Siebel System Monitoring and Diagnostics Guide

Siebel Innovation Pack 2013

Version 8.1/8.2

September 2013

ORACLE

Copyright © 2005, 2013 Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Contents

Chapter 8: Collecting Siebel Environment Data

Appendix A: List of Statistics and State Values

1 What's New in This Release

What's New in Siebel System Monitoring and Diagnostics Guide, Version 8.1/8.2

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

What's New in Siebel System Monitoring and Diagnostics Guide, Version 8.1, Rev. C and 8.2, Rev. A

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

Table 1. What's New in Siebel System Monitoring and Diagnostics Guide, Version 8.1, Rev. C and 8.2, Rev. A

Topic	Description
"About Using SQL Tagging to Trace Long-Running Queries in Siebel Business Applications" on page 54	Modified topic. The SQL tagging feature is available in Siebel CRM version 8.1.1.1 and later, and Siebel CRM version 8.2.
"About Siebel Process Failure Diagnostics" on page 58 "How Siebel Process Failure Diagnostics Work" on page 59 "Scenario for Working with Siebel Process Failure Diagnostics" on page 59 "Investigating Failed Siebel Server Processes" on page 60 "Example of Investigating a Failed Siebel Server Process" on page 61	New topics. They describe the Process Failure Diagnostics view that helps administrators to allocate system resources and maintain an efficient and reliable environment. If a process failure is suspected, navigate to this view to confirm and drill down into the potential source of any detected occurrence. This feature requires Siebel CRM version 8.1.1.10 or later, or Siebel CRM version 8.2.2.3 or later.
"About Component Log Files" on page 67	Modified topic. It provides more detailed information about the component log file naming convention.
About Siebel Server Task IDs	Deleted topic. It is no longer applicable.

2 Configuring SWSE Logging and Monitoring

This chapter describes configuring the Siebel Web Server Extension (SWSE) for logging and monitoring purposes. It includes the following topics:

About SWSE Logging

The Siebel Web Server Extension (SWSE) generates one or more log files as a result of connection attempts with the Siebel Server. These log files reside in *SWEAPP_ROOT*\log directory (*SWEAPP_ROOT* is the installation directory of the Siebel Web Server Extension).

The format of the SWSE log filename is as follows:

```
PlatformPrefix_Timestamp_Proc_ID.log
```

where:

- *PlatformPrefix* is ss.

  NOTE: PlatformPrefix is a siebsnap (ss) command. For more information about siebsnap commands and SWSE log filename formats, see "About SDDC Output" on page 151.

- *Timestamp* is the date of the log file in the format: YYMMDD.
- *Proc_ID* is the operating system process ID for the Web server that hosts the SWSE.

Depending on the logging level you choose, these files record errors, warnings, and general information. You can set log levels using environment variables on the computer hosting the Web server. For information on configuring SWSE logging, see "Configuring SWSE Logging" on page 12.

NOTE: SWSE does not use logging event levels as defined for Siebel Server and Siebel Server components.

Events such as Web server failures or invalid configuration of the Siebel Web Engine are captured in these logs. Analyzing the log files can provide clues for troubleshooting problems with the SWSE.

Configuring SWSE Logging

Use the following procedure to configure SWSE logging. The former method of configuring SWSE logging by setting a parameter in the eapps.cfg file is no longer valid.

To configure SWSE logging

1. On the computer running the Web server, set the following environment variable to the given value:

   - SIEBEL_LOG_EVENTS is 4 (or higher)

   For further information on setting environment variables, see *Siebel System Administration Guide*.

2. Optionally, set the following environment variables to add detailed information on session manager and SISNAPI tracing in the SWSE log file:

   - SIEBEL_SESSMGR_TRACE is 1
   - SIEBEL_SISNAPI_TRACE is 1

   NOTE: Configuring detailed logging uses a greater amount of disk space. Make sure sufficient disk space is available.

   For further information on these environment variables, see "About Environment Variables for System Logging" on page 85.

3. Stop and restart the Web server for these environment variables to take effect.

NOTE: Reset the original values of these variables after troubleshooting the SWSE.

Parsing a SWSE Log File Cookie

The SWSE log file encodes system data in hexadecimal format at the end of the user session cookie. An example cookie follows:

```
cookie (siebel.TCPIP.NONE.none://172.19.14.20:2320/siebel/eCommunicationsObjMgr/
!24.8c4.1779.3db56d28)
```

Table 2 provides descriptions of the SWSE cookie data.

Table 2. Example SWSE User Session Cookie Data Description

Cookie Data in Hexadecimal Format	Data Type	Decimal Format	Description
24	Server ID	36	ID number for Siebel Server.
8c4	OS PID	2244	Operating system ID number for of the Siebel Application Object Manager (AOM) that handles the user session.

Table 2. Example SWSE User Session Cookie Data Description

Cookie Data in Hexadecimal Format	Data Type	Decimal Format	Description
1779	Task ID	6009	Siebel task ID for the application OM that handles the user session.
3db56d28	Date	1035300136	Operating system timestamp format of the action for a cookie in POSIX (UNIX) time.

About SWSE Monitoring

Monitor the Siebel Web Server Extension (SWSE) by configuring and reading the SWSE Statistics page. This HTML page provides current information about the operations and communications of the SWSE, which allows system administrators to have a better understanding of the use of the Web server. Each of the sections of the Statistics page lists measurable objects, their values, mean values, and standard deviations.

CAUTION: As the SWSE Statistics page provides sensitive information about the type of requests running and potentially active sessions, it is strongly recommended that this page be protected with the Web server's, or a third party's, authentication mechanism.

Configuring the SWSE Statistics Page

The SWSE Statistics page is configured in the [swe] section of the eapps.cfg file by the parameter StatsPage. By default this value is:

```
[defaults]

StatsPage = _stats.swe
```

CAUTION: For security reasons, change the default value for the StatsPage parameter. Otherwise, others without permission can access this data. Make sure the new filename retains the .swe suffix. For further information on security, see "About SWSE Logging" on page 11 and *Siebel Security Guide*.

The eapps.cfg file contains an additional parameter that defines content in the SWSE Statistics page: SessionMonitor.

SessionMonitor specifies if statistics are gathered on current sessions and then reported to the application's SWSE Statistics page. If SessionMonitor is enabled (TRUE), when sessions are created they are entered into the statistical repository and appear on the application's SWSE Statistics page. This setting allows system administrators to determine who is logged onto the system at any given time, and to determine the session ID with a given user in a non-debug log level. However, performance is slightly degraded by using this feature. If SessionMonitor is disabled (FALSE), sessions are not monitored by the statistical repository and do not appear in an application's SWSE Statistics page.

This parameter is configured in the [swe] section of the eapps.cfg. The default value is FALSE and appears as follows:

```
[swe]

SessionMonitor = FALSE
```

Accessing the SWSE Statistics Page

The Siebel Web Server Extension (SWSE) Statistics page is generated by the SWSE plug-in. To access the SWSE Statistics page, enter the following URL in a Web browser:

http://*host*/*application*/_stats.swe

In addition to defining the name of the SWSE Statistics page accessory handle, you can configure if currently active sessions appear on the page as well. For information about monitoring currently active sessions, see information on the SessionMonitor parameter in "Configuring the SWSE Statistics Page" on page 13.

When accessing the SWSE Statistics page URL, additional parameters can be appended to the URL, which modify the display and content of the page.

Statistical Page Verbosity Option. This option allows the user to dictate the amount of information to appear in SWSE Statistics page. There are three settings as shown in Table 3.

Table 3. Statistical Page Verbosity Settings

Verbose Parameter Setting	Description
Verbose=Low	Default value if not present. Displays only system and application-level statistics.
Verbose=Medium	Displays the low-setting information, plus the lock statistics.
Verbose=High	Displays the medium-setting information, plus currently active operations to the Siebel Server.

Statistical Page Reset Option. This option allows the user to dictate if the statistics are reset after viewing. There are two settings as shown in Table 4.

Table 4. Statistical Page Reset Settings

Verbose Parameter Setting	Description
Reset=True	Resets noncounter and current operational statistics.
Reset=False	Default value if not present. Does not reset current operational statistics.

An example of the SWSE Statistics page request with parameters:

- http://*host*/*application*/_stats.swe?Verbose=High&Reset=True

  This request displays the System Stats, Applications, Current Sessions, Locks, and Current Operations Processing statistical categories and then resets noncounter and current operational statistics.

- http://*host*/*application*/_stats.swe?Reset=True

  This request displays the System Stats and Applications statistical categories and then resets noncounter and current operations statistics.

SWSE Statistics Page

The individual events and objects measured on the SWSE Statistics page are described in the following list. For examples of these metrics, see "Example of SWSE Statistics Page" on page 17.

Open Session Time. This event reflects the total amount of time it took to open a session. In the general stats section, the count is the number of times a session was opened and the mean reflects the average time it took to open a session.

Response Time (waiting for service event). This event measures the time it takes to receive a callback response from the Siebel server. This event functions with CTI and internal login callbacks. A callback is a mechanism used by the Siebel Server to initiate communication with the plug-in.

