple users configuring on the same machine, the server might run out of memory.

Consider another scenario. Assume that the enterprise has customizable products that these users might be configuring, with 50 concurrent users configuring each customizable product. However, because there is no way to know in advance which AOM process these users might be connected to, you would have to cache two factories and two workers for each AOM, which would be a less-than-satisfactory solution.

Example of Deployment Sizing with a Dedicated Configurator Server

This topic is part of "Siebel Configurator Deployment Planning" on page 86.

Consider the sizing example for the deployment option of running the Configurator on the same server as the application server (see "Example of Sizing the Cache with SnapShot Mode" on page 89). Size it instead for the deployment option of Configurator running on a separate server.

Assumptions

The requirement is to support 5000 concurrent Siebel Call Center users. Among them, at any time, 100 users use the Configurator. This means:

- The enterprise needs to support 5000 concurrent Call Center users.
- Of these 5000 Call Center users, 100 are expected to use the Configurator concurrently.
- There is only one customizable product in the product portfolio.

Sizing

Because all caching and services are specific to the AOM process on a Siebel Server, first you must estimate the size of the Call Center deployment. (The numbers used below are an example only and not indicative of Call Center sizing.)

- Assume you are supporting the 5000 Call Center users on seven application servers (each being a Pentium 4 machine with four CPUs and 4 GB of memory), with each server handling 720 users.
- Each application server itself is run with 25 AOMs, with each AOM supporting 25 users.
- Assume that one server has been configured to run the Configurator that will support the 100 users.
- The Configurator server is configured to run with four AOMs, with each AOM supporting 25 users.

To support cached objects, factories, and workers for all 100 users, the following conclusions can be drawn:

- At least one factory needs to be cached for every AOM process. You must cache four factories for the Configurator server or one for each AOM.
- To support all 100 concurrent users to get a cached worker, you must cache, at a minimum, 100 workers across the Configurator server. This means you must cache 25 workers for each AOM on the Configurator server.

In the example above, the cache size in this case for each AOM equals the size of the factory cache plus the size of the worker cache. Expressed as a formula, it looks like this: (5 times 1) plus (25 times 25) equals 630 MB for each AOM. Therefore, the Configurator cache requires a total of 4 times 630, which equals 2520 MB for each server.

The server parameters would be set as shown in Table 10 on page 93 for the Siebel Servers running the AOMs and the Configurator servers:

Table 10. Example Parameter Settings for Siebel Configurator Server

Parameter Name	Setting	Where to Set
eProdCfgServer	Name of the Siebel Server running the Configurator.	Set On: Each Siebel Server running the AOM (see Table 11 on page 94)
eProdCfgSnapshotFlg	True	Set On: Each Siebel Server running the AOM and each Configurator server
eProdCfgNumOfCachedObjects	1000	Set On: Configurator server
eProdCfgNumbOfCachedFactories	1	Set On: Configurator server
eProdCfgNumbOfCachedWorkers	25	Set On: Configurator server

This type of deployment across an enterprise with eight servers, one a dedicated server to support Configurator, requires 2520 MB of cache. This figure is much lower than the 6000 MB required for the eight application server deployment option. Choosing this deployment option makes better use of the cache.

Moreover, since the Configurator server is configured to allow only 25 connections to each AOM, there would never be a case where a user does not find a cached worker to work with. In a scenario with multiple customizable products, this deployment would be much more efficient in terms of memory usage.

Server Settings for Dedicated Configurator Server Deployment Mode

Table 11 on page 94 shows server settings for dedicated (remote) Configurator server deployment mode. Except where noted, set these parameters on the AOM component.

Table 11. Parameter Settings for Dedicated Configurator Deployments

Parameter Name	Display Name	Data Type	Default Value	Description
eProdCfgRemote	Product Configurator - Use Remote Service	Boolean	False	Setting to determine if Configurator is running on a different server from the AOM. On the AOM: set it to TRUE when running a dedicated Configurator server. On the dedicated Configurator server: leave this set to FALSE.
eProdCfgServer	Product Configurator - Remote Server Name	Text	None	Name of the Siebel Server on which you are running a dedicated Configurator server. If you are using multiple dedicated Configurator server, separate the entries with semi-colons (;).

