Deployment Planning Guide

Version 7.8

February 2005

Contents

1 What’s New in This Release

What’s New in Deployment Planning Guide, Version 7.8

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

Table 1. New Product Features in Deployment Planning Guide, Version 7.8

Topic	Description
“Siebel Configurator Server Components” on page 81	Added deployment planning guidelines including sizing information and deployment options.
“About Siebel File System” on page 16	Added best practices information.
“Specific Failures and Associated Impact” on page 47	New topic.
“Mapping Siebel Deployment Elements to Platforms” on page 35	Added additional information on distributing Siebel components across multiple servers.
“Server Clustering Planning” on page 61	New topic.

2 Siebel Architecture Overview

This chapter includes the following topics:

Building Blocks of a Siebel Deployment

Figure 1 shows an example of the elements in a Siebel deployment. A brief description of these elements appears in Table 2 on page 11.

The current release may not support specific database and operating system platforms, as well as certain combinations of them. For a list of all operating system platforms and RDBMS products supported by this release, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

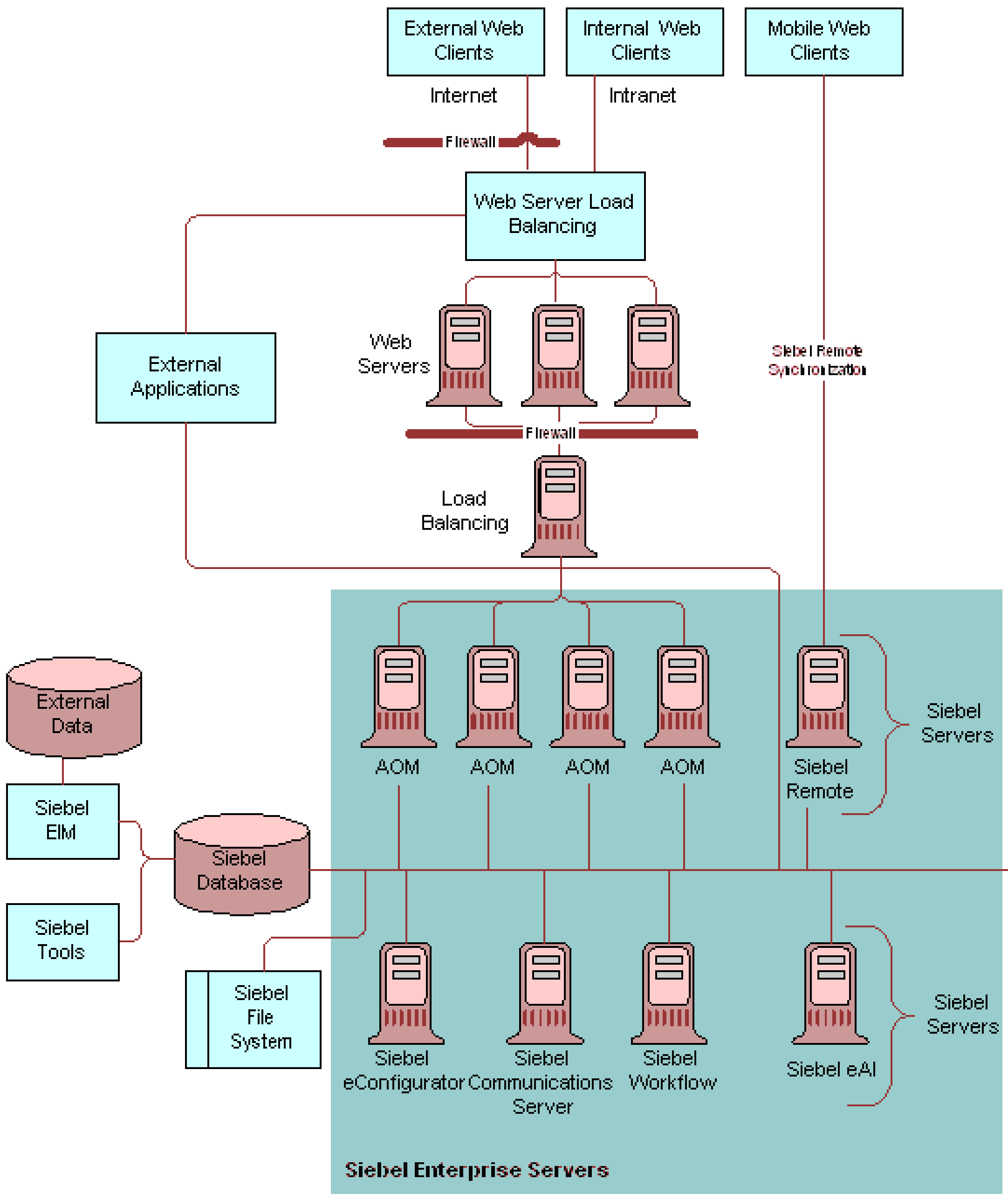


Figure 1. Example of a Siebel Deployment

Table 2. Siebel Deployment Elements

Entity	Description
Siebel Web Clients	Includes the following client types: ■ Siebel Web Client ■ Siebel Mobile Web Client ■ Siebel Wireless Client ■ Siebel Handheld Client
Siebel Web Server Extension (SWSE)	Installed on third-party Web server. 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 Load Balancing	The two options for Siebel Server load balancing are Siebel load balancing and third-party HTTP load balancers. Siebel load balancing is part of the Siebel Web Server Extension (SWSE). When you install the SWSE, the installation 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.
Siebel Gateway Name Server	Functions as a name server and stores Siebel Server configuration 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 that stores the data and physical files used by Siebel clients and Siebel Enterprise Server.
Siebel deployment	All of the elements required to deploy Siebel applications. This includes the Siebel Enterprise Server, Siebel Servers, Siebel database, Siebel Gateway Name Server, Siebel Web Server Extension, and related components such as third-party HTTP load balancers.

Table 2. Siebel Deployment Elements

Entity	Description
Siebel Enterprise Integration Management (EIM) and Siebel Enterprise Application Integration (EAI)	Allows importing and exporting of data from other databases to the Siebel database.
Siebel Tools	Provides an object-oriented, Windows-based environment for developing or modifying Siebel applications, business services, and other Siebel objects.

About Siebel Web Clients and Web Server Extension

This section lists the types of Siebel Web Clients.

Siebel Web Client

Siebel Web Client runs in a standard browser on the end user's client computer. ActiveX controls and JavaScript routines are downloaded to the browser automatically when it runs Siebel applications in Siebel high interactivity mode. 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 computer.

Other considerations about the Siebel Web Client are as follows:

- **Installed software.** No additional application software is required on the client. Requires only a Web browser.
- **Application connection.** This is a connection through a Web server to the Siebel Enterprise Server. Applications run on Siebel Server and forward pages to the client.
- **Database connection.** This is a connection through the Siebel Server to the remote Siebel database. No Siebel database or database client is installed on the client.

Wireless Client

Siebel Wireless 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 device communicates using wireless application protocol (WAP) to a wireless gateway server. The wireless gateway server translates HTTP messages to WAP. The Siebel interface is rendered on the mobile device using wireless markup language (WML). Specific XML- and HTTP-based wireless browsers are also supported. For a list of Siebel Business Applications that support wireless access, see the *Siebel Wireless Administration Guide*. For a list of supported wireless browsers, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

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.
- **Application connection.** Applications run on each client. Applications display in a Web browser.
- **Database connection.** A Siebel database and Siebel File System are installed on each client. Applications access the client’s local database.

  Users periodically synchronize the client’s local database and Siebel File System with a remote Siebel Enterprise Server's Siebel database and Siebel File System. Users synchronize data using Siebel Remote. Siebel Remote allows users to connect directly to the remote Siebel database and Siebel File System without going through the Enterprise Web servers or Siebel Servers.

  The Mobile Web Client runs in a standard browser on the end user's client computer, such as a laptop.

Siebel Handheld Client

The Siebel Handheld Client is a streamlined version of the Siebel Mobile Web Client. It includes only the functionality required by end users’ field technicians. 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 third-party Web servers. It identifies requests for Siebel information 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 load balancing module. This module provides round-robin load balancing for Application Object Managers running on Siebel Servers.

You must also install all language packs installed on Siebel Servers on your Web servers. However, you do not need to install all language packs on all Web servers. If you do not, you must provide a way to route user requests to Web servers that have the correct language support.

About Siebel Enterprise Server and Siebel Server

The Siebel Enterprise Server is a logical grouping of Siebel Servers that connect to one Siebel database. The Siebel Servers in a Siebel Enterprise Server are configured, managed, and monitored as a single logical group, allowing the Siebel administrator to start, stop, monitor, or set server parameters for all Siebel Servers within the Siebel Enterprise Server.

Siebel Server

The Siebel Enterprise Server is composed of one or more Siebel Servers. Siebel Servers function as application servers and are 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 components start tasks automatically in response to user requests. The tasks end when the user ends the session. Examples of interactive mode tasks are the Synchronization Manager and all Application Object Managers (AOMs).
- **Background mode.** Background components handle background processing tasks. Typically, background tasks are called by interactive mode tasks. Background tasks run until explicitly shut down. Examples of background tasks are 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 mode components are Database Extract and Enterprise Integration Manager (EIM).

Many of the Siebel Server components can operate on multiple Siebel Servers simultaneously. This allows Siebel applications to scale across many Siebel Servers to support large numbers of users.

Other Siebel Server components provide additional functionality besides application support. This includes 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)

This server component provides load balancing for multiple Application Object Managers (AOMs) running on the same Siebel Server.

Siebel Server Implementation

The Siebel Server runs as a system service under Windows and 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.

You can configure interactive mode and batch mode components to run as multiple processes or as multithreaded processes. Background mode components run as multiple processes only.

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

Application Object Manager (AOM)

One of the most important types of server components is the Application Object Manager (AOM). These server components run in interactive mode. They process user requests and are application- or service-specific. For example, the Siebel Employee Relationship Management component group contains the Employee Relationship Object Manager. This AOM provides the session environment in which this application runs.

AOMs also contain 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 database server.
- 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.

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 uses the Siebel Server to communicate with the Siebel Database, the Web server (through the SWSE), and other Siebel Enterprise Server 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 AOMs.

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, it clears this information 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 information changes, such as during the installation or configuration of a Siebel Server, it is written to the configuration file on the Name Server.

In a production environment, there can be only one Gateway Name Server installed per environment. Do not share the same Gateway Name Server for your 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.