Close Session Time. This event reflects the amount of time it takes to close a session. Closing the session might involve signaling to the session manager to close the session. The session manager might or might not close the TCP/IP connection.

Request Time (waiting for service method to process). This event is the amount of time it takes to submit a request to the Siebel Server and to get a response back. For example, if the user (on the browser) clicked on a button then the plug-in receives the request and invokes a service on the Siebel Server. The value for Request Time is the total amount of time for invoking that service.

Applications. This section displays information about the various applications, for example, session life span and number of attempts to use the application.

Current Sessions. This section contains information about the current active sessions open. The parameter SessionMonitor must be set to True for this to take effect (see "Configuring the SWSE Statistics Page" on page 13 for further information on SessionMonitor). If verbose mode is used, then this section also displays the anonymous sessions (see "Accessing the SWSE Statistics Page" on page 14 for further information on verbose mode).

Current Operations Processing. Use the following information when troubleshooting a process that might have stopped responding.

The Current Operations Processing section contains a table that shows current requests that are in progress. Table 5 shows the operations that are running and the duration of each operation (in seconds). Requests highlighted in bold have been running for more than 10 seconds. A request highlighted in bold with a large duration value indicates that this request might not be responding. If a request never completes, then it has effectively stopped responding.

Table 5. Example of a Current Operations Processing Table

Operation	Duration
<server>://172.20.232.19:2320/siebel/SCCObjMgr/!7.fb8.ddde. <snip>	**3888.0282**
<server>://172.20.232.19:2320/siebel/SCCObjMgr/!8.f54.df75.3c07ef90 <snip>	**20.8209**
<server>://172.20.232.19:2320/siebel/SCCObjMgr/!8.f54.df75.3c07ef90 <snip>	0.2796

For example, the first operation in Table 5 has most likely stopped responding. The second operation in the table has been running for over ten seconds, so it might also have stopped responding.

Both application and database server delays can exhibit this behavior. Typically, if the SWSE Statistic page cannot be accessed, then it is likely that the Web Server itself has stopped responding.

For more information about the Current Operations Processing section of the SWSE Statistic page, see Siebel Troubleshooting Steps on My Oracle Support.

Locks. Programming locks synchronize internal SWSE processing. If you access the SWSE statistics with verbose mode set to medium or high, the following locks statistics appear:

- /application/InitLock. Used by SWSE to synchronize initialization of configuration parameters.
- /application/anonSessionLock. Used by SWSE to synchronize handling of anonymous sessions.
- SWEWebPublishMutex. Used by SWSE to synchronize the loading of web images.

Table 6 provides descriptions of some SWSE statistics and is followed with the definition of those statistics.

Table 6. SWSE Statistics Descriptions

Statistic	Type of Statistic		Description
	General	Frequency	
Count	Yes	No	Accumulative count of the events
Mean	Yes	No	Average value for the event
	No	Yes	Average period for which one such event occurs (in seconds)
Standard deviation	Yes	No	Standard deviation for the event
	No	Yes	Standard deviation (in seconds)

An example follows Table 6 describing how these statistics might occur with two different types of events. Statistic results can vary depending on the session type. For example, for an anonymous session requested from the pool event type, the statistics provide the following information:

- **General statistics count.** The accumulative count of the anonymous session requests.
- **General statistics mean.** The average value for anonymous session requests (the value is 1 or greater).
- **General statistics standard deviation.** The standard deviation for anonymous session requests (the value is zero [0] or greater).
- **Frequency mean.** The average period between anonymous session requests in seconds.
- **Frequency stddev.** The standard deviation in seconds.

However, for an open session time event type, the statistics provide the following information:

- **General statistics count.** The accumulative open session times.
- **General statistics mean.** The average value for an open session time.
- **General statistics standard deviation.** The standard deviation for an open session time.
- **Frequency mean.** The average period between open session events in seconds.
- **Frequency standard deviation.** The standard deviation in seconds.

Example of SWSE Statistics Page

A sample SWSE Statistics page is reproduced in the tables in this topic. The information contained in these tables encompasses one SWSE Statistics page.

Table 7 shows sample system statistics.

Table 7. System Statistics Sample

Event	Value (Seconds)	General Statistics (Count, Mean, Standard Deviation)	Frequency (Mean, Standard Deviation)
Open Session Time	191.6682	12 15.9723 34.4210	61.9689 128.9318
Response Time (waiting for service event)	0.0000	0 0.0000 0.0000	0.0000 0.0000

Table 7. System Statistics Sample

Event	Value (Seconds)	General Statistics (Count, Mean, Standard Deviation)	Frequency (Mean, Standard Deviation)
Close Session Time	0.0000	0 0.0000 0.0000	0.0000 0.0000
Request Time (waiting for service method to process)	349.9513	23 15.2153 70.4652	3374.4503 16020.5422

Table 8 shows sample application statistics.

Table 8. Application Statistics Sample

Application Name	Totals (Seconds)	General Statistics (Count, Mean, Standard Deviation)	Frequency (Mean, Standard Deviation)
/echannel/	13.0000	13 1.0000 0.0000	5970.1458 21303.1122
/echannel/Session Lifespan	0.0000	0 0.0000 0.0000	0.0000 0.0000

Table 9 shows sample current sessions.

Table 9. Current Sessions Sample

Event	Total Time (Seconds)	General Statistics (Count, Mean, Standard Deviation)	Frequency (Mean, Standard Deviation)
siebel://test:2320/siebel/objmgr/test/!1.64c.14.3bb0e99fuser0	3.9228	4 0.9807 0.8953	85.9297 168.6426
siebel://test:2320/siebel/objmgr/test/!9.34b.1fe.3bbf349fuser1	338.4631	9 37.6070 112.8092	59.4458 116.0594
siebel://test:2320/siebel/objmgr/test/!1.56.1ef.4c0a0e99fuser2	3.3424	3 1.1141 0.8227	25665.0354 44450.4096

Table 10 shows a sample current operations processing.

Table 10. Current Operations Processing Sample

Operation	Duration (Seconds)
NewAnonSession_00000022_499	0.9581
Open Session Time_00000023_499	0.9580

Table 11 shows a sample lock.

Table 11. Locks Sample

Event	Total (Seconds)	General Statistics (Count, Mean, Standard Deviation)	Frequency (Mean, Standard Deviation)
/*application*/InitLock	0.0000	1 0.0000 0.0000	0.0002 0.0000
/*application*/anonSessionLock	0.0003	25 0.0000 0.0000	3104.4834 15393.1114
SWEWebPublishMutex	0.0000	2 0.0000 0.0000	0.8005 1.1318

3 Monitoring Siebel Server Run-Time Operations

Monitoring Oracle's Siebel Server run-time operations is a necessary, on-going aspect of administering a Siebel application. Use metrics such as log files, state values, and statistics to monitor the Siebel application performance.

This chapter includes the following topics:

About Siebel Server States

After installation, a Siebel Server is always in one of the following states when connected to the Server Manager component (alias ServerMgr):

- **Starting Up.** Indicates that the Siebel Server is in the process of starting up. When this process is complete, the state changes to Running.
- **Running.** Indicates that the Siebel Server is running and that Siebel Server components can operate. This is the normal mode of operation for the Siebel Server. When the Siebel Server Service starts, it sets the Siebel Server to the Running state by default (depending on the value of the Auto Startup Mode Siebel Server-level parameter, which defaults to TRUE).

  When the Siebel Server starts, its components are enabled and the default number of tasks is instantiated for the background mode components (the number of tasks is determined by the value of the Default Tasks parameter for each component).
- **Shutting Down.** Indicates that the Siebel Server is in the process of shutting down. When this process is complete, the state changes to Shutdown.
- **Shutdown.** Indicates that the Siebel Server is running, but component tasks are not currently running (other than the Siebel Server Manager component, which is operational whenever the Server Manager is connected) and new tasks are not allowed to start. The only processes that can run when the Siebel Server is in a Shutdown state are the Siebel Server System Service itself and the Server Manager for a Siebel Server Manager client.

  Shut down the Siebel Server using the Server Manager whenever you want to shut down the:

  - Server computer on which the Siebel Server is running. This allows a clean shutdown of each Siebel Server component.
  - Siebel Server to perform maintenance.
  - Siebel Server to perform an automatic upgrade on the Siebel Server's software using Siebel Upgrade Wizard.

  NOTE: Individual components might be shut down or disabled without having to shut down the entire Siebel Server.

If the Siebel Server is not connected to the Server Manager component (alias ServerMgr), the following states are applicable:

- **Not available.** Indicates that the Siebel Server has not been started. Indicates that the Server Manager cannot connect to the Siebel Server; you cannot run any tasks or perform any administrative functions on that Siebel Server.
- **Connect Failed**. Indicates that Server Manager is able to get the connect string for the ServerMgr component from the Siebel Gateway Name Server but is unable to connect to the Siebel Server.
- **Handshake Failed**. On startup, Server Manager sends a handshake request to the Siebel Server for the ServerMgr component. If that request fails then this state occurs. Also, if the ServerMgr component on that particular Siebel Server cannot start any more tasks (because it has reached Maximum Tasks (alias MaxTasks) number of tasks) for the administration clients, this state occurs. For more information on the MaxTasks parameter, see *Siebel System Administration Guide* and *Siebel Performance Tuning Guide*.