Table 11. Parameter Settings for Dedicated Configurator Deployments

Parameter Name	Display Name	Data Type	Default Value	Description
eProdCfgTimeOut	Product Configurator - Time Out of Connection	Integer	20	Setting in seconds that determines for how long the Siebel Server would try to initiate a connection with the remote Configurator server before returning an error to the user.
eProdCfgKeepAliveTime	Product Configurator - Keep Alive Time of Idle Session	Integer	900	Setting in seconds to determine the maximum interval of inactivity during a configuration session. If the interval of inactivity reaches this value, the user session is ended and the worker returns to the pool. If this parameter is not set, an infinite interval is assumed. Set this parameter on the AOM only. It does not apply on the remote Configurator server. **NOTE:** On the remote Configurator server (eProdCfgObjMgr component), set the parameter ConnIdleTime to a value like eProdCfgKeepAliveTime plus 1 second.

Siebel Workflow Deployment Planning

Siebel Workflow lets you define, manage, and enforce your business processes or workflows. It allows you to design complex workflow processes and automate the enforcement of business policies and procedures. This module includes the following:

- **Workflow Processes.** This module lets you define your company's business processes using a familiar flowcharting interface. A workflow process consists of one or more process steps, such as start steps, subprocesses, decision points, and tasks. You configure workflow processes using the Process Designer in Siebel Tools.
- **Workflow Policies.** This module lets you define policies that can act as triggers to execute a process. A policy consists of conditions and actions. When policy conditions are met, the policy action executes the relevant process.

- **State Models.** This module is used for defining business object states and state transitions.

For information on using and administering Siebel Workflow, see *Siebel Business Process Framework: Workflow Guide*.

Each user request to the Workflow Process Manager starts a new thread. However, sessions for Object Manager components (such as EAI Object Manager or Application Object Manager) that may invoke workflow processes are cached and reused for subsequent requests. When you size a system, the maximum number of workflow tasks you expect to have active at a given time helps determine the maximum number of Object Manager sessions created for Siebel applications.

The exact CPU and memory consumption of each task depends on the actions performed in your workflow processes. To estimate CPU and memory consumption in your production environment, run a single task, measure its resource consumption, and make an estimation based on your maximum concurrent sessions. Take session caching into account when making these measurements.

If you need a large number of sessions, you may want to run Workflow Process Manager on multiple Siebel Server machines. You can then load-balance requests across the Siebel Servers. If you plan to run a significant number of tasks per server (such as 100 or more), you may also want to run multiple multithreaded processes.

If you are going to run several different types of workflows, run each type in a separate process. Doing so makes it easier to monitor the overall CPU and memory usage of each process type.

The number of multithreaded processes and the number of tasks for each process are controlled through the parameters MaxMTServers (Maximum MT Servers), MinMTServers (Minimum MT Servers), and MaxTasks (Maximum Tasks).

These parameters are for each Siebel Server. For example, MaxMTServers refers to how many multithreaded processes to run on each Siebel Server machine.

Business Integration Manager and Business Integration Batch Manager have been deprecated, so if you were using either one in your business processes you need to replace them with Workflow Process Manager or Workflow Process Batch Manager, respectively.

If a workflow policy action is set up to use the Run Integration Process workflow policy program, which invokes the Business Integration Manager (BusIntMgr) component, create a new workflow policy action using the Run Workflow Process workflow policy program. The new workflow policy action will instead invoke Workflow Process Manager. Delete the old workflow policy action from the workflow policy, add the new workflow policy action to the workflow policy, and restart Workflow Monitor Agent tasks.

For detailed information about server components, see *Siebel System Administration Guide*.