Language Pack Installation

You do not need to install all the languages that your Siebel deployment may run on the Siebel Gateway Name Server. However, the Gateway Name Server installation includes utilities used for Siebel Server administration. Siebel administrators only see some server administration error messages in the languages that have been installed on the Gateway Name Server.

About Siebel File System

The Siebel File System is a shared file system directory. 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.

Siebel File System Best Practices

The following lists gives 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.

- When using a large number of files in the Siebel File System (300,000 or more) in a Windows NTFS folder, you should disable short filename generation. For more information, see http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/prodtechnol/winxppro/reskit/prkc_fil_tdrn.asp.
- During normal operation of the Siebel Enterprise, 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 binary (bin) subdirectory within the Siebel Server root directory. You can find more information on the use of SFSCLEANUP in the *Siebel System Administration Guide*.
- There are no 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 as part of an independent file server farm. For small-to-medium deployments, a typical scenario would 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.

For more information on the Siebel File System, see the *Siebel System Administration Guide*.

About 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.

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

Load Balancing Before Siebel 7.7

Before Siebel 7.7, Siebel Systems implemented server load balancing using a third-party software product, Resonate Central Dispatch. When multiple Siebel Servers ran the same Application Object Manager, Central Dispatch distributed server requests across the Siebel Servers. Siebel Servers were integrated with Central Dispatch to maintain session continuity.

Load Balancing as of Siebel 7.7 and Later

For Siebel 7.7 and later releases, Siebel Systems supports two methods for implementing Siebel Server load balancing:

- The first method is Siebel-provided load balancing, called Siebel 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 load balancing instead of third-party HTTP load balancers.

- Siebel Systems has certified a number of third-party, hardware-based HTTP load balancers for use in a Siebel deployment. For a list of these, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

  Certification means Siebel Systems has tested interoperability with the load balancer extensively, and specific configuration instructions are available. Siebel Technical Support will work with customers on configuration and interoperability issues specific to Siebel deployments.

  If customers are using a noncertified load balancer and encounter load balancing issues, they should contact the third-party load balancer vendor directly. If customers encounter a load balancing or connectivity issue, they should try to reproduce the issue with Siebel load balancing to isolate the cause. It is recommended that you use a certified load balancer to minimize potential compatibility issues. Although Siebel applications are designed to work with standard, third-party HTTP applications, customers should perform compatibility testing before using an uncertified load balancer.

  Configuration and troubleshooting information for third-party load balancers is available on Siebel SupportWeb as Technical Notes.

Siebel Connection Broker

On each Siebel Server, the Siebel Connection Broker (SCBroker) provides intraserver load balancing. SCBroker distributes server requests across multiple instances of Application Object Managers running on the server.

Discontinuation of Resonate Support

For Siebel Servers, Resonate is classified as an uncertified third-party load balancer, and the following are no longer supported:

- Automatic registration of Resonate rules
- Siebel Server parameters for Resonate registration

Figure 2 shows an example of third-party load balancing.

About SISNAPI

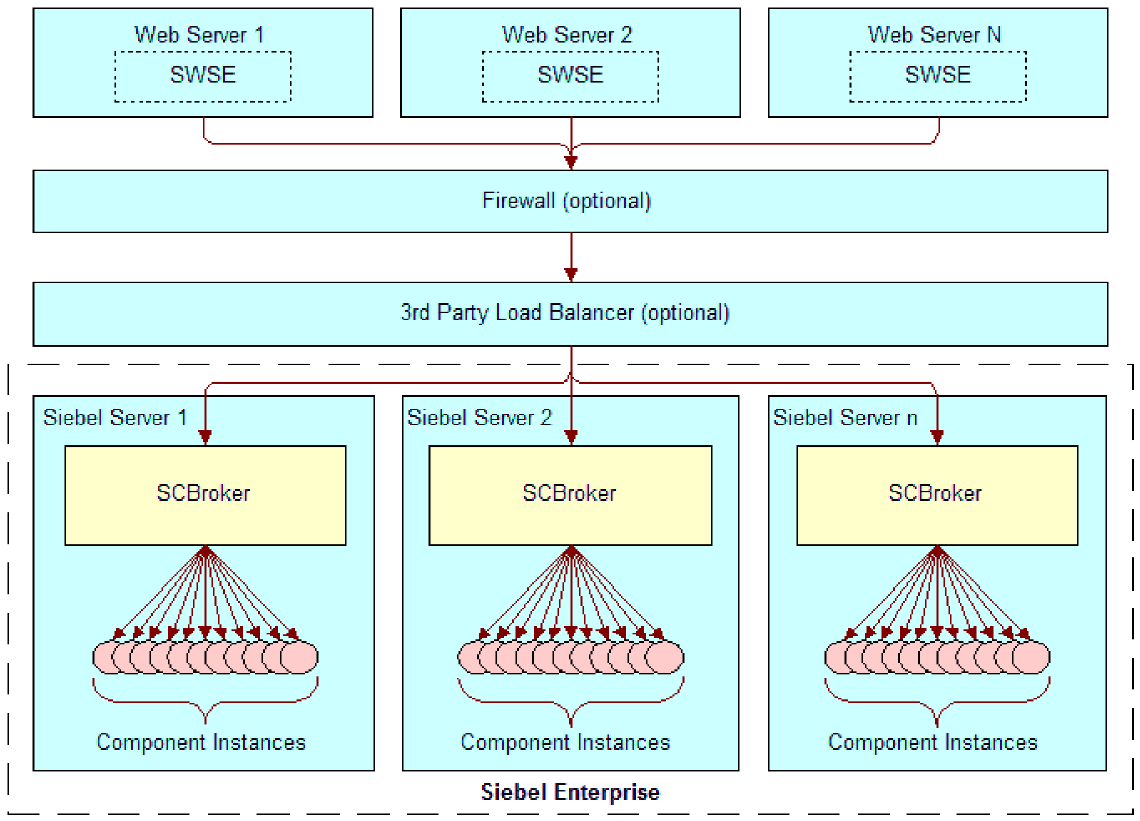


Figure 2. Example of Third-Party Load Balancing

Siebel Internet Session Network API (SISNAPI) is a Siebel proprietary message-body format running on top of TCP/IP. SISNAPI communicates 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.

User Request Types

The Siebel Web Server Extension (SWSE) generates three types of user requests. Each request causes a new connection to a Siebel Server through the load balancer: initial request, retry request, and reconnect request. The Siebel load balancing module in the SWSE recognizes these requests 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 in message 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 the Siebel Connection Broker

The Siebel Connection Broker (SCBroker) is a server component that provides intraserver load balancing. SCBroker distributes server requests across multiple instances of Application Object Managers (AOMs) 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 with the least number of running tasks. This creates a user session. Thereafter, SCBroker forwards requests that apply to this session directly to the AOM 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 allows it to restart automatically. Setting this parameter to Off/False is not recommended.
- **ConnForwardTimeout.** The connection forward time-out determines how long SCBroker will wait for an Object Manager to accept a request. The default is 500 milliseconds. This time-out minimizes wait time when SCBroker forwards a connection request to an Application Object Manager, 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 an RR to the Siebel URL. The SWSE then retries the request by forwarding it to the third-party HTTP load balancer. This causes the load balancer to use the round-robin routing rule to forward the request to the next available Siebel Server.

- **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 ConnForwardTimeout.

About Server Request Broker

Server Request Broker (SRBroker) processes synchronous server requests—requests that must be run immediately, and for which the calling process waits for completion. SRBroker in Siebel 7 replaces Server Request Manager (SRMSynch) in previous versions.

Where SRMSynch could only run server requests on a local Siebel Server (that is, the same Siebel Server that SRMSynch was running on), SRBroker can run server requests on any server in the 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. Server Request Broker 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 (a round-robin algorithm).
- If the required component is not available anywhere in the Enterprise, then the server request will fail.

This provides *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 59.

About Server Request Processor

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

The Server Request Processor 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:

- Be the correct state (Submitted)
- Its start time must have passed
- The target Siebel Server must either not be specified, or must be the Siebel Server on which the SRProc instance 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.

This provides resilient processing for server-initiated tasks. As long as a SRProc task is running somewhere in the Siebel Enterprise Server, the request will be processed. For more information on resilient processing, see "About Resilient Processing" on page 59.

About Siebel Enterprise Application Integration (EAI)

Siebel EAI provides components for integrating Siebel Enterprise 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. It also includes tools for cross application integration through Universal Application Network (UAN).

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, BizTalk, and OLE DB; Sun Microsystems Java and J2EE; XML, and HTTP, and many other standards.

These 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 on EAI, see *Overview: Siebel Enterprise Application Integration Volume I*.

About Siebel Enterprise Integration Manager (EIM)

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

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

For more information on using Siebel EIM, see *Siebel Enterprise Integration Manager Administration Guide*.

About Siebel Tools

Siebel Tools is a Windows-based, object-oriented development environment for creating and customizing Siebel Business Applications. 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 with different purposes:

- Physical User Interface (UI) Layer: Templates and tags that render the UI.
- 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. This means, for example, that you can make changes to a Siebel application without changing the underlying database structure. Similarly, you can extend the Siebel database schema without affecting the Siebel application.

For additional information on Siebel Tools, see *Configuring Siebel Business Applications*.

Example of User Request Flow in a Siebel Deployment

Figure 3 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.