- **Login Failed**. Server Manager connects to every Siebel Server for authentication. If the authentication fails for any Siebel Server, the Login Failed state appears.
- **Disconnected**. When Server Manager connects to the Siebel Server, the Siebel Server starts a task for the ServerMgr component. If that task exits (because of a malfunction or other problems), the Disconnected state appears.

Siebel Server Status Fields

Each Siebel Server record has three fields in which the Siebel Server status appears. Table 12 provides the Siebel Server status fields.

Table 12. Siebel Server Status Fields

GUI Column Name	Command-Line Interface Column Name	Description
Server State (Internal)	SBLSRVR_STATE	The state of the Siebel Server.
State	SV_DISP_STATE	The state of the Siebel Server using the appropriate language code.
State (Icon)	Not applicable	A stoplight representation of the state of the Siebel Server. Green indicates normal conditions. Red indicates a non-operational condition. Clicking the icon field reveals the state value associated with the color code. **NOTE:** The State (Icon) field is blank when you are not connected to a Siebel Server.

About Siebel Server Component Group States

A component group might be in one of several states. The run state is dependent on the enable state; only component groups that have an Online enable state when the Siebel Server was started can have a run state of Online or Running:

- **Online.** Every component within the component group is enabled to run tasks.
- **Running.** Every component within the component group is enabled, and at least one component within the component group is running a task.
- **Shutdown.** Every component within the component group is shut down. Tasks cannot run for any components within the component group.
- **Part shutdown.** At least one component within the component group is shut down or shutting down.
- **Offline.** Every component within the component group is offline.
- **Part offline.** At least one component within the component group is offline or unavailable.

- **Starting up.** At least one component within the component group is starting up.

Server Component Group Status Fields

Each Siebel Server component group record has three fields in which the status appears as shown in Table 13.

Table 13. Siebel Server Component Group Status Fields

GUI Column Name	Command-Line Interface Column Name	Description
State	CA_RUN_STATE	The state of the server component group using ENU language code.
Run State (internal)	CA_RUN_STATE	The state of the server component group using the appropriate language code.
State (Icon)	Not applicable	A stoplight representation of the state of the server component group. Green indicates normal conditions. Yellow indicates a temporary non-operation condition. Red indicates a non-operational condition. Clicking the icon field reveals the state value associated with the color code.

About Siebel Server Component States

A Siebel Server component might be in one of the following states: Starting Up, Online, Running, Offline, Shutting Down, Shutdown, or Unavailable.

The Siebel Server component state is dependent on the assignment state of the component group to which it belongs; only Siebel Server components within assigned component groups when the Siebel Server was started can be Running or Online:

- **Starting Up.** Indicates that the Siebel Server component is in the process of starting up. When this process is complete, the state changes to Online. When a new task is started for the component, the component state changes to Starting Up during the initialization phase and then to Running.
- **Online.** Indicates that tasks are currently not running for the Siebel Server component, but new tasks might be started through the Siebel Server Manager (or in response to client requests, for interactive-mode components). When the Siebel Server starts, components for which processes are *not* started by default are online.
- **Running.** Indicates that tasks are currently running for the Siebel Server component on the Siebel Server, and new tasks are allowed to start (up to the value of the Maximum Tasks parameter for the component). When the Siebel Server starts up, background-mode components for which processes are started by default (components with a Default Tasks parameter set to a nonzero value) start.

- **Offline.** Indicates that new tasks might not be started for the component, though current running tasks can continue running (for background-mode components) or run to completion (for batch-mode and interactive-mode components).

  You might want to disable an individual component to perform a system maintenance operation outside of the Siebel Server. For example, you might disable the Synchronization Manager component to do a file system reorganization on the docking subdirectory.

  To minimize the number of multithreaded processes started on the Siebel Server, you might want to disable components that you do not plan to run.

  You might also want to disable components due to database licenses. If you have exceeded the maximum licensed connections for your database, then you might want to disable the Siebel Server components that you plan not to use. You must only disable components for which you do not plan to run tasks across the entire enterprise. Setting the Min MT Servers parameter to 0 for multithreaded Siebel Server components renders the server component unable to run tasks.

  An offline component might be set to Online (or Started, if there are still tasks running for the offline component) or Shutdown, in which case, any running tasks are stopped as cleanly as possible.

- **Shutting Down.** Indicates that the Siebel Server component is in the process of shutting down. When this process is complete, the state changes to Shutdown.

- **Shutdown.** Indicates that processes are not running for the component and new tasks might not be started. Each task running when the component shuts down is stopped as soon as possible. Components are set to Shutdown when the Siebel Server shuts down, with the exception of the Siebel Server Manager component, which remains Online to perform administrative commands executed by the Siebel Server Manager. Background-mode components that are set to Shutdown but have a Default Tasks parameter set to a nonzero value might be set to Online or Started.

- **Unavailable.** Indicates that processes are not running for the component when a Siebel Server process is running. Multithreaded Siebel Server components change to an Unavailable component state when the Min MT Servers parameter is set to a value greater than 0 and no Siebel Server processes are actually running for that component. In this case, the Siebel Server component might exit with an error and become unavailable because it failed to initialize. Siebel Server components might also go into this state if the database connection is down. In this case, you must restart the Siebel Server component after the database connection has been reestablished.

Server Component Status Fields

Each server component record has two fields in which the status appears as shown in Table 14.

Table 14. Server Component Status Fields

GUI Column Name	Command-Line Interface Column Name	Description
State	CP_DISP_RUN_STATE	The state of the Siebel Server component using the appropriate language code.
State (Icon)	Not applicable	A stoplight representation of the state of the Siebel Server component. Green indicates normal conditions. Yellow indicates a temporary non-operation condition. Red indicates a non-operational condition. Clicking the icon field reveals the state value associated with the color code.

About Siebel Server Task States

A Siebel Server task is an instantiation of a Siebel Server component. To run a Siebel Server task, you must run a component job, which requests one or more Siebel Server tasks to run. For information on component jobs, see *Siebel System Administration Guide*.

A Siebel Server task might be in one of four fundamental states: Running, Paused, Stopping, or Completed.

- **Running.** Indicates that the task is executing normally. While the task is running, it periodically updates its task status, a component-generated message that indicates the task progress (or phase of operation).
  - Background mode component tasks run until stopped manually, or until the Siebel Server or the server component shuts down.
  - Batch mode component tasks run to completion when their assigned unit of work is done.
  - Interactive mode component tasks run until the client signs off from the connection (or until the task, server component, or Siebel Server is shut down).

  You might explicitly stop any currently running component task.
- **Paused.** Indicates that the task has been temporarily placed in a suspended state. A paused task does not exclusively hold any shared system resources (such as file locks or database locks), or expend any processor or I/O cycles. You might choose to pause a running task to temporarily free up the system to process other critical tasks without having to restart the entire task. You might then resume or stop the paused task.

  NOTE: Only tasks from certain component types can be paused. For a list of these component types, see *Siebel System Administration Guide*.

- **Stopping.** Indicates that the task has been instructed to stop, or the server component or Siebel Server is being shut down. Occasionally, the shutdown process might take a while, in which case you might issue another Stop command, and the shutdown is forced (this state might appear as Forcing Shutdown). After a task has been instructed to stop, it might not be resumed.
- **Completed.** Indicates that the task is no longer running. After a task is completed, it might not be restarted, though you might start a new task for the same server component. Several variations exist for the Completed state, depending on the manner in which the task finished processing:
  - *Completed* indicates that the task ran to completion and exited normally (batch mode and interactive mode tasks only).
  - *Exited with Error* indicates that the task encountered an error during its processing (such as bad input values or database errors). In this case, the Task Status field displays the error identifier for the error that has occurred.
  - *Killed* indicates that the task was not able to shut down cleanly, and you forced the task to shut down.

About Task Status Fields

Each Siebel Server record has three fields in which the Siebel Server status appears. Table 15 provides the task status fields.

Table 15. Task Status Fields

GUI Column Name	Command-Line Interface Column Name	Description
State	TK_RUNSTATE	The state of the task using the appropriate language code.
Status	TK_STATUS	Every component task sets various state values during the course of its operation. The Status column in the tasks view and the TK_STATUS column in the command-line interface displays the value for the state value Task Status (alias TaskStatus).
State (Icon)	Not applicable	A stoplight representation of the state of the task. Green indicates normal conditions. Yellow indicates temporary non-operational conditions. Red indicates a non-operational condition. Clicking the icon field reveals the state value associated with the color code.

About Component Job States

After the creation of a component job, it is always in one of the states in the following list. For further information on starting component jobs, see *Siebel System Administration Guide*. For further information on monitoring component job status, see "Monitoring Component Job Status" on page 40.

- **Creating.** Indicates the component job record is in the process of being defined.