Planning Batch Processing When Using Siebel Remote

Long-running batch jobs can create transaction gaps in the Master Transaction Log for the Siebel Remote module for Oracle's Siebel Business Applications. If the wait-time for the missing transactions expires, the Transaction Processor component skips the missing transactions. The skipped transactions are not routed to mobile users.

Batch jobs could be performed by Siebel Assignment Manager, Siebel EIM, or other components.

For more information on understanding gaps in the Master Transaction Log, see 477585.1 (Article ID) on My Oracle Support. This document was previously published as Siebel Technical Note 499.

For an example with Siebel Assignment Manager, gaps in the Master Transaction Log can occur as follows:

1. Assignment Manager is processing a batch of transactions.
2. Assignment Manager obtains a group of transaction IDs. These are issued in numeric, sequential order.
3. Assignment Manager then commits these transactions. This process takes several minutes.
4. In the meantime, another process obtains a transaction ID.
5. The process commits the transaction and writes it to Siebel Remote's Master Transaction Log. A sequence gap is created because the Assignment Manager transactions have not yet been written to the Master Transaction Log.
6. Transaction Processor detects the gap and waits a specified period called the wait-time (default is 600 seconds).
7. The wait-time expires before Assignment Manager completes the commit and writes the missing transactions to the Master Transaction Log.
8. When the wait-time expires, Transaction Processor skips the missing transactions and moves on to the transaction from the other process.
9. Transaction Processor logs information about the missing transactions. The Assignment Manager transactions are not routed to mobile users, even though they are later written to the Master Transaction log.

Conditions That Can Cause Missed Transactions

The following conditions in Assignment Manager can cause increased commit times. Longer commit times increase the risk that the Transaction Processor wait-time will expire before the commit occurs, and Transaction Processor will not process all the transactions in the transaction log.

Increasing the Assignment Manager Batch Commit Parameter

The default batch commit size (BatchSize) for Assignment Manager is 100. After processing 100 rows, transactions are committed to the database. If the batch commit size is increased, this increases the risk of exceeding the wait-time.

Increased Number of Batch Assignment Threads

When multiple Assignment Manager threads are logging transactions, this creates a latency in accessing the transaction log table. This latency increase the risk of exceeding the wait-time.

Complicated Assignment Rules

When Assignment Manager has to resolve complicated assignment rules, this can increase commit times. Longer commit time together with the conditions above can increase the risk of exceeding the wait-time.

Avoiding Missed Transactions

To avoid exceeding the Transaction Processor wait-time during batch processing, adopt the following best practice recommendations. Experiment with applying them in combination to achieve the best performance while minimizing the risk of exceeding the wait-time.

Monitor the Transaction Processor Logs

As you apply the best practices techniques described below, use the Transaction Processor logs to see the result and help you optimize system performance.

Transaction Processor writes these warning messages to its log file when it skips transactions:

```
GenericLog: GenericError: 0003-11-18 17:04:51

WARNING: A transaction gap has been detected after transaction 122.

Probable Cause: There maybe long-running transactions in your system which are not
committing transactions within the specified duration (600 sec)

Recommendation: Reduce the batch size of your transactions. This will allow the
transactions to be committed to the database within the wait-time window.
```

If skipped transactions occur while a batch job is running, investigate the cause. The skipped transactions may not have been routed to mobile users. If so, mobile users may have to re-extract the database.

Set a Lower BatchSize for Assignment Manager

Setting a lower BatchSize value reduces the number of records processed before each commit. Reducing this figure reduces the commit times and the risk of exceeding the wait-time. If your performance goals require you to increase the BatchSize parameter, do so only after analyzing the number of Assignment Manager threads that you have under average and peak workloads. The fewer Assignment Manager threads, the higher you can set the BatchSize parameter.

You can get performance statistics on Assignment Manager threads by raising Assignment Manager's log level. For information on raising the log level, see *Siebel System Administration Guide*.

Serialize Batch Jobs

Consider staggering the start time of batch jobs. Running batch jobs in staggered or serial order can reduce the risk of exceeding the wait time.

Index

T

U

W