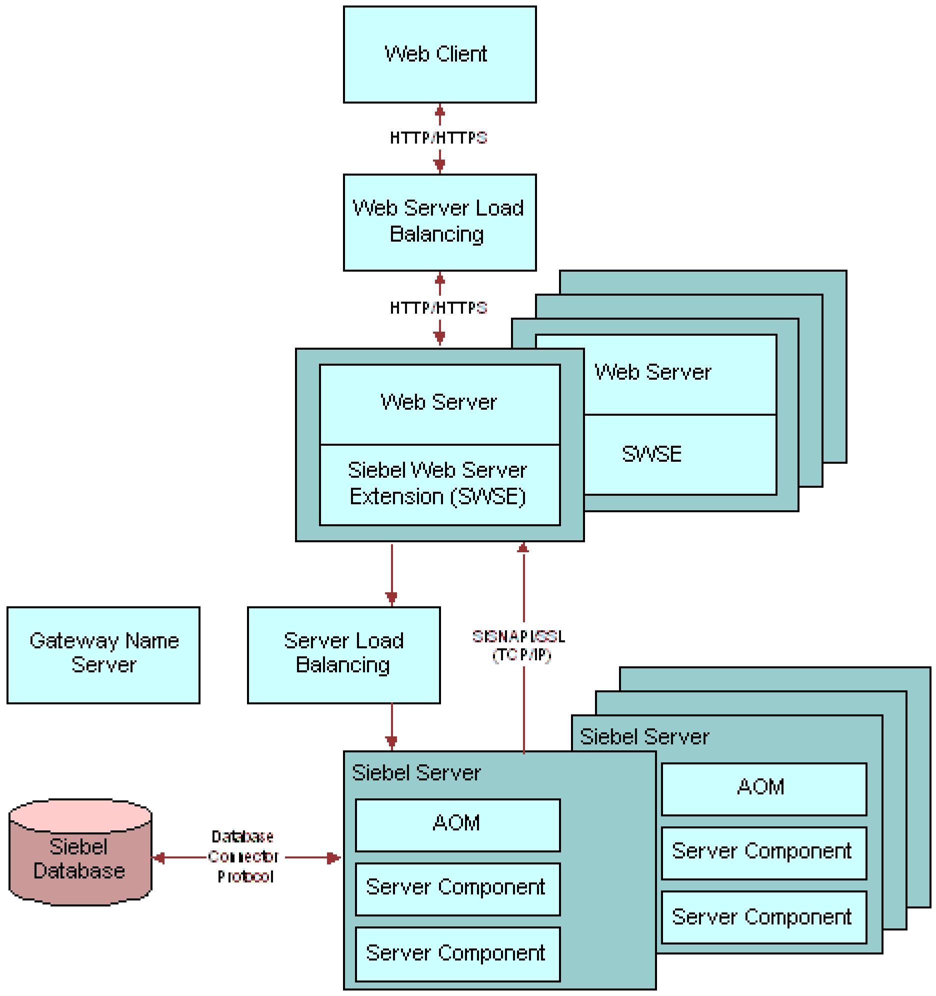


Figure 3. Generic User Request Flow in Siebel Business Applications

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.
   - If using Siebel load balancing, SWSE forwards the request to a Siebel Server in round-robin fashion.
   - If 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.

     SISNAPI (Siebel Internet Session application programming interface) is a messaging format that runs on top of the TCP/IP protocol. It is used for network communication between Application Object Managers (AOMs) and SWSE.
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 goes through the database connection, using 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 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.

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

This process shows you how to determine the Siebel infrastructure requirements for a production environment. Along with a production environment, you should also plan for a software development and a test environment.

Use the following steps to plan your Siebel deployment infrastructure:

1. "Determining How the System Will Be Used" on page 28
2. "Defining Data Flows and Integration Requirements" on page 29
3. "Determining Database Requirements" on page 30
4. "Mapping Business Requirements to Siebel Server Components" on page 31
5. "Defining High-Availability Policies" on page 33
6. "Mapping Siebel Deployment Elements to Platforms" on page 35
7. "Determining Network Requirements" on page 39
8. "Defining a Test and Transition Plan for the Siebel Deployment" on page 40

Determining How the System Will Be Used

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.

This topic is a step in "Process of Infrastructure Planning" on page 27.

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 user type will perform using the system. Start with each key business process and map its steps to tasks. This allows you to verify that 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. Table 3 shows an example of how you would list transaction volumes.

Table 3. Denver Transaction Volumes

User Type	Number	Task	Avg. Vol./ Day	Max. Vol./ Day
Call Center Agents	25	1. Inbound customer order	300	500

Defining Data Flows and Integration Requirements

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.

This topic is a step in "Process of Infrastructure Planning" on page 27.

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 shows an example of a model of a data flow. The example shows a call center running Siebel Communications. 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 adaptor. Siebel Communications also queries the Phone Number Management System for available phone numbers in real time. The Phone Number Database then receives assigned phone numbers from the Siebel database using Siebel EIM.

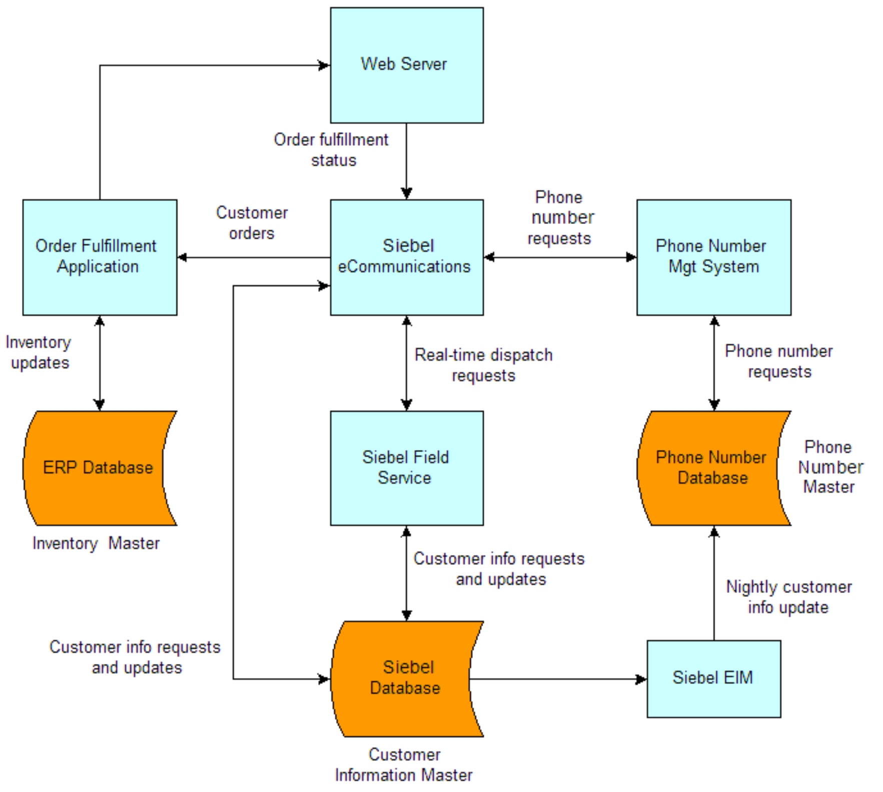


Figure 4. Example of a Data Flow Model

Determining Database Requirements

This 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. This allows you to estimate database size requirements and expected growth.

This topic is a step in "Process of Infrastructure Planning" on page 27.

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. Contact Siebel Expert Services for information on mapping records to Siebel tables.
- How much space will database indexes occupy? Typically, indexes require as much space as the database. For example, a 50GB database will require an additional 50GB 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 tablespace, log files, and space required for loading data.

Mapping Business Requirements to Siebel Server Components

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

This topic is a step in "Process of Infrastructure Planning" on page 27.

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.

A common problem when Siebel Expert Services does an implementation readiness review at customer sites is that second-tier component requirements are not correctly identified. To avoid this, work closely with the application development 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

Siebel Systems supports two load balancing methods—Siebel load balancing and third-party HTTP load balancers. For a description of these two methods see "About Load Balancing" on page 17.

Siebel load balancing and third-party HTTP load balancers provide similar features. Table 4 compares key characteristics of Siebel load balancing with third-party HTTP load balancers. In the table, SISNAPI is the Siebel protocol used to communicate with Siebel Servers.

Table 4. Load Balancing Method Comparison

Feature Area	Siebel 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 per 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 application server.	Generally supports NAT, VIPs, VPorts. Also supports packet inspection and filtering.

Table 4. Load Balancing Method Comparison

Feature Area	Siebel Load Balancing	Third-Party HTTP Load Balancer
Administration and configuration	Configured using text file. Administered through Siebel Server Administration.	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.
- They provide desired security features.
- A more sophisticated load-balancing scheme is desired.
- The site requires centralized monitoring and management of system hardware and network infrastructure.

Siebel 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 Siebel Field Service. Siebel Call Center must compete for resources with Siebel Field Services on one of the servers. This 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 *Performance Tuning Guide*.

Defining High-Availability Policies

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

This topic is a step in "Process of Infrastructure Planning" on page 27.

Siebel Servers

For each business location, assess the impact of losing each server component. Think about the component failing rather than the hosting platform. A common problem when Siebel Expert Services does implementation readiness reviews at customer sites is that failure of individual server components has not been adequately analyzed. The result is that server components important to normal application function are not recognized as 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 for each 20 to 40 users. The size of rollback extents should be 100K/100K. If you are using Siebel EIM, 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 Server 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 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 30.
- Required Siebel Server components. See "Mapping Business Requirements to Siebel Server Components" on page 31.
- High-availability policies. See "Defining High-Availability Policies" on page 33.

This topic is a step in "Process of Infrastructure Planning" on page 27.

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. This 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. *Customers should contact Siebel Expert Services for information on how to size Siebel Servers.* Additional information on calculating the settings for MaxTasks, MaxMTServers and MinMTServers is discussed in detail in the *Performance Tuning Guide.*

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. 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 the following scenario. An implementation plan involves having 1,000 connected users with light Assignment Manager usage, light Workflow usage, heavy off-peak EIM batch jobs, and a moderate Communication Server load. The implementation also requires some degree of high-availability and resiliency. One possible architecture solution would be as follows:
     - Clustered Gateway. 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.
   - Engage Siebel 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.
   - 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 the product model design and how the product model 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. 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.
   - For larger implementations, consider running Siebel Reports Server (Actuate) on dedicated hardware. Generating reports can cause peaks in usage distribution and cause CPU spikes. If the Siebel Reports Server is isolated from other servers, CPU spikes will not affect the end-user experience.

- 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 great deal of Siebel Remote activity. Once again, it is important to identify those processes with the potential for spikes of resource consumption and 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 Siebel terms, active-active clustering means 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, 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 on a second group of machines, and remote user synchronization 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. This simplifies server administration.

Topology Planning Guidelines

Use 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. This includes multiple Application Object Manager types, each with their own SRF.

Table 5 lists recommended deployment schemes.

Table 5. Deployment Schemes

Deployment Scheme	Recommended?
A single Siebel Gateway Name Server and multiple Siebel Enterprise Servers running on a single machine or UNIX hardware partition. Each Siebel Enterprise Server has its tableowner and database server.	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 UNIX hardware partition or on a single unpartitioned machine.	No. Very difficult to set up. It requires manual workarounds, and requires manually editing registry on Windows platforms.
Multiple Siebel Enterprise Servers sharing a DBMS tableowner.	No. Each Siebel Enterprise Server is required to have its own set of tables.
Multiple Siebel Enterprise Servers, each with their own DBMS tableowner and sharing the same instance of the DBMS executable.	No. If the one Database Server goes down, all of the Siebel Enterprise Servers will go down. Too much dependency on one machine.
A Siebel Enterprise Server hosting multiple Siebel Servers within a single hardware partition.	No. There is no additional scalability, throughput, or performance benefit to this configuration. 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 is the most common way to deploy Siebel version 6.x and later.