- **Queued.** Indicates the component job record was started and is scheduled to run. The component job field Scheduled Start defines when the component job runs.
- **Active.** Indicates the scheduled component job is running.
- **On Hold.** Indicates the component job is on hold and will not run at the Scheduled Start time. Only component jobs in the queued state can be put on hold.
- **Cancelled.** Indicates the component job is cancelled. Only component jobs in the queued or on hold state can be cancelled.
- **Canceling.** Indicates the component job is in the process of being cancelled.
- **Error.** Indicates the component job ran, but encountered an error during operation.
- **Success**. Indicates the component job ran and completed successfully.
- **Completed.** Indicates that repeating component jobs completed successfully.
- **Expired.** Indicates the component job has expired. The component job field Expiration Date defines when the component job expires.
- **Parent Request Cancelled.** Indicates the first component job of a repeating component job was cancelled. The first component job of a repeating component job is considered the parent job.
- **Parent Request On Hold.** Indicates the first component job of a repeating component job is on hold. The first component job of a repeating component job is considered the parent job.

About User Sessions

User sessions include data on any user logged into the Siebel Server as well as sessions created by the Siebel application. User sessions comprise all interactive component tasks.

User sessions run based on a Siebel Server component task. Therefore, user sessions have the properties of Siebel Server component tasks. The Session ID field of an individual user session shares the same ID number as the Task ID of the component task that runs the session. That is, information on user sessions can be viewed as either a user session or a task.

For information and procedures on monitoring user sessions, see “Monitoring User Session Status” on page 40. For information and procedures on monitoring tasks, see “Monitoring Server Component Task Status” on page 37.

About Siebel Application Statistics

Various statistics are recorded at the task level for every Siebel Server component task. You might use these statistics to:

- Monitor the progress and performance of a task, component, or Siebel Server
- Optimize system performance

When the task completes its operation, task-level statistics (gathered dynamically during the operation of a task) roll up to the component and Siebel Server levels.

Two types of statistics exist for task-level Siebel Server statistics:

- **Subsystem statistics.** Common to every component process (such as process management, networking, database access, and file I/O) and tracked for each component task.
- **Component-specific statistics.** Only applicable to the component for which the statistics are defined.

When a task for a component completes its operation, both generic and component-specific statistics roll up to the component level. Only generic statistics roll up to the Siebel Server level.

Statistics on the component level includes data for completed tasks on interactive and batch mode components. Statistics for component tasks that are still running are not included. Check the tasks directly to monitor statistics for running tasks on interactive and batch mode components. For information on monitoring task statistics, see "Monitoring Server Component Task Statistics" on page 39. For background mode components, the statistic rollup behavior is slightly different because the component tasks are never complete. For background components, the component statistics change whenever a statistic value is updated by the running component task. For a listing and brief descriptions of Siebel application statistics, see Appendix A, "List of Statistics and State Values."

NOTE: If some Siebel application statistics are not visible, set the Show Advanced Objects (alias ShowAdvancedObjects) parameter to TRUE for the server component Server Manager (alias ServerMgr). For further information on advanced objects, see *Siebel System Administration Guide*.

About Siebel Application State Values

State values contain information about the current operation of a task or the component for which the task is running. Component tasks periodically update their state values to indicate information about their current processing, such as the current phase of operation. State values are defined at the component and task levels. Component-level state values refer to the state of the component as a whole. Task-level state values refer to the state of an individual process for a Siebel Server component.

Two types of state values exist for components and component tasks:

- **Subsystem state values.** Kept for every component (such as Component Start Time and Component Stop Time) and component task (such as Task Start Time and Task Stop Time) that uses that subsystem.
- **Component-specific state values.** Kept for every component and component task. Only applicable to the component for which they are defined.

Monitoring Siebel Enterprise Server Status

Monitor the status of Siebel Servers in a Siebel Enterprise Server by using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For configuration tasks and background information on the Siebel Enterprise Server, see *Siebel System Administration Guide*.

To monitor a Siebel Enterprise Server using the Server Manager GUI

- Navigate to the Administration – Server Management screen, Enterprises, and then the Servers view.

  The following information appears:

  - The name and description of the Siebel Enterprise Servers available are in the Enterprise Servers list.
  - The state of the Siebel Servers for the selected Siebel Enterprise Server are available in the Servers list. For details on Siebel Server states, see "About Siebel Server States" on page 22.
  - The state of the Siebel Server components for the selected Siebel Server are available in the Components list. For details on Siebel Server component states, see "About Siebel Server Component States" on page 24.

To monitor Siebel Enterprise Server using srvrmgr

- At the srvrmgr program prompt, enter:

  ```
  list servers
  ```

CAUTION: Make sure you do not start the Server Manager command-line interface program for a particular Siebel Server; that is, do not start the Server Manager command-line interface with the /s flag.

For details on starting, running, and configuring the Server Manager command-line interface program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Status

Monitor the status of Siebel Servers by using the Server Manager GUI or Server Manager command-line interface program (srvrmgr program). The following topics describe procedures that monitor the Siebel Server:

For background information Siebel Servers, including running and configuring procedures, see *Siebel System Administration Guide*.

Monitoring Siebel Server State

Monitor the status of a Siebel Server by using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For details on the possible states of the Siebel Server, see "About Siebel Server States" on page 22. For information on monitoring other Siebel Server run-time operations, see "Monitoring Siebel Server Status" on page 30.

To monitor the Siebel Server state using the Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Servers view.
2. In the Servers list, select the Siebel Server of interest.
3. Review the state of the selected Siebel Server by viewing the State (Icon) or Server State fields.

To monitor the Siebel Server state using srvrmgr

- At the srvrmgr program prompt, enter:

```
list servers
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Component Groups

Monitor the status of component groups for a Siebel Server using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For details on Siebel Server component group states, see "About Siebel Server Component Group States" on page 23. For information on monitoring other Siebel Server run-time operations, see "Monitoring Siebel Server Status" on page 30.

To monitor component groups using Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Servers view.
2. In the Servers list, select the Siebel Server of interest.
3. From the view tabs, click Component Groups.
4. Review the state of the component groups for the selected Siebel Server by viewing the State (Icon) and State fields of each component group record.

To monitor component groups on srvrmgr

- At the srvrmgr program prompt, enter

```
list component groups for server siebel_server_name
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Log Files

Monitor the log files for a Siebel Server using the Server Manager GUI. You can also review Siebel Server log files by manually accessing the file or querying the file with the Log File Analyzer (LFA) utility.

For background information on:

- Siebel Server log files, see "About Siebel Server Log Files" on page 67.
- LFA, see "About the Log File Analyzer" on page 91.
- Event logging, see "About Configuring Siebel Server and Component Logging" on page 63.

For information on monitoring other Siebel Server run-time operations, see "Monitoring Siebel Server Status" on page 30.

To monitor Siebel Server log files on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Servers view.
2. In the Servers list, select the Siebel Server of interest.
3. From the view tabs, click Log.

   Each entry in the Log view list represents an event logged in the Siebel Server log file. For further details on each entry, click the record of interest and review information in the Info Detail view.

NOTE: The Server Manager GUI accesses Siebel Server log files from the log directory of each individual Siebel Server. Siebel Server log files use the following name convention: *EnterpriseServerName*.*SiebelServerName*.log.

Monitoring Siebel Server Statistics

Monitor Siebel Server statistics using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For background information and a list of Siebel Server statistics, see Appendix A, "List of Statistics and State Values." For information on monitoring other Siebel Server run-time operations, see "Monitoring Siebel Server Status" on page 30.

To monitor Siebel Server statistics on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Servers view.
2. In the Servers list, select the Siebel Server of interest.
3. From the view tabs, click Statistics.

   Statistics for the selected Siebel Server appear in the Statistics list. For a list and description of Siebel Server statistics, see Appendix A, "List of Statistics and State Values."

To monitor Siebel Server statistics on srvrmgr

- At the srvrmgr program prompt for a particular Siebel Server, enter:

```
list statistics for server siebel_server_name
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Tasks

Monitor Siebel Server component tasks for a particular Siebel Server by using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr).

For details on Siebel Server component task states, see "About Siebel Server Task States" on page 26. For information on monitoring other Siebel Server run-time operations, see "Monitoring Siebel Server Status" on page 30.

To monitor Siebel Server tasks on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Servers view.
2. In the Servers list, select the Siebel Server of interest.
3. From the view tabs, click Tasks.
4. Review the status of the tasks for the selected Siebel Server by viewing the State (Icon), State, and Status fields.

For more information on monitoring individual tasks, note the Task number and see "Monitoring Server Component Task Status" on page 37.

To monitor Siebel Server tasks on srvrmgr

- At the srvrmgr program prompt, enter:

```
list tasks for server siebel_server_name
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server User Sessions

Monitor user sessions for a particular Siebel Server by using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr).

For background information on user sessions, see "About User Sessions" on page 28. For information on monitoring other Siebel Server run-time operations, see "Monitoring Siebel Server Status" on page 30.

To monitor Siebel Server user sessions on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Servers view.

2 In the Servers list, select the Siebel Server of interest.

3 From the view tabs, click Sessions.

4 Review the status of the users' sessions for the selected Siebel Server by viewing the State (Icon), Task Hung State, and State fields.