Table 5. Deployment Schemes

Deployment Scheme	Recommended?
A single Siebel Server managing multiple applications. Each Application Object Manager type having its own copy of the SRF.	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 SRFs 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 database all on different operating systems.	Yes. This is supported for Siebel 7.x. 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 *System Requirements and Supported Platforms* on Siebel SupportWeb.

Determining Network Requirements

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

This topic is a step in "Process of Infrastructure Planning" on page 27.

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 diagram created in the previous topic ("Mapping Siebel Deployment Elements to Platforms" on page 35).
3. Define firewall requirements. If you are creating a network 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, you plan to use HTTP/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 load balancing is deployed, then the load balancing configuration file must reference this port number. See "Choosing a Load Balancing Method" on page 32.

   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

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.

This topic is a step in "Process of Infrastructure Planning" on page 27.

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

1 **Separate production environment.** Keep the development and test environments physically separate from the production environment. Never conduct development and test activities on the production Siebel Database and preferably not on the production database server.

2 **Server stress testing.** Test Siebel Enterprise Server performance under average and peak workloads. Siebel Expert Services finds that performance problems at customer sites are frequently caused by the following things:

- 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. Undetected single points of failure within the Siebel deployment is a common problem found during implementation readiness reviews by Siebel Expert Services.

  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 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.
  - **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. Services include both hardware platforms and software applications.

Web Clients

Client PC hardware failure and browser crashes are the most common causes of Web Client failure. Operating System crashes can also cause this, but are rare. When the Web Client fails, user sessions are lost even though the sessions usually continue running on the Siebel Server.

This is 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 fail over 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, users 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. This 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.

  If AutoStart is enabled, then the Siebel Server process tries to restart the terminated AOM process. If successful, the new AOM process can host new user sessions.

- **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 as follows:
- **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 Configurator Object Manager, and communication between AOMs and the Reports Server. Some of these components, such as the Reports Server and 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 a number of 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. 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 are 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.

Impact of Service Failures

Table 6 summarizes the impact of failure of services in the Siebel deployment. The table includes information on specific services not already covered.

Table 6. How Service Failures Affect the Siebel Deployment

Service Failed	Affected Component	Impact
Gateway Name Server	Siebel Server components and Siebel Configurator Object Manager	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 Reports Server and report functionality	If connection information has been cached, the Reports Server can still be called. By default, connection information is cached when the connection is made.
	Siebel Configurator Object Manager	You can still launch product configurator sessions, as long as the connection information has been cached. By default, the connection information is cached when the first connection is made.
	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 Servers.
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).

Table 6. How Service Failures Affect the Siebel Deployment

Service Failed	Affected Component	Impact
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 Adaptor	Unavailable.
Siebel database	Client access, background tasks, batch tasks	Unable to access Siebel Business Applications. The Siebel Enterprise Server 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

Siebel Systems conducted a benchmark of potential failure scenarios when running a Siebel deployment. Table 7 summarizes these 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.

Test Environment

The test environment included multiple, load balanced Web servers. Web server load balancing was 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 and the request distribution mechanism was provided by Service Request Broker (SRB) and Service Request Processor (SRP). Product Configurator Object Manager servers were used and load balanced by the Configurator-provided load balancing scheme. A clustered pair of database servers was used. A clustered Siebel Gateway Name Server was also deployed.

Table 7. How Service Failures Affect the Siebel Deployment

Service Failed	Affected Component	Impact
Gateway Name Server	Siebel Server components and Siebel Configurator Object Manager	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 Reports Server and report functionality	If connection information has been cached, the Reports Server can still be called. By default, connection information is cached when the connection is made.
	Siebel Configurator Object Manager	You can still launch product configurator sessions, as long as the connection information has been cached. By default, the connection information is cached when the first connection is made.
	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 Servers.

Table 7. 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).
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 Adaptor	Unavailable.
Siebel database	Client access, background tasks, batch tasks	Unable to access Siebel Business Applications. The Siebel Enterprise Server 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

Siebel Systems conducted a benchmark of potential failure scenarios when running a Siebel deployment. Table 8 summarizes these 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.

Test Environment

The test environment included multiple, load balanced Web servers. Web server load balancing was 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 and the request distribution mechanism was provided by Service Request Broker (SRB) and Service Request Processor (SRP). Product Configurator Object Manager servers were used and load balanced by the Configurator-provided load balancing scheme. A clustered pair of database servers was used. A clustered Siebel Gateway Name Server was also deployed.

Table 8. Specific Failures and Associated Impact

Component Tested	Failure Scenario	Observed Behavior
Siebel Database Server	Observe system behavior while driving server CPU load to 100%.	■ Significant response time impact. ■ No failures were observed.
Siebel 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.
Siebel 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 observed when attempting to log in additional users.

Table 8. 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 observed when attempting to log in additional users.
Siebel 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 crash 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 crash 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 WF, users get a “no server connect string” error message. ■ Failures were observed for the above step.

Table 8. Specific Failures and Associated Impact

Component Tested	Failure Scenario	Observed Behavior
WFProcmgr (Workflow)	Simulate a process crash 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.
Siebel Object Manager	Consume all available tasks on an Object Manager and observe the result.	■ Object Managers 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.
Siebel Object Manager	Simulate resource leaks while server recycling is enabled, and verify how process recycling works under load.	■ New Object Manager gets created when MemoryLimit is hit. ■ Old Object Manager 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 hit, then the whole component restarts. All traffic went to the other server.

Table 8. Specific Failures and Associated Impact

Component Tested	Failure Scenario	Observed Behavior
Siebel 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 (as documented in Bookshelf). ■ User that visits the URL hangs, even after browser scripts get updated.
Siebel Object Manager	Simulate a hanging thread or process.	■ Can still log in to the app server with the spinning process. ■ After simulating a hanging thread, 100 extra users were added. After that, killing the spinning process causes about 40 running users to fail (the number of users on that OM). This implies: ■ Object Managers with hanging threads still receive new connections. ■ You cannot safely kill a spinning process unless you offline/ shutdown the whole comp group or 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, users 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.

Within a Siebel deployment, a service is defined as one of the following:

- Siebel Gateway Name Server
- Siebel Server
- Siebel Database Server
- 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 book 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. When you distribute workload across AOMs, this indirectly distributes workload across the server components that AOMs call. This is called *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 59.

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 (Siebel Gateway Name Server or Siebel Remote) or because of hardware constraints.

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

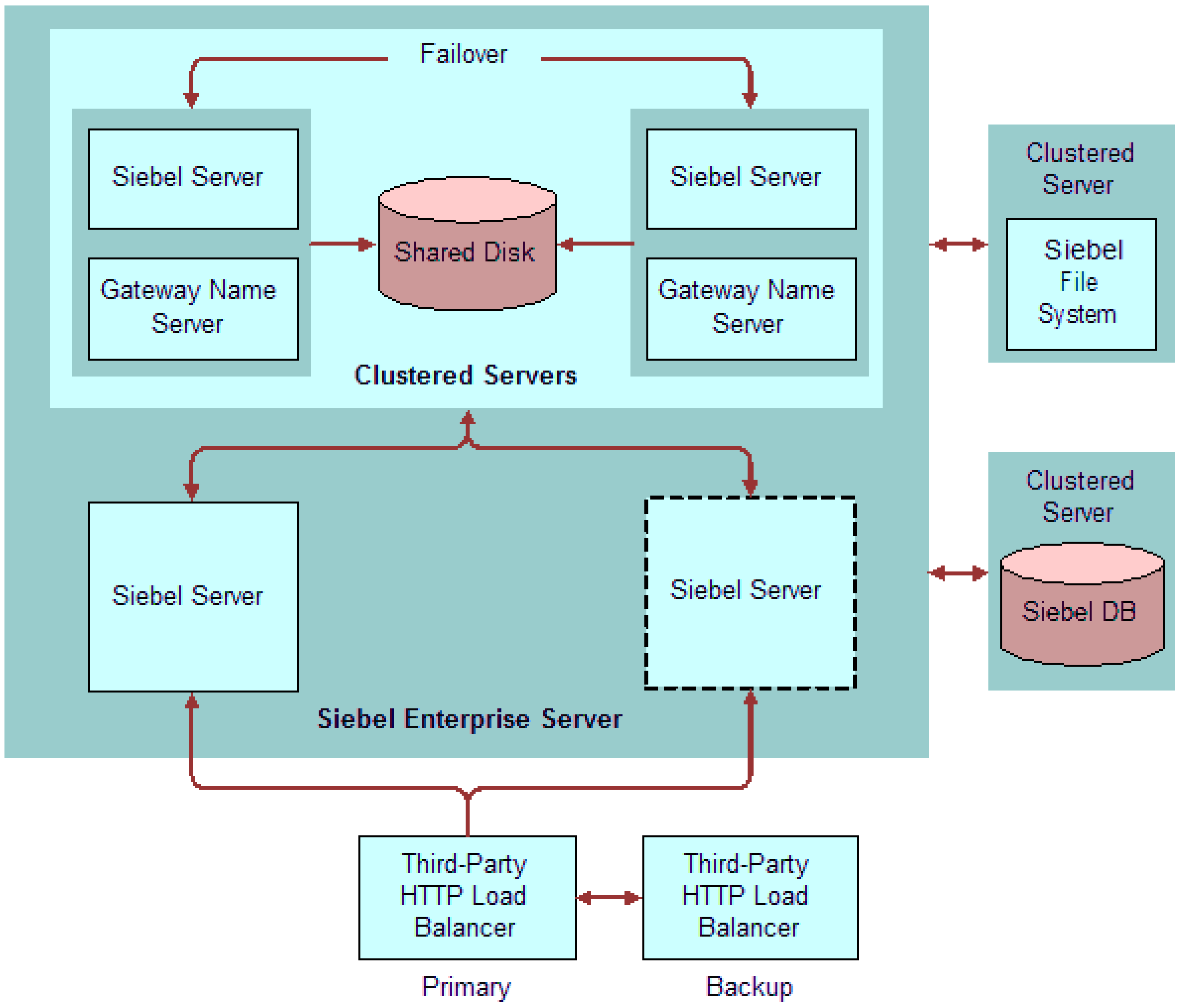


Figure 5. Example of a High Availability Deployment

Recommended High Availability Techniques for Specific Services

The three high availability techniques that Siebel Systems supports are server clustering, load balancing, and resilient processing. Table 9 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 and is the one 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.
- **Blank.** The high availability techniques in the table are not available for this component.

Table 9. High-Availability Support Matrix for Siebel Components

Component	Clustering	Load Balancing	Resilient Processing
Gateway Name Server database (siebns.dat)	Preferred		
Application Object Managers	Supported	Preferred	
Communications Manager	Supported		Preferred
Dynamic Assignment	Preferred		
Siebel Configurator	Supported	Preferred. Uses its own load balancing method	
Document Server	Supported[1]		Preferred
Pricer	Supported		Preferred
EAI (adapters and connectors)	Supported	Preferred, whenever possible[2]	Supported
EAI Object Manager	Supported	Preferred	
Field Service	Supported		Preferred
File System Manager	Supported		Preferred
Interactive Assignment	Supported		Preferred
MQ Series Receiver	Preferred		
Replication Agent	Preferred		
SAP BAPI Integration	Preferred		
SAP IDOC Receiver	Preferred		
SAP IDOC Receiver for MQ	Preferred		
Server Request Broker	Supported		Preferred
Server Request Processor	Supported		Preferred
Siebel File System	Supported		
Siebel Marketing	Supported		Preferred
Siebel Remote	Preferred		
Workflow Monitor	Preferred		
Workflow Process Manager	Supported		Preferred

1. Supported as long as Microsoft Office has been installed on all clustered nodes. This is 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 information about the best approach for your deployment, consult Siebel 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 per 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 Siebel Systems. Set up the Web server load balancer to allow user requests to fail over 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.** Servers hosting an Application Object Manager (AOM) should be load-balanced. 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 Configurator OM is an exception. 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. This provides 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 the 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.** You can use either a third-party HTTP load balancer or Siebel load balancing. Third-party HTTP load balancers typically offer more management capabilities, while Siebel load balancing is less complex to set up and maintain. Servers hosting an Application Object Manager (AOM) should be load-balanced. 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.
- **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 Siebel Systems. At a minimum, use a RAID 5 disk array for your file system. In addition, make regular backups of your data.
- **Siebel database.** Deploy a clustering solution supported by your RDBMS vendor. To guarantee data availability and integrity, use data replication techniques such as mirroring and the disk arrays to keep the backup instance of the database in sync with the primary instance.

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 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 Server, 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.

If you use server clustering, resilient processing is not required, as the cluster provides a resilient environment. Resilient processing is not available for all server components. You cannot use it for server components that must run on only one server (for example Siebel Remote). In these cases, the only high availability option is server clustering.

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

- Communications Manager
- CORBA Object Manager
- Document Server
- Pricer
- Field Service
- File System Manager
- Interactive Assignment
- Server Request Broker
- Server Request Processor
- Siebel Marketing

- Workflow Process Manager

Resilient processing uses two server components:

- **Server Request Processor.** This component handles asynchronous server requests. See "About Server Request Processor" on page 22.
- **Server Request Broker.** This component handles synchronous server requests. See "About Server Request Broker" on page 22.

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. Siebel Systems has certified a variety of third-party vendors to provide server clustering for the Siebel deployment. For a list of vendors and requirements, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

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.

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 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 an instance of the Siebel Database Server and Siebel Server to fail over to each other is supported, but not recommended.

Potential Port Conflicts

Some Siebel Server components, such as Siebel Connection Broker (SCBroker), Remote Synch Manager, Handheld Synch, and Siebel Gateway Name Server 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 a number of 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. This takes 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

The Siebel application supports 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 Siebel Application Object Managers do not support or require clustering.
- Siebel File System
- Siebel Database Server. 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:

- Dynamic Assignment
- Email Agent
- MQ Series Receiver
- Replication agent
- SAP BAPI integration
- SAP IDOC Receiver

- SAP IDOC Receiver for MQ
- Siebel Remote. Make sure that the DockString parameter in the remote client configuration file is referencing the virtual server name. This must be configured correctly for remote synchronization to work after failover.
- Workflow Monitor.

Server Clustering for Some Components Is Not Supported

Siebel Systems does not support server clustering for the following Siebel Server components:

- Siebel CORBA Object Manager
- LDAP/ADSI Directory Server. Vendor may provide built-in replication.
- Universal Queuing Server
- Informatica
- Documents Server (Microsoft server-side integration)
- Hummingbird Search Server
- First Logic Data Quality
- CTI hardware/switch
- Chartworks

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, AOM1, AOM2, and AOM3 running on a Siebel Server. 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 Server a high priority because it is a single point of failure. When clustering the Siebel Database Server, have it already installed and running.
- The Siebel Database Server 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 installer allows you to install all servers at once 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 the appendices of the *Siebel Installation Guide* for the operating system you are using), fill out the section related to server clustering, so that you can refer to this during installation.

About Third-Party Server Clustering Products

Siebel Systems supports several third-party server cluster products. These products are listed in the *System Requirements and Supported Platforms* on Siebel SupportWeb. Some vendors provide documentation on their Web sites that describes how to install these products in a Siebel deployment. This documentation is listed in Table 10.

Table 10. Vendor Documentation for Installing Cluster Products

Vendor	Product	URL
Hewlett Packard	MC/ServiceGuard Cluster	Contact your Hewlett Packard representative.
IBM	HACMP/E	Contact your IBM representative.
Microsoft (Doc is not specific to Siebel applications)	■ Windows 2000 Server ■ Windows 2003 Server	■ http://www.microsoft.com/windows2000/en/datacenter/help/ (For Windows 2000 Server) ■ http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windowsserver2003/proddocs/datacenter/default.asp (For Windows 2003 Server)
Sun	Sun Cluster Data Service for Siebel	http://docs.sun.com/db/doc/817-1540?q=Siebel
Veritas	Volume Manager	Contact your Veritas representative.

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 *System Requirements and Supported Platforms* on Siebel SupportWeb.

In your planning, you need to allocate space for system storage, rollback or temporary storage space, log files, and other 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.

One Database Platform Per Enterprise Server

The Siebel Servers in a Siebel Enterprise Server can connect to only one database. For example, you cannot configure both an Oracle and DB2 UDB 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 per user (for example, 100 accounts per sales representative) and calculate an estimated total number of records per entity for your total user base.
4. Using standard sizing procedures for your specific database, calculate the average record size per 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. This determines 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.
6. Factor a margin of error into your total size calculation.

Database Table Planning

In most implementations, the Siebel tables listed in Table 11 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 tablespace and, if possible, on different physical devices from the tables on which they are created.

Siebel tablespaces on DB2 UDB should be database-managed tablespaces (DMS) rather than system-managed tablespaces (SMS).

Table 11. 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.

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.

Oracle Recovery Planning

Many companies today use RAID storage systems that make Oracle 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 crash recovery.

Also, when redo logs are mirrored at the RAID storage system level (usually RAID1 or RAID0+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 (especially beneficial for read operation).

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—for example, the current System Change Number (SCN) or transaction tables—ripple across all members of the control file specification.

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 tablespaces 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 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. Ideally, all DB2 devices should reside on different disk spindles to minimize I/O contention. When this approach is not possible, spread devices that contain database objects that are often used together across different spindles. These objects include tables, their indexes, and commonly joined tables.

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 redundant array of independent disks (RAID) 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 the indexes whose names end with _P1 from the tables on which they are created. These tables are heavily used in join operations.

If you will make frequent use of Siebel Enterprise Integration Manager (EIM), you may want to put the interface 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 12 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 12. 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.
3–5 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 65 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 a discussion of this, see "Database Physical Device Planning" on page 68.

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 interacting with the application, browsing through records, and so on. This 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. Such features include:
  - 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. For a list of supported browser versions, see *System Requirements and Supported Platforms* on Siebel SupportWeb.

Deploying Siebel applications in high interactivity mode requires adhering to strict guidelines regarding the deployed operating system, Internet Explorer version and settings, and Java software environment. For a discussion of which Siebel applications must be deployed in high interactivity mode and the related browser requirements, see *Siebel System Requirements and Supported Platforms*.

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 71.

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. This 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. This 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 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 71.

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 a browser for all users. Be sure to use the same browser type for development, testing, and production environments.

For a list of browsers that support standard interactivity, see *Siebel System Requirements and Supported Platforms*.

8 Application-Level Deployment Planning

Some Siebel applications can be deployed in more than one way. 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 additional information on application deployment planning, see the *Performance Tuning Guide* and the *Siebel System Administration Guide*.

This chapter includes the following topics:

Session Communications Server Components

Session communications refers to using 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) and inbound and outbound communications (such as email).

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.
- **Communications Inbound Receiver (CommInboundRcvr).** This server component receives and queues inbound work items, and queues them for processing by Communications Inbound Processor. Work items may include email messages (for Siebel Email Response), voice work items that are to be routed using Siebel Universal Queuing (for Siebel CTI), or inbound wireless messages for Siebel Wireless Messaging.
  - For nonreal-time work items, such as email messages for most deployments of Siebel Email Response, Communications Inbound Receiver queues work items it has received for further processing by Communications Inbound Processor.
  - For real-time work items, such as phone calls for Siebel CTI or email messages for some deployments of Siebel Email Response, Communications Inbound Receiver processes work items it has received. Communications Inbound Processor is not used.

- **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 or other types of messages.

Siebel Product Module

In addition to Siebel CTI or Siebel Email Response, you may be using the following Siebel product modules for session communications:

- **Siebel CTI Connect.** This module consists of CTI middleware, a communications driver, and sample communications configuration data. Siebel CTI Connect is based on third-party CTI middleware—Intel NetMerge, formerly Dialogic CT Connect. For Siebel CTI Connect, consult Intel documentation provided on the Siebel Business Third-Party Bookshelf.
- **Siebel Universal Queuing.** This module routes communications work items to agents.
- **Siebel Smart Answer.** This module analyzes the content of email and search requests and returns an automatic response or suggests one or more responses to the user for approval.

Siebel Smart Answer is based on third-party products from Banter. See *Siebel Smart Answer Administration Guide* and consult Banter documentation provided on the Siebel Business Third-Party Bookshelf.