For further details on monitoring individual user sessions, note the Session ID number and see "Monitoring User Session Status" on page 40.

To monitor Siebel Server user sessions on srvrmgr

- At the srvrmgr program prompt, enter:

  `list sessions for server siebel_server_name`

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Component Status

Monitor the status of Siebel Server components by using the Server Manager GUI or Server Manager command-line interface program (srvrmgr). The following topics describe procedures that monitor the Siebel Server components:

For background information on Siebel Server components, including running and configuring procedures, see *Siebel System Administration Guide*.

Monitoring Siebel Server Component State

Monitor the status of Siebel Server components using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr).

For details on Siebel Server component states, see "About Siebel Server Component States" on page 24. For information on monitoring other Siebel Server component run-time operations, see "Monitoring Siebel Server Component Status" on page 34.

To monitor the Siebel Server component state on Server Manager GUI

1 Navigate to the Administration – Server Management screen, then Components view.

2 In the Components list, select the Siebel Server component of interest.

3 Review the state of the selected Siebel Server component by viewing the State (Icon) and State fields.

The Components list view lists the Siebel Server components from all Siebel Servers operating in the Siebel Enterprise Server.

To monitor the component state on srvrmgr

- At the srvrmgr program prompt, enter:

```
list component
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Component State Values

Monitor Siebel Server component state values using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For background information and a list of Siebel Server state values, see Appendix A, "List of Statistics and State Values." For information on monitoring other Siebel Server component run-time operations, see "Monitoring Siebel Server Component Status" on page 34.

To monitor component state values on Server Manager GUI

1 Navigate to the Administration – Server Management screen, then Components view.

2 In the Components list, select the Siebel Server component of interest.

3 From the view tabs, click State Values.

State values for the selected Siebel Server component appear in the State Values list. For a list and description of Siebel Server state values, see Appendix A, "List of Statistics and State Values."

To monitor component state values on srvrmgr

- At the srvrmgr program prompt, enter:

```
list state values for component component_alias_name
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Component Statistics

Monitor Siebel Server component statistics using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For background information and a list of Siebel Server component statistics, see Appendix A, "List of Statistics and State Values." For information on monitoring other Siebel Server component run-time operations, see "Monitoring Siebel Server Component Status" on page 34.

To monitor component statistics on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Components view.
2. In the Components list, select the Siebel Server component of interest.
3. From the view tabs, click Statistics.

   Statistics for the selected Siebel Server component appear in the Statistics list. For a list and description of Siebel Server statistics, see Appendix A, "List of Statistics and State Values."

To monitor component statistics on srvrmgr

- At the srvrmgr program prompt, enter:

  `list statistics for component` *`component_alias_name`*

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Siebel Server Component Tasks

Monitor tasks for a particular Siebel Server component by using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For details on Siebel Server component task states, see "About Siebel Server Task States" on page 26. For information on monitoring other Siebel Server run-time operations, see "Monitoring Siebel Server Status" on page 30.

To monitor Siebel Server tasks on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Components view.
2. In the Components list, select the Siebel Server component of interest.
3. From the view tabs, click Tasks.
4. Review the status of tasks for the selected Siebel Server component by viewing the State (Icon), State, and Status fields.

For further details on monitoring individual tasks, note the Task number and see "Monitoring Server Component Task Status" on page 37.

To monitor component tasks on srvrmgr

- At the srvrmgr program prompt, enter:

```
list tasks for component component_alias_name
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Server Component Task Status

Monitor the status of Siebel Server component tasks by using the Server Manager GUI or Server Manager command-line interface program (srvrmgr). The following topics describe procedures that monitor Siebel Server component tasks:

A task, in the context of a Siebel application, is an instantiation of a Siebel Server component. Administrators start tasks by creating jobs. Tasks are also started by the Siebel application itself. For background information on Siebel Server component tasks, including running and configuring procedures, see *Siebel System Administration Guide*.

Monitoring Server Component Task State

Monitor the state of Siebel Server component tasks using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For details on Siebel Server component task states, see "About Siebel Server Task States" on page 26. For information on monitoring other task run-time operations, see "Monitoring Server Component Task Status" on page 37.

To monitor tasks on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Tasks view.
2. In the Tasks list, select the task of interest.
3. Review the state of the selected task by viewing the State (Icon), State, and Status fields.

The Tasks view lists tasks from all Siebel Servers operating in the Siebel Enterprise Server. To isolate tasks on a particular Siebel Server, see "Monitoring Siebel Server Tasks" on page 33. To isolate tasks for a particular Siebel Server component, see "Monitoring Siebel Server Component Tasks" on page 36.

NOTE: You cannot sort tasks from different Siebel Servers across the enterprise.

To monitor tasks on srvrmgr

- At the srvrmgr program prompt, enter:

```
list tasks
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

To list tasks that have exited in error

1. Make sure that the SvrTaskPersist component (which belongs to the SystemAux component group) is enabled.
2. Run an SQL statement to query tasks that exit in error for a specific table.

   For example, you might use the following query to return tasks with errors from the S_SRM_TASK_HIST table:

```
select CREATED, SRVR_PROC_ID_VAL, SRVR_LOGFILE_NAME, SRVR_STATUS
  from SIEBEL.S_SRM_TASK_HIST
  where SRVR_TASK_ID_VAL='123456789';
```

   All tasks that exited in error are returned by the SQL statement with the status set to ERROR.

Monitoring Server Component Task Log Files

Monitor the log files for a Siebel Server component task using the Server Manager GUI. Also review task log files by manually accessing the file or querying the file with the Log File Analyzer (LFA) utility.

For background information on:

- Event logging, see Chapter 4, “Configuring Siebel Server and Component Logging.”
- Task log files, see “Configuring Siebel Server Component Logging” on page 71.
- LFA, see Chapter 6, “Querying System Log Files.”

For information on monitoring other task run-time operations, see “Monitoring Server Component Task Status” on page 37.

To monitor task log files on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Tasks view.
2. In the Tasks list, select the task of interest.
3. From the view tabs, click Log.

   Each entry in the Log view list represents an event logged in the task log file.

Monitoring Server Component Task State Values

Monitor Siebel Server component task state values using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For background information and a list of task state values, see Appendix A, "List of Statistics and State Values." For information on monitoring other task run-time operations, see "Monitoring Server Component Task Status" on page 37.

To monitor task state values on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Tasks view.
2. In the Tasks list, select the task of interest.
3. From the view tabs, click State Values.

   State values for the selected task appear in the State Values list. For a list and description of task state values, see Appendix A, "List of Statistics and State Values."

To monitor task state values on srvrmgr

- At the srvrmgr program prompt, enter:

```
list state values for task task_number
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Server Component Task Statistics

Monitor Siebel Server component task statistics using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). For background information and a list of task statistics, see Appendix A, "List of Statistics and State Values." For information on monitoring other task run-time operations, see "Monitoring Server Component Task Status" on page 37.

To monitor task statistics on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Tasks view.
2. In the Tasks list, select the task of interest.
3. From the view tabs, click Statistics.

   Statistics for the selected task appear in the Statistic list. For a list and description of task statistics, see Appendix A, "List of Statistics and State Values."

To monitor task statistics on srvrmgr

- At the srvrmgr program prompt, enter:

```
list statistics for task task_number
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring Component Job Status

Monitor the status of Siebel Server component jobs using the Server Manager GUI. For background information on starting Siebel Server component jobs, see *Siebel System Administration Guide*. For information on component job states, see "About Component Job States" on page 27.

To monitor component job status

1. Navigate to the Administration - Server Management screen, then Jobs view.
2. In the Jobs list, select the component job of interest.
3. Review the status of the component job by viewing the Status field.

To monitor component job status requested by your User ID

1. Navigate to the Jobs screen.
2. In the My Jobs list, select the component job of interest.
3. Review the status of the component job by viewing the status field.

Monitoring User Session Status

Monitor the status of user sessions by using the Server Manager GUI or Server Manager command-line interface program (srvrmgr). The following topics describe procedures that monitor user sessions:

- "Monitoring User Session State" on page 40
- "Monitoring User Session Log Files" on page 42
- "Monitoring User Session State Values" on page 42
- "Monitoring User Session Statistics" on page 43

For background information on user sessions, see "About User Sessions" on page 28.

Monitoring User Session State

Monitor the state of Siebel Server user sessions using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). The state of the user session is that of the associated Siebel Server component task that represents the user session.

For background information on user sessions, see "About User Sessions" on page 28. For background information on Siebel Server component task states, see "About Siebel Server Task States" on page 26. For information on monitoring other Siebel Server user session run-time operations, see "Monitoring User Session Status" on page 40.

To monitor user sessions on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Sessions view.
2. In the Sessions list, select the Siebel Server user session of interest.
3. Review the state of the selected Siebel Server user session by viewing the State (Icon), Task Hung State, and State fields.

The Sessions view lists Siebel Server user sessions from all Siebel Servers operating in the Siebel Enterprise Server. To isolate sessions on a particular Siebel Server, see "Monitoring Siebel Server Tasks" on page 33.