Session Communications Performance Factors

Depending on your deployment, your agents may be handling phone calls (Siebel CTI), email messages (Siebel Email Response), work items of other communications channels, or a 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 Server and Siebel Server. Think time is an important factor in overall system load. Estimation of think time 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 per agent, and how these factors relate to your organization's service goals are also important factors influencing performance. Some agents receive a large number of work items from ACD queues or Siebel Universal Queuing, 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 are using 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 *System Requirements and Supported Platforms* on Siebel SupportWeb.)

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 and queues inbound work items, and queues them for processing by Communications Inbound Processor. Work items may include email messages (for Siebel Email Response), voice work items that are to be routed using Siebel Universal Queuing (for Siebel CTI), or inbound wireless messages for Siebel Wireless Messaging.
  - For nonreal-time work items, such as email messages for most deployments of Siebel Email Response, Communications Inbound Receiver queues work items it has received for further processing by Communications Inbound Processor.
  - For real-time work items, such as phone calls for Siebel CTI or email messages for some deployments of Siebel Email Response, Communications Inbound Receiver processes work items it has received. Communications Inbound Processor is not used.
- **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 Components or Modules

In addition to Siebel Email Response, you may be using the following Siebel components or modules:

- **Siebel Smart Answer.** This module analyzes the content of email and search requests and returns an automatic response or suggests one or more responses to the user for approval.

  Siebel Smart Answer is based on third-party products from Banter. See *Siebel Smart Answer Administration Guide* and consult Banter documentation provided on Siebel Business Third-Party Bookshelf.

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

- **Siebel Universal Queuing and session communications components.** If you are using Siebel Universal Queuing to route email work items, then additional session communications components apply. The communications toolbar is enabled in the Siebel application to support accepting new work items.

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 *System Requirements and Supported Platforms* on Siebel SupportWeb.

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.

  If you are deploying Siebel Smart Answer, you must also consider the size of the knowledge base.

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 topic focuses on inbound and outbound email processing. In a multichannel environment, or when Siebel Universal Queuing is deployed, session communications performance issues also apply. Using Siebel Smart Answer, especially for auto-response capabilities, reduces the number of agents needed to handle incoming email and reduces corresponding demand on session-related computing resources, such as AOM or CommSessionMgr.

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.

CommOutboundMgr and Siebel Smart Answer (Smart Answer Manager) may be run together on a different machine (or machines), as appropriate.

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. Business rules are defined using constraint statements contained in models stored on the Siebel File System.

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

- **Application Object Manager (AOM).** The Siebel Configurator solution engine functions within an AOM, such as Call Center Object Manager (SCCObjMgr) for Siebel Call Center.
- **Siebel Product Configurator Object Manager (eProdCfgObjMgr).** This is an AOM containing the Siebel Configurator solution engine. It can be deployed on a separate Siebel Server from where Siebel Configurator sessions are invoked.
- **Siebel File System.** This component stores cached object definitions for customizable product models in the CFGCache directory on the Siebel File System.

Configurator’s architecture has been greatly enhanced in Siebel 7. With Siebel 7 a completely modular, service-based architecture has been put in place with different services interacting with each other to result in a Configurator user session.

At the same time, recognizing that configuration may be computationally expensive in some cases, some basic infrastructural enhancements allow flexibility of different deployment options to take care of different business needs. This document discusses the different deployment options along with the considerations for choosing an option in a later section.

Before discussing the actual parameters and where they can be set, it is worthwhile to go through a brief overview of Configurator's architecture and to get an understanding of the various services that are run to result in a multi-user Configurator deployment.

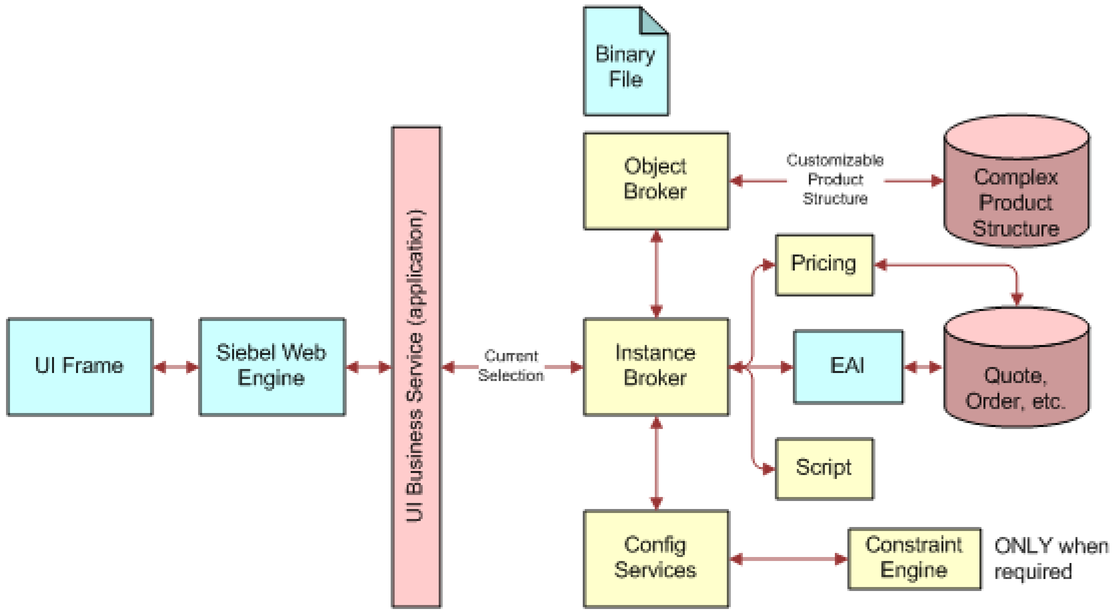


Figure 6. Detailed Configurator Architecture

Figure 6 shows detailed configurator architecture and the interaction of various services with each other during run time. What follows is a description of the important services depicted in this diagram:

- **UI Business Service.** The UI business service is a service that the configurator uses to render the UI 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 service is the way the user interacts with the configurator. A unique instance of the UI 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.
- **Object Broker.** The Object Broker is a 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 creates a translation of the customizable product definition that is retrieved by the Object broker into a format the worker (defined below) can understand.
- **Constraint Engine.** The constraint engine is also called the worker.
- **Worker.** The worker is a service that enforces all the rules associated with the configuration and validates all user selections, as they are made to ensure a valid configuration.

For more information about elements of Siebel Configurator’s internal architecture, including Instance Broker (Complex Object Instance Service business service) and Object Broker (Cfg Object Broker business service), see *Product Administration Guide*.

Siebel Configurator Performance Factors

Siebel Configurator performance has two aspects:

- **Customizable product loading time.** This is the time elapsed from the moment a user clicks Customize in a quote or order until the user interface for the customizable product has been loaded and displayed to the user.
- **Selection response time.** This is 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 key performance factors that influence these times are as follows:

- **Use of SnapShot Mode caching.** This feature caches customizable product models in memory, which significantly reduces the amount of time required to load customizable products for each new user. This feature 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 *Product Administration Guide*.
- **Number of concurrent configuration users.** This is the number of concurrent users who access customizable product models. This figure will be some percentage of the total number of concurrent users on the AOM.

  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 product models.** This represents the total size and complexity of each customizable product model, 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 product models.** This is the number of customizable product models accessed by users. It is assumed that each user accesses no more than one customizable product model at one time. A given group of concurrent users may access multiple models, however, each of which must be separately cached.

Siebel Configurator Performance Drivers

Based on an understanding of Configurator's architecture, the following section examines some performance drivers for a Configurator deployment. Run-time performance of the Configurator can be divided into two areas. They are:

- **Response time of loading configuration.** This is the time elapsed from the instance user clicks Customize in a quote or order, until the Configurator UI is loaded and displayed to the user. This will depend upon a number of factors like the model size and complexity, the time taken to extract the model definition from the database, and instantiating all the services that are required to create the session.
- **Response time for each selection made by the user.** This is the time elapsed from the instance a user makes a selection until the Configurator returns with a response that may be an update to the configuration or a conflict message. This will depend on the model size and complexity.

  This section of the *Deployment Planning Guide* deals with ways to minimize the load time and not with response time for each selection.

  At the highest level, the response time of a configuration to load would be best under the following circumstances:

  - The model 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 the load time performance of 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. This is why a caching strategy has to be devised for each deployment.

  To enable administrators to implement the caching strategy that is best suited for their deployment, a number of “switches” in the form of server parameters have been provided.

Siebel Configurator Deployment Planning

There are two major approaches to deploying Siebel Configurator:

- Running Siebel Configurator in an AOM component.
- Running Siebel Configurator on one or more dedicated Siebel Servers. Such servers are sometimes referred to as remote servers, because they are remote to the machine on which AOM is running. In general, this section uses the term dedicated servers.

Running Siebel Configurator in an AOM Component

You can run Siebel Configurator in the AOM component, such as for Siebel Call Center.

If a small number of concurrent users require configuration sessions, or there are a small number of customizable product models, 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 other than the AOM. This component is Siebel Product Configurator Object Manager (eProdCfgObjMgr).

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

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

Configurator Caching

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 from a list in the server parameters later in the process. 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 the memory; you can configure the number of factories that are cached from a list in the server parameters later in the process.

Worker (7.5.3 and Earlier)

The worker is a service that enforces all the rules associated with the configuration and validates all user selections, as they are made to ensure a valid configuration. A unique instance of the worker, specialized for the customizable product the user is configuring, is required for each user. Unlike factories, multiple users cannot share workers at the same time. However, when a user exits configuration, the worker is freed and it can be used by another user for a session. A server setting is available for caching workers and avoiding the performance impact of creating a worker at run time.

Worker (7.7 and Later)

After 7.5.3, the purpose of the worker has not changed, except that it is now a “shared” worker. 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 model.

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 this 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 decides the upper bound of caching.
- Cache is created as one goes along. This means the first user request would result in the first set of cache being created. The second user may end up using the same cache because the user wants to configure the same customizable product that user 1 configured. Meanwhile, the third user, who wants to configure a different product, creates a new cache and so on.

Determining Server Settings for Cache Management in SnapShot Mode (7.0.4 and 7.5)

Table 13 shows server settings for SnapShot mode.

Table 13. SnapShot Mode Cache Parameters

Name	Display Name	Data Type	Defaul t Value	Description
eProdCfgSnapshotFlg	Product Configurator - Collect and Use the snapshots of the Cfg Objects	Boolean	FALSE	Setting to determine if configurator objects would be cached or not.
eProdCfgNumOfCachedObjects	Product Configurator - Number of Objects Cached in Memory	Integer	1000	Setting to determine the number of Objects cached by the Object Broker in memory for each user.

Table 13. SnapShot Mode Cache Parameters

Name	Display Name	Data Type	Default Value	Description
eProdCfgNumbOfCachedFactories	Product Configurator - Number of Factories Cached in Memory	Integer	10	Number of factories cached in memory.
eProdCfgNumbOfCachedWorkers	Product Configurator - Number of Workers Cached in Memory	Integer	50	Number of workers cached in memory.
eProdCfgSnapshotFlg	Product Configurator - Collect and Use the snapshots of the Cfg Objects	Boolean	FALSE	Setting to determine if configurator objects would be cached or not.

Determining Server Settings

After outlining the reasons for caching and listing the parameters that are used to manage the cache, you need to determine the settings. The cache parameters will vary widely depending upon the customizable product structure, rules, size, and so on. However, there are some quick tests that you can do to determine the size of the customizable product. From these tests, you can estimate the size of the factory and worker and determine the cache parameters.

The procedure below describes what you need to do. It is part of an example that explains how to determine server settings. Note that all numbers used for memory in the example that follows are purely hypothetical and are not in any way representative of the actual resource requirements in a production environment.

Test Steps

- Run the Siebel application in a connected client mode. The database may or may not be on the same machine; it does not matter.
- Launch the Siebel application and navigate to Quote/Order (whichever the end user would be using).
- Launch Windows Task Manager and note the memory used by Siebel.exe (for example, if memory is 20 MB, call it X).
- Add the customizable product to the order (assuming this is the end user scenario).
- Launch Configurator by clicking Customize. Note the memory after the model is loaded (for example, if memory is 40 MB, call it Y).
- Add items to the configuration as an end user would.

- Do the same for a couple of conflicts and their resolution. After following the steps that a normal end user would, note the memory (for example, if memory is 50 MB, call it Z).
- When you have added enough items as an end user would, click Done to save and exit configuration.

Determining Factory and Worker Size

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-X)= .75(40-20) = 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-X)-Factory size= (50-20)-5= 25MB.

Object Cache Size

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

Example of Sizing Cache Parameters with SnapShot Mode

Assumptions

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

- The enterprise needs to support 5000 concurrent call center users.
- Of these 5000 call center users, 100 need to be using Configurator concurrently.
- There is only one customizable product in the product portfolio. (Multiple-product cases will be discussed in a later section.)

Sizing

Because all caching and services are specific to the Object Manager 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 not indicative of call center sizing.)

- Assume you are supporting the 5000 call center users on eight application servers (each being a Pentium 4 machine with 4 CPUs and 4GB of memory), with each server handling 625 users.
- Each application server itself is run with 25 Object Managers, with each Object Manager 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 per server or one per OM.
- 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 Object Manager process. This means you must cache 25 workers per server or one per OM.

In the example above, the cache size in this case for each OM equals the size of the factory cache plus the size of the worker cache. Expressed as a formula, it looks like this: 5+25= 30MB per OM. Therefore, the Configurator cache requires a total of 30*25= 750MB across the whole server for each server.

The server parameters would be set as follows for each application server:

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

Observations About Sizing

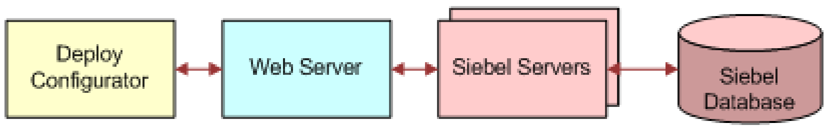
From the sizing exercise above, it is clear that the Configurator cache needs to be actively managed for best performance using appropriate resources. This is why it is extremely important to go through the exercise of sizing the cache. In some cases the cache requirements may be such that they require 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 allocated been 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 object manager process size. The average OM process size depends on the number of OM processes, the total memory available, and the maximum process size for the OS being used. For additional assistance in this area, contact Siebel Expert Services.

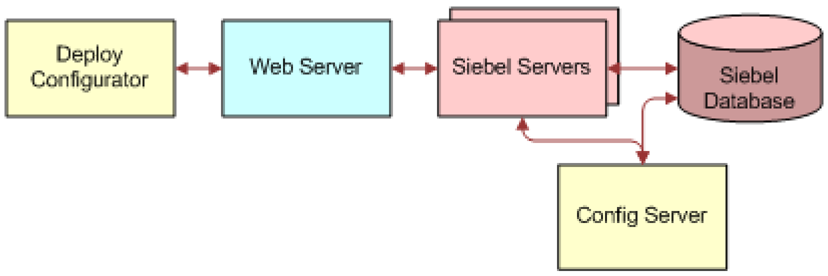
Deployment Options

As mentioned at the beginning of the discussion on architecture, one of the enhancements made to the Configurator is the flexibility of deployment options offered with Siebel 7. These deployment options are as follows:

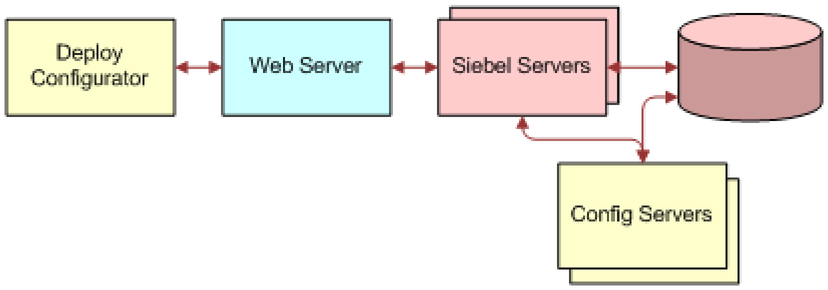
- Deploy Configurator to run on the same machine as the base application server, as shown in the figure below.



- Deploy Configurator to run on a different machine than the base application server. The figure below depicts this deployment option.



- The modes shown above are the supported deployment options for 7.0.3/7.0.4 releases. Release 7.5 supports the following third deployment mode:



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

- The number of OMs 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*8= 200 factories and 200 workers to be cached.
- In memory terms:
  - 200*5= 1000MB for caching factories, and
  - 200*25= 5000MB for caching workers

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

Because the requirement is to support 100 concurrent users, a large number 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 Object Manager process the user who is configuring is connected to. For this reason, caching must be done across all object managers.

There are other problems with this scenario. For instance, what happens when two users are connected to the same OM 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 OM process these users might be connected to, you would have to cache two factories and two workers for each OM. This is a less-than-satisfactory solution in this instance.

Example of Sizing the Deployment with a Dedicated Configurator Server

Consider the example that was sized for the deployment option of running the Configurator on the same servers as the application server, and size it for the deployment option of Configurator running on a separate server.

Assumptions

The requirement is to support 5000 concurrent 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 need to be using the Configurator concurrently.
- There is only one customizable product in the product portfolio. (Multiple-product cases will be discussed in a later section.)

Sizing

Because all caching and services are specific to the Object Manager 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 4 CPUs and 4GB of memory), with each server handling 720 users.
- Each application server itself is run with 25 Object Managers, with each Object Manager 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 Object Managers, with each Object Manager 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. You must cache four factories for the Configurator server or one per OM.
- 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 Object Manager on the Configurator server.

In the example above, the cache size in this case for each OM equals the size of the factory cache plus the size of the worker cache. Expressed as a formula, it looks like this: 5*1+25*25= 630MB per OM. Therefore, the Configurator cache requires a total of 4*630= 2520 MB across the whole Configurator server for each server.

The server parameters would be set as follows for the application and Configurator servers:

■ eProdCfgSnapshotFlg:	True	Set On: Each application server
■ eProdCfgServer:	Name	Set On: Each application server (Server parameter listed below)
■ eProdCfgSnapshotFlg:	True	Set On: Configurator server
■ eProdCfgNumOfCachedObjects:	1000	Set On: Configurator server
■ eProdCfgNumbOfCachedFactories:	1	Set On: Configurator server
■ eProdCfgNumbOfCachedWorkers:	25	Set On: Configurator server

The implications of this across the whole enterprise of eight servers, with one of the servers working dedicatedly to support Configurator, means 2520MB of cache. This is much lower than the 6000MB 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 OM, there would never be a case where a user does not find a cached worker to work with. In a multi-model scenario, this deployment would be much more efficient in terms of memory usage.

Server Settings for Dedicated Configurator Server Deployment Mode (7.5 and later)

Table 14 shows server settings for dedicated Configurator server deployment mode.

Table 14. Siebel 7.5 (and Later) Dedicated Configurator Parameter Settings

Name	Display Name	Data Type	Default Value	Description
eProdCfgRemote	Product Configurator - Use Remote Service	Boolean	False	Setting to determine if configurator is being run on a different server from the app server. Change it to True for running a configurator server.
eProdCfgServer	Product Configurator - Remote Server Name	Text		Remote server name for product configuration service. For multiple configurator servers list them with a ; separating them.

Table 14. Siebel 7.5 (and Later) Dedicated Configurator Parameter Settings

Name	Display Name	Data Type	Default Value	Description
eProdCfgTimeOut	Product Configurator - Time Out of Connection	Integer	20	Setting in Seconds that determines the time for which the app server would try to initiate a connection with the remote configurator server before returning error to user
eProdCfgRemote	Product Configurator - Use Remote Service	Boolean	False	Setting to determine if configurator is being run on a different server from the app server. Change it to True for running a configurator server.
eProdCfgServer	Product Configurator - Remote Server Name	Text		Remote server name for product configuration service. For multiple configurator servers list them with a ; separating them.
eProdCfgRemote	Product Configurator - Use Remote Service	Boolean	False	Setting to determine if configurator is being run on a different server from the app server. Change it to True for running a configurator server.

Table 14. Siebel 7.5 (and Later) Dedicated Configurator Parameter Settings

Name	**Display Name**	**Data Type**	**Default Value**	**Description**
eProdCfgKeepAliveTime	Product Configurator - Keep Alive Time of Idle Session	Integer	900	Setting in seconds to determine the interval of inactivity during a configuration session after which the user session is killed.
eProdCfgMaxNumbOfWorkerReuses *(Note: this parameter is not used in Siebel versions 7.8 and higher.)*	Product Configurator - Number of Reuse for each Worker	Integer	10	Number of times a cached worker is reused before it is destroyed. Workers need to be recycled after a certain number of reuses for the purpose of garbage collection and cleaning. It is recommended that the following server parameter should not be set to a value more than 10 to ensure efficient garbage collection and cleaning

Workflow Deployment Planning

Siebel Business Process Designer is a customizable business application that lets you define, manage, and enforce your business processes. It allows you to design complex workflow processes and automate the enforcement of business policies and procedures. For information on using and administering Siebel Business Process Designer, see *Siebel Business Process Designer Administration Guide*.

The application has the following modules:

- **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.

  The Workflow Process Designer is located 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.

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 sizing a system, look at the maximum number of workflow tasks you expect to have active at a given time. This determines the maximum number of Object Manager sessions Siebel applications create.

Starting with Siebel 7.0, 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.

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, you should run each type in a separate process. This 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 per process are controlled through the parameters MaxMTServers (Maximum MT Servers), MinMTServers (Minimum MT Servers), and MaxTasks (Maximum Tasks).

These parameters are per Siebel Server. For example, MaxMTServers refers to how many multithreaded processes to run on each Siebel Server machine. For details, see *Siebel System Administration Guide*.

Creating a Siebel Reports Server Cluster

NOTE: When using clustering, only one Encyclopedia volume can be used for each Actuate iServer cluster. The clustered environment uses one encyclopedia volume (by default it is the volume owned by the cluster master). All nodes in the cluster point to that volume.

New as of Siebel 7.7 is the ability to cluster the Siebel Reports Server. You can take advantage of Actuate clustering and fail over support to achieve higher levels of report generation performance and high availability. The Reports Server can run in a standalone configuration or be configured to run in a cluster with no single point of failure. Requests are automatically routed to the appropriate machine based on the load and availability to services on each machine.

The Siebel Reports Server is clustered using Actuate's native clustering capability that does not require any additional clustering hardware or software.

Before installing the cluster environment, determine where the actual Encyclopedia files will reside. Nodes with the Encyclopedia, View, or Factory services enabled require direct file access to the Encyclopedia volume data.

For more information about these configurations, read the following sections (and subsections) in the *Administering Actuate iServer System* manual in the Actuate folder on the *Siebel Third-Party Bookshelf* CD-ROM.

- "Understanding Encyclopedia volume configuration"
- "Understanding the effects of disk I/O speed"

Following is general information for creating a Siebel Reports Server cluster. It is strongly recommended that you review Chapter 4, "Performing Actuate iServer System cluster administration tasks" in the *Administering Actuate iServer System* manual in the Actuate folder on the *Siebel Third-Party Bookshelf*.

To create the Siebel Reports Server cluster

1. Using the Siebel Reports Server Installer, install as a standalone the Actuate iServer and Management Console for Siebel component on each machine that you are planning to cluster.

   NOTE: Set the Actuate iServer to start manually and not automatically on PMD start.

   When installing the Actuate components, select the default port numbers. You can choose other port numbers, but make sure to note down those port numbers.

   Determine which node will act as cluster master, which one as backup cluster master, and which node will serve as Volume owner and backup owner.

2. Note the path to your Encyclopedia volume data.

   All cluster nodes require read/write access to this directory.

   Use the following steps for adding all nodes to a cluster.

3. From the Actuate Management Console for each of the iServers that will join the cluster as nodes, log in to System Administration.

4. Select Servers from the left-side panel.

5. Select the General tab, and make a note of the following parameter values for each iServer:

   - Hostname or daemon IP address: = the node's host name (for example, qareport1)
   - Daemon listen port: = the value for port (for example, 8100)
   - Server IP address: = the host name or IP address (for example, qareport1 or 172.20.72.93)
   - Server port: = the port number (for example, 9010)
   - Server port base: enter port number (for example, 9050)
   - Server port count: 500
   - RPC listen base: 0
   - RPC listen count: 1

6. For the nodes that have been determined to be the cluster master and the backup cluster master and for additional nodes that will run the MDS, select the Message Distribution Service (MDS) tab and make a note of the following parameter values:

- Message distribution IP address = the host name or IP address (for example, qareport1 or 172.20.72.93)
- Message distribution Port = the node’s port number (for example, 8000)
- Max. Concurrent SOAP Requests = 1000

You must take all nodes, except for the cluster master, offline before adding them to a cluster system.

7 Select System from the left-side panel and click Stop to take the node offline.

NOTE: Nodes must be offline before they can join a cluster.

The following steps will create the actual cluster.

8 Using Management Console, log in to the System Administration page of the iServer machine that will be the cluster master.

9 Add the master machine to the cluster first by clicking System > Create Cluster and click OK.

By default the master node has MDS, View Service, and Factory Service enabled on it. By default the cluster master is the owner of the Volume.

For more information, see the “Creating a cluster from stand-alone Actuate iServer” section in the *Administering Actuate iServer System* manual in the Actuate folder of the *Siebel Third-Party Bookshelf*.

10 Select Servers from the left-side panel.

11 Select the Partitions tab and enter a value for the partition path. This partition path was noted in Step 2 on page 88.

This is the valid Encyclopedia directory path that will be accessible to the cluster master and all nodes. You must specify the same partition for the cluster master and for nodes that you add to the cluster.

Make sure that all machines have read/write access to the Encyclopedia for the account that is used to install/run the Actuate processes on each node.

Click Test to verify that the server can access the directory specified.

12 Click Apply.

The following steps will add nodes to the cluster.

Each node is added individually to the cluster. The nodes are added from the cluster master's System Administration screen.

13 Select Servers from the left-side panel.

14 Click Add Server.

15 On the General tab, add the information from Step 5 on page 88:

- Hostname or daemon IP address: = enter the node's host name (for example, Node1)
- Daemon listen port: = enter the value for port
- Server IP address: = enter the node's host name or IP address

16 Click the check boxes to enable Message distribution service, View service and for Factory service as needed.

NOTE: MDS must be enabled on the node designated as the cluster master backup.

17 If you have enabled the Message distribution service in Step 16 on page 89, then select the Message Distribution Service tab and add the information from Step 6 on page 88:

- Message distribution IP address = enter the node's machine name (for example, 172.20.72.93). The default is usually the machine name.
- Message distribution Port = Enter the node port number (for example, 8000)

18 Select the Partitions tab and enter a value for the partition path. This partition path was noted in Step 2 on page 88.

Click Test to verify that the server can access the directory specified.

19 Click Apply.

For each node to be added to the cluster, repeat Step 13 on page 89 through this step.

The steps below set up the cluster master backup.

20 Select System from the left-side panel and click Properties.

NOTE: For the node designated as the cluster master backup, make sure that the MDS is enabled.

21 Select the Backup Master Assignment tab. From the Available Servers list, highlight the server you designated as the cluster master backup and move to the Assigned backup master servers list.

22 Click Apply.

The steps below will set up the backup volume server.

23 Select System Volumes from the left-side panel.

24 Click the volume and select Properties.

25 Select the Server Assignments tab.

26 From the list of Available servers, highlight the server or servers that you have designated as the backup owner of the volume and move to Assigned backup servers.

27 Click OK.

Setting the Configuration Home Parameter

After the cluster is created, you will complete the following tasks to set the configuration home parameter.

To set the Configuration home parameter

1 From the Management Console of the Reports Server acting as the cluster master, log in to System Administration.

2 Select System from the left panel and click Properties.

3 Select the General tab and go to the Configuration home section.

4 Select the partition from the drop-down menu, for example DefaultPartition.

5 Click OK.

For more information, see the "Specifying the location of the configuration file" section, and the "About the Actuate iServer cluster configuration file" subsection in the *Administering Actuate iServer System* manual in the Actuate folder of the *Siebel Third-Party Bookshelf*.

You must bring the nodes in the cluster online. For more information on how to start a node, see the "Starting and stopping Actuate iServer nodes" subsection in the *Administering Actuate iServer System* manual in the Actuate folder of the *Siebel Third-Party Bookshelf*.

Configuring the Siebel Servers to Interact with the Siebel Reports Server Cluster Environment

After creating the cluster, you will configure the Siebel Server to interact with the Siebel Reports Server in its cluster environment.

To configure the Siebel Servers to interact with the Siebel Reports Server cluster environment

1 Create the rptinfo.ini file.

   In the Siebel Server's BIN directory, create an rptinfo.ini file and specify the desired MDS refresh interval in seconds. For example:

```
RefreshInterval=60
```

2 Enable the Actuate Parameters from the Siebel application:

   a From the application-level menu, select Navigate > Sitemap > Administration - Server Configuration > Servers.

   b Choose the required component from the Components tab in the second list applet.

   c For the Actuate Server Report Host parameter, specify the cluster master and the cluster master backup servers.

      The cluster master node should be listed first followed by the cluster master backup nodes, using commas to separate each node (*server1:port_number/volume_name*, *server2:port_number/volume_name*).

      For example:

```
Actuate Server Report Server Host =
bpt4000i007:8000/bpt4000i007,bptx330i019:8000/bpt4000i007
```

   d Restart the server for these changes to take effect.

The other parameters are the same as for a regular Reports Server configuration.

Siebel Reports Server and Firewall Planning

If your network infrastructure includes a demilitarized zone (DMZ), you must enable specific ports on Active Portal and iServer.

This requirement applies in the following circumstances:

- The DMZ is bounded by an outer firewall and an inner firewall. The outer firewall filters traffic between the Internet and the reverse proxy server in the DMZ. The inner firewall filters communications between the reverse proxy server and the Siebel deployment.
- Actuate Active Portal is installed in the DMZ.
- Actuate iServer is installed behind the inner firewall along with the Siebel deployment.

You must enable ports as follows to make sure the Reports Server functions normally:

- On the outer firewall, enable the Actuate HTTP Service Communications port. The default port number is 8700.
- On the inner firewall, activate the iServer port. The default is 8000.
- On the inner firewall, activate the PMD port. The default is 8100.

These port numbers are defaults and you can configure them.

Planning Batch Processing When Using Siebel Remote

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

This 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. This increases 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 can 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. This 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.

For additional information on understanding gaps in the transaction log, see the Technical Note: Siebel Remote Transaction Gaps on SupportWeb.

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.

If skipped transactions occur when no batch jobs are running, the gap is most likely permanent and is caused by failed commits. If the gap is permanent, nothing is lost, and mobile users receive the correct information.

Set a Lower BatchSize for Assignment Manager

Setting a lower BatchSize reduces the number of records processed before each commit. This reduces the commit times and the risk of exceeding the wait-time. If performance requires that you increase the BatchSize parameter, do so only after analyzing the number of Assignment Manager threads that you have under average and peak workloads. The lower the number of 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

No index available.