To monitor user sessions for a Siebel Server using srvrmgr

- At the srvrmgr program prompt, enter:

```
list sessions for server siebel_server_name
```

To monitor user sessions for a Siebel Server component using srvrmgr

- At the srvrmgr program prompt, enter:

```
list sessions for comp component_alias_name
```

To monitor user sessions for a Siebel Application Object Manager using srvrmgr

- At the srvrmgr program prompt, enter:

```
list sessions for login object_manager_login
```

To list user sessions that are not responding using srvrmgr

- At the srvrmgr program prompt, enter:

```
list hung sessions for server siebel_server_name [or]comp component_alias_name
[or]login object_manager_login
```

To list active user sessions using srvrmgr

- At the srvrmgr program prompt, enter:

```
list active sessions for server siebel_server_name [or]comp component_alias_name
[or]login object_manager_login
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Monitoring User Session Log Files

Monitor the log files for Siebel Server user sessions using the Server Manager GUI. User session log files are those of the associated Siebel Server component task that represents the user session. Also review Siebel Server user session log files by accessing the associated task log file or querying the associated task log file with the Log File Analyzer utility.

For background information on:

- User sessions, see "About User Sessions" on page 28.
- Siebel Server component task log files, see "Configuring Siebel Server Component Logging" on page 71.
- Log File Analyzer, see Chapter 6, "Querying System Log Files."
- Event logging, see Chapter 4, "Configuring Siebel Server and Component Logging."

For information on monitoring other Siebel Server user session run-time operations, see "Monitoring User Session Status" on page 40.

To monitor user session log files on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Sessions view.
2. In the Sessions list, select the Siebel Server user session of interest.
3. From the view tabs, click Log.

   Each entry in the Log view represents an event logged in the Siebel Server component task log file, which represents the user session.

Monitoring User Session State Values

Monitor Siebel Server user session state values using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). User session state values are those of the associated Siebel Server component task that represents the user session.

For background information on user sessions, see "About User Sessions" on page 28. For background information and a list of task state values, see Appendix A, "List of Statistics and State Values." For information on monitoring other Siebel Server user session run-time operations, see "Monitoring User Session Status" on page 40.

To monitor user session state values on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Sessions view.
2. In the Sessions list, select the Siebel Server user session of interest.
3. From the view tabs, click State Values.

   State values for the selected task that represent the user session appear in the State Values list. For a list and description of task state values, see Appendix A, "List of Statistics and State Values."

To monitor user session state values on srvrmgr

- Use the srvrmgr command to list task state values as described in "To monitor task state values on srvrmgr" on page 39. Use the Session ID for the task number parameter in this command.

Monitoring User Session Statistics

Monitor Siebel Server user session statistics using the Server Manager GUI or the Server Manager command-line interface program (srvrmgr). User session statistics are those of the associated Siebel Server component task that represents the user session.

For background information on user sessions, see "About User Sessions" on page 28. For background information and a list of task statistics, see Appendix A, "List of Statistics and State Values." For information on monitoring other Siebel Server user session run-time operations, see "Monitoring User Session Status" on page 40.

To monitor user session statistics on Server Manager GUI

1. Navigate to the Administration – Server Management screen, then Sessions view.
2. In the Sessions list, select the Siebel Server user session of interest.
3. From the view tabs, click Statistics.

   State values for the selected task that represent the user session appear in the State Values list. For a list and description of task state values, see Appendix A, "List of Statistics and State Values."

To monitor user session statistics on srvrmgr

- Use the srvrmgr command to list task statistics as described in "Monitoring Server Component Task State Values" on page 39. Use the Session ID for the task number parameter in this command.

About Monitoring Application Server Operations Across an Enterprise

As part of the Siebel Management Framework, the Siebel Management Server and Management Agents allow you to monitor one or more application servers across a Siebel enterprise. Siebel Management Agents are enabled with JMX (Java Management Extension) technology and are implemented as Managed Bean (MBean) servers with multiple MBeans running. Using the JMX API (Java Management Extensions application program interface), you can write Java code that performs monitoring operations specific to your deployment.

NOTE: This across-enterprise functionality is only available if the Siebel Servers and the Siebel Management Server and Management Agents are up and running.

For more information about the Siebel Management Server and Management Agents, see *Siebel Installation Guide* for the operating system you are using. For more information about JMX, see *Siebel Application Deployment Manager Guide*.

Table 16 lists the application server monitoring operations you can perform at the enterprise, server, and component level.

Table 16. Application Server Monitoring Operations

Operation	Enterprise	Server	Component
Retrieve a list of application servers in an enterprise	Yes	No	No
Retrieve an enterprise-level value	Yes	No	No
Check component definitions and availability across an enterprise	Yes	No	No
Retrieve the current state of a Siebel Server	No	Yes	No
Retrieve a list of components and their states on a Siebel Server	No	Yes	No
Retrieve server-level runtime attributes (such as parameters, state values, and statistics) for a Siebel Server	No	Yes	No
Get generic attributes (such as parameters, state values, and statistics) of a component running on a Siebel Server	No	No	Yes

For information about monitoring application server operations at the enterprise, server, and component levels, see "Process of Monitoring Siebel Application Server Operations Across an Enterprise" on page 44.

Process of Monitoring Siebel Application Server Operations Across an Enterprise

Using JMX (Java Management Extension) APIs, you can monitor the current health of your enterprise or a particular server or component.

To monitor application server operations across an enterprise, perform the following tasks:

- "Preparing to Monitor Siebel Application Server Operations Across an Enterprise" on page 45
- "Monitoring Enterprise Operations for Siebel Application Servers Across an Enterprise" on page 47
- "Monitoring Server Operations for Siebel Application Servers Across an Enterprise" on page 48
- "Monitoring Component Operations for Application Servers Across an Enterprise" on page 49

NOTE: You must perform the prerequisite tasks before performing these monitoring tasks.

For more information about monitoring application server operations across an enterprise, see "About Monitoring Application Server Operations Across an Enterprise" on page 43.

Preparing to Monitor Siebel Application Server Operations Across an Enterprise

Before beginning your application server monitoring operations across an enterprise, perform the following procedure.

This task is a step in "Process of Monitoring Siebel Application Server Operations Across an Enterprise" on page 44.

To prepare for monitoring application server operations across an enterprise

1 Make sure:

- The appropriate JDK (Java Development Kit) is installed and the installation path is set in the user environment variables.
- One or more Siebel Servers are running
- The Siebel Management Server and a Management Agent or Agents are running

For more information about what version of JDK to use, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network. For information about setting user environment variables and Siebel Server, see *Siebel System Administration Guide*. For more information about the Siebel Management Server and Management Agents, see *Siebel Installation Guide* for the operating system you are using and *Siebel Application Deployment Manager Guide*.

2 Create a working Java directory from which you can monitor operations.

For example, you might create a source folder in the D directory (`D:\src`).

NOTE: It is recommended that you create this directory in the same location where JDK is installed.

3 Copy the siebelmgr.jar file from `<management agent root directory>\lib` to `D:\src`.

This file provides the communication between the server and management agent as well as the notification of messages from the server to the agent.

For example, you might copy from `D:\SP1_08_CORE\ses\siebsrvr\mgmtagent\lib`.

4 Copy the siebeljmxapi.jar file from *`management server root dir`*`\lib` to `D:\src`.

This file connects the server to the management agent and converts the various queries received into something the agent can understand.

For example, you might copy from `D:\DiagTool\lib`.

5 Create the following files using the code in Appendix B, "Sample Files for Monitoring Siebel Application Servers," and save the files to the Java directory you created in Step 2:

- key1.txt
- 1srvr.xml
- 2srvr.xml
- Enterprise.java
- Server.java
- Component.java

6 Edit the appropriate sample XML file (1srvr.xml or 2srvr.xml) by performing the following:

a Generate an encrypted password to include in your XML file by executing the following command:

```
java -cp <dir1>\siebelmgr.jar com.siebel.management.util.Decoder
<dir2> <key_file> <clear_text_password>
```

where:

- *dir1* is the directory where the siebelmgr.jar is located.
- *dir2* is the directory where the security.properties file is located.
- *key_file* is a file containing a continuous string of characters that represents your chosen key, and that character string is used as the encryption key. For example, the key_file might contain the following string:

Mississippi123

- *clear_text_password* is the encrypted password.

For example, you might execute the following command:

```
D:\src>java -cp D:\src\siebelmgr.jar com.siebel.management.util.Decoder
D:\DiagTool\security\ D:\src\key1.txt MyPassword)
```

As a result, the following output is generated:

XQj12uTtvi8 (the encrypted base64 encoded password for MyPassword)

b Make sure the enterprise name, Siebel Server name, user name, connect string (URL), and password are correct for your specific deployment.

For more information about which XML file to use and how to use it, Appendix B, "Sample Files for Monitoring Siebel Application Servers."

7 Create the following files using the code in Appendix B, "Sample Files for Monitoring Siebel Application Servers," and save the files to the Java directory you created in Step 2:

- Enterprise.java
- Server.java
- Component.java

8 Edit the appropriate sample Java file (Enterprise.java, Server.java, or Component.java), making sure the enterprise and component names are correct for your specific deployment.

You are now ready to begin monitoring application server operations at the enterprise, server, or component level. For more information, see:

- "Monitoring Enterprise Operations for Siebel Application Servers Across an Enterprise" on page 47
- "Monitoring Server Operations for Siebel Application Servers Across an Enterprise" on page 48
- "Monitoring Component Operations for Application Servers Across an Enterprise" on page 49

Monitoring Enterprise Operations for Siebel Application Servers Across an Enterprise

When you want to know the current health of your enterprise or a particular server or component. Perform the following procedure to monitor enterprise operations for application servers across an enterprise.

This task is a step in "Process of Monitoring Siebel Application Server Operations Across an Enterprise" on page 44.

To monitor enterprise operations for application servers across an enterprise

1 Edit the Enterprise.java file to change String ent = "*enterprise_name*" where enterprise_name is the name of your Siebel enterprise.

2 From the Start menu, select Run, and then navigate to the working Java directory you created in Step 2 on page 45 (`D:\src`).

3 Set the class path by executing the following command:

```
set CLASSPATH=<Java_directory>siebelmgr.jar;<Java_directory>\siebeljmxapi.jar;
```

4 Compile Enterprise.java by executing the following command:

```
javac Enterprise.java
```

5 Display the enterprise operations to the console by executing one of the following commands:

- If you are monitoring a single server, enter:

```
java -Dcom.siebel.management.jmxapi.cfgFileName=<Java_dir>\1server.xml
Enterprise
```

- If you are monitoring two or more servers, enter:

```
java -Dcom.siebel.management.jmxapi.cfgFileName=<Java_dir>\2srvr.xml Enterprise
```

where:

Java_dir is the directory you created in Step 2 on page 45.

The command console displays the enterprise operations for the application server or servers.

Monitoring Server Operations for Siebel Application Servers Across an Enterprise

Perform the following procedure to monitor server operations for application servers across an enterprise.

This task is a step in "Process of Monitoring Siebel Application Server Operations Across an Enterprise" on page 44.

To monitor server operations for application servers across an enterprise

1. Edit the Server.java file to change the following settings:
   - String ent = "*enterprise_name*"
   - String srv = "*server_name*"

   where:
   - *enterprise_name* is the name of your Siebel enterprise
   - *server_name* is the name of the application server that you want to monitor
2. From the Start menu, select Run, and then navigate to the working Java directory that you created in Step 2 on page 45.
3. Set the class path by executing the following command:

   ```
   set CLASSPATH=<Java_dir>siebelmgr.jar;<Java_dir>\siebeljmxapi.jar;
   ```

   where:

   Java_dir is the directory you created in Step 2 on page 45.
4. Compile Enterprise.java by executing the following command:

   ```
   javac Enterprise.java
   ```
5. Display the server operations to the console by executing the following command:

   ```
   java -Dcom.siebel.management.jmxapi.cfgFileName=<Java_dir>\1srvr.xml Server
   ```

   where:

   Java_dir is the directory you created in Step 2 on page 45.

   The command console displays the server operations for the application server.

Monitoring Component Operations for Application Servers Across an Enterprise

Perform the following procedure to monitor component operations for application servers across an enterprise.

This task is a step in "Process of Monitoring Siebel Application Server Operations Across an Enterprise" on page 44.

To monitor component operations for application servers across an enterprise

1. Edit the Component.java file to change the following settings:
   - String ent = "*enterprise_name*"
   - String srv = "*server_name*"
   - String comp = "*comp_name*"

   where:
   - *enterprise_name* is the name of your Siebel enterprise
   - *server_name* is the name of the application server that you want to monitor
   - *comp_name* is the name of the component that you want to monitor
2. From the Start menu, select Run, and then navigate to the working Java directory you created in Step 2 on page 45.
3. Set the class path by executing the following command:

   ```
   set CLASSPATH=<Java_dir>siebelmgr.jar;<Java_dir>\siebeljmxapi.jar;
   ```

   where:

   Java_dir is the directory you created in Step 2 on page 45.
4. Compile Enterprise.java by executing the following command:

   ```
   javac Enterprise.java
   ```
5. Display the component operations to the console by executing the following command:

   ```
   java -Dcom.siebel.management.jmxapi.cfgFileName=<Java_dir>\1srvr.xml Component
   ```

   where:

   Java_dir is the directory you created in Step 2 on page 45.

   The command console displays the component operations for the designated component on the designated application server.

Analyzing System Data with Siebel Run-Time Data

Analyze operating system data with Siebel run-time data using the following procedures.

Identifying Task Log Files From the Siebel Server Log File

Map the Siebel Server log file to its Siebel Server components and their log files by identifying the task ID in the Siebel Server log file. Review the task log file for further information on the task performance.

NOTE: The detail of the log file depends on logging levels set for event types for each component. For details on event types and event logging, see Chapter 4, "Configuring Siebel Server and Component Logging."

For information on analyzing other Siebel application diagnostic data, see "Analyzing System Data with Siebel Run-Time Data" on page 50.

To identify task IDs from Siebel Server log files

1. Access a Siebel Server log file by using the Server Manager GUI. For details on this procedure, see "Monitoring Siebel Server Log Files" on page 32.

   Also access Siebel Server log files by:

   - Using the Log File Analyzer. For details on this procedure, see Chapter 6, "Querying System Log Files."
   - Opening the log file itself. For details on locations and naming convention of Siebel Server log files, see "About Siebel Server Log Files" on page 67.

2. Review the Text field of each log file entry for the Siebel Server component of interest.
3. The text field of each Siebel Server component log file entry also contains the task ID number started for this component.
4. Access the Siebel Server component task list. For details on this procedure, see "Monitoring Server Component Task State" on page 37.
5. Query the list with the task ID number identified in the Siebel Server log file.

6 Review the status of the Siebel Server component task by reviewing the log file, state value, and statistics for this task. For details on these procedure, see "Monitoring Server Component Task Status" on page 37.

NOTE: The task ID number identified in step 3 can also be used to find the individual task log file stored in the log folder. The name of the task log file contains the task ID for the component. For example, in `SCCObjMgr_enu_19369.log`, the task ID is `19369`.

Process of Mapping Tasks with Operating System Data

Mapping tasks to operating system data allows you to view operating system CPU and memory usage for each task. Once you map a task to an operating system process ID, you can use operating system tools, such as task manager on Windows or the ps (process list) function on UNIX systems, to view other information about the process and task including CPU utilization, memory usage, and so on.

NOTE: Multithreaded components can have several tasks mapped to a single operating system process ID, so that the operating system tools do not necessarily break down the data by task.

Map the Siebel Server component task to the operating system data by:

1 Identifying the operating system process ID (PID) for a task. For this procedure, see "Identifying Operating System PID for a Task" on page 51.

2 Reviewing the PID in the operating system. For this procedure, see "Identifying Operating System PID for a Task" on page 51.

For information on analyzing other Siebel application diagnostic data, see "Analyzing System Data with Siebel Run-Time Data" on page 50.

Identifying Operating System PID for a Task

Identifying operating system PID numbers is a task in the "Process of Mapping Tasks with Operating System Data." Identify operating system process ID numbers (PID) for tasks by one of the following methods:

- From the Server Manager GUI
- From the Siebel Server log file
- From the Task log file

NOTE: PIDs are only available in the Server Manager for running tasks.

To identify operating system PID for a task from the Server Manager GUI

1 Access the Siebel Server component task list. For details on this procedure, see "Monitoring Server Component Task State" on page 37.

2 Query the task list for a specific Siebel Server component task or task ID.

3 Note the value in the PID field for that particular task.

To identify operating system PID for a task from a Siebel Server log file

1 Access a Siebel Server log file by using the Server Manager GUI. For details on this procedure, see "Monitoring Siebel Server Log Files" on page 32.

   Also access Siebel Server log files by:

   - Using the Log File Analyzer. For details on this procedure, see Chapter 6, "Querying System Log Files."
   - Opening the log file itself. For details on locations and naming convention of Siebel Server log files, see "About Siebel Server Log Files" on page 67.

2 Review the Text field of each log file entry for the Siebel Server component of interest.

3 The Text field of each Siebel Server component log file entry also contains the process ID number started for this component task.

To identify operating system PID for a task from a task log file

1 Access the Siebel Server component task log file of interest. For details on locations and naming convention of Siebel Server component task log files, see Chapter 4, "Configuring Siebel Server and Component Logging."

2 The first entry of the task log file contains the header information. The header information contains the PID number. For a parsing of the header file and to identify the PID number, see "About Event Attributes and Log File Format" on page 65.

Reviewing the PID in the Operating System

Reviewing the process ID number in the operating systems allows the identification of CPU and memory usage for individual tasks. To identify the PID number for a task, see "Identifying Operating System PID for a Task."

Reviewing the PID numbers in the operating system is a task in the "Process of Mapping Tasks with Operating System Data."

To review PID numbers under Microsoft Windows

1 Using the right mouse button, click a blank area on the taskbar.

2 Choose Task Manager.

   The Windows Task Manager dialog box appears.

3 Select the Processes tab and query for the task PID number.

   NOTE: If the PID column is not visible, click View, then Select Columns.

To review PID numbers under UNIX

- Enter the command:

```
ps -ef | grep <PID>
```

or:

```
ps -aux <PID>
```

where:

PID is the number of interest.

Mapping User Sessions to Siebel Servers or AOMs

Map user sessions from the Web server to individual Siebel Servers or Siebel Application Object Managers (AOMs) by accessing the user session cookie in the Siebel Web Server Extension (SWSE) log file. For information on analyzing other Siebel application diagnostic data, see "Analyzing System Data with Siebel Run-Time Data" on page 50.

To map user session to a Siebel Server

1. Access the SWSE log file. For details on locations and naming convention of Web server SWSE file, see Chapter 4, "Configuring Siebel Server and Component Logging."
2. Identify the Server ID number in the user session cookie entry for the SWSE log file. For details on reviewing SWSE cookies, see "Parsing a SWSE Log File Cookie" on page 12.
3. Start the Server Manager command-line interface program (srvrmgr) at the enterprise level.

   For information on starting and running srvrmgr, see *Siebel System Administration Guide*.
4. Enter the following command:

   ```
   list servers show SBLSRVR_NAME, SV_SRVRID
   ```

To map user session to a Siebel Application Object Manager (AOM) task

1. Access the SWSE log file. For details on locations and naming convention of Web server SWSE files, see Chapter 4, "Configuring Siebel Server and Component Logging."
2. Identify the operating system ID number (PID) in the user session cookie entry for the SWSE log file. For details on reviewing SWSE cookies, see "Parsing a SWSE Log File Cookie" on page 12.
3. Access the Siebel Server component task list. For details on this procedure, see "Monitoring Server Component Task State" on page 37.
4. Query the task list for the specific PID to isolate the AOM task for that user session.
5. Review data on that AOM task.

For details on these procedures, see "Monitoring Server Component Task Status" on page 37.

About Using SQL Tagging to Trace Long-Running Queries in Siebel Business Applications

The SQL tagging feature in Siebel Business Applications provides administrators with the ability to trace the origin of long-running or slow-performing SQL statements (queries) back to a specific task and user who triggered it. This topic describes the SQL tagging feature and syntax. It also provides sample code.

After SQL tagging is enabled, tagging information is added to the Siebel Application Object Manager (AOM)-generated SELECT statements. When a poor performing SQL statement is suspected, the database administrator can use this tagging information at the database level to trace which component task and user initiated the SQL without the need to reboot any Siebel Server or components. Database administrators can then find the component task log file for more in-depth analysis of the performance issue. For information about enabling SQL tagging, see "Enabling and Disabling SQL Tagging" on page 55.

NOTE: Other SQL statements generated by Siebel Application Object Manager, such as INSERT, UPDATE, and DELETE are not tagged. The SQL tagging feature is available for Siebel CRM version 8.1.1.1 and later, and Siebel CRM version 8.2.

SQL Tagging Format

SQL tagging information is formatted as a comma-separated list of values using the following syntax:

```
<componentname>,<servername>,<taskid>,<userid>,<flowid:sarmid>,<busobjname>,
<buscompname>,<viewname>
```

where:

- *componentname* is the alias of the component, for example, SCCObjMgr_enu.
- *servername* is the name of the Siebel Server on which the component or task is running.
- *taskid* is the task ID of the user who generated the query.
- *userid* is the login name of the user who generated the query.
- *flowid* is the flow ID of the component or task.
- *sarmid* is the SARM ID of the component or task.
- *busobjname* is the business object name.
- *buscompname* is the business component name.
- *viewname* is the view name (only in UI mode).

NOTE: Any optional elements of a tag that are irrelevant or missing for the query are replaced with an empty string.

Sample SQL-Tagged Code

The following is a sample of how a tagged SQL statement might appear in a log file when the SQL statement was generated from a Siebel Call Center (SCCObjMgr_enu AOM) component and an Oracle database. The changes made by the SQL tagging feature appear in italics.

```
SELECT

   FIRST_NAME,
   LAST_NAME,
    …,
   :1

FROM

   TBO.S_CONTACT
   …

WHERE

   LAST_NAME LIKE:2

ORDER BY

…
```

Bind variable 1:
SCCObjMgr_enu,sdchs20i046,10485776,SADMIN,00000089489108a8:50557,Account,Account,
Account List View
Bind variable 2: Foo*

For information about enabling SQL tagging, see "Enabling and Disabling SQL Tagging" on page 55. For information about setting log levels for SQL tagging, see "Setting Log Levels for SQL Tagging" on page 57.

Enabling and Disabling SQL Tagging

The SQL tagging feature in Siebel Business Applications is a diagnostic tool that allows administrators to trace long-running or slow-performing queries back to the user or action that triggered it. For more information about SQL tagging, see "About Using SQL Tagging to Trace Long-Running Queries in Siebel Business Applications" on page 54.

By default, SQL tagging is disabled. Administrators can enable SQL tagging by setting the OM SQL Tagging (alias is ObjMgrSqlTag) server component event for a Siebel Application Object Manager (AOM) server component, such as Call Center Object Manager (ENU). The OM SQL Tagging event is available to all object manager-based components, such as AppObjMgr, EAIObjMgr, BusSvcMgr, and so on. You can enable and disable SQL tagging at any time if the AOM server component is running.

Enabling SQL Tagging

Use the following procedure to enable SQL tagging.

To enable SQL tagging

1. Navigate to the Administration - Server Configuration screen, then the Components view.
2. In the Components list, select the appropriate Siebel Application Object Manager for the application that you want to enable SQL tagging.

   For example, if the application you are using is Siebel Call Center, select Call Center Object Manager (ENU).
3. Click the Events subview, and then select the OM SQL Tagging event type.
4. Set the log level to a number greater than zero.

   For more information on setting the log levels, see "Setting Log Levels for SQL Tagging" on page 57.

Disabling SQL Tagging

Use the following procedure to disable SQL tagging.

To disable SQL tagging

1. Navigate to the Administration - Server Configuration screen, Components view.
2. In the Components list, select a Siebel Application Object Manager, for example, Call Center Object Manager (ENU).
3. Click the Events subview, and then select the OM SQL Tagging event type.
4. Set the log level to zero.

About Setting Log Levels for SQL Tagging

The Object Manager SQL Log (alias ObjMgrSqlLog) server component event type has an SqlTag event subtype. The SqlTag subtype sets the log level for an SQL tag at a lower lever than where SQL statements are typically logged. It enables logging of the SQL tags in the log file without logging the complete SQL statements.

The OM SQL Logging (ObjMgrSqlLog) and OM SQL Tagging (ObjMgrSqlTag) events are independent of each other and can coexist. The ObjMgrSqlLog event controls the level of SQL logging detail in an object manager log file. The ObjMgrSqlTag event controls whether SQL statements are tagged or not and how much tagging information is generated, irrespective of whether or not the SQL statements are logged in the log file. For example, if the ObjMgrSqlLog event log level is set to 1, neither the SQL statements nor the SQL tags are logged in the log files, even if the ObjMgrSqlTag subevent is active. Whereas, if the ObjMgrSqlLog event log level is set to 4, full details about the SQL information is generated in the log files. However, if the ObjMgrSqlTag subevent is not active, the SQL is not tagged, nor are the SQL tags logged in the log files.

If SQL tagging is active and the SQL Logging event is set at the SqlTag level, only the SQL tags are logged in the log files as shown in the following example:

```
Begin: Execute SqlObj 'Account' at 11160f10 with
SqlTag=SCCObjMgr_enu,sdchs20i046,10485776,,SADMIN,00000089489108a8:50557,
Account,Account,Account List V
```

This configuration is useful in diagnosing long-running SQLs without generating too much SQL logging in the log files.

NOTE: The END statement is logged so that you can run a script to identify log files where the BEGIN SQL tag statement is present, but there is no END statement. This syntax helps determine which log files might contain long-running SQL statements.

For information about setting log levels for SQL tagging, see “Setting Log Levels for SQL Tagging” on page 57.

Setting Log Levels for SQL Tagging

The OM SQL Tagging (ObjMgrSqlTag) server component event controls whether SQL tagging is enabled and how much tagging information is generated. Event log levels (event subtypes) control the level of detail that is added to each tag. The higher the log level, the higher the level of detail. For more information about setting SQL tagging log levels, see “About Setting Log Levels for SQL Tagging” on page 56.

To set the log level for SQL tagging

1. Navigate to the Administration - Server Configuration screen, then the Components view.
2. In the Components list, select the appropriate Siebel Application Object Manager for the application that you want to enable SQL tagging.

   For example, if the application you are using is Siebel Call Center, select Call Center Object Manager (ENU).
3. Click the Events subview, and then select the OM SQL Tagging event type.

4 Change the log level to a number using the following guidelines.

Log Level	Description
0	SQL tagging is disabled.
1	SQL tagging is disabled.
2	SQL tagging includes only the component name, server name, and task ID. **NOTE:** Use this log level when you do not want user IDs exposed in an OM SQL Tagging event log.
3	SQL tagging includes the component name, server name, task ID, user ID, and flow ID with the SARM ID.
4	SQL tagging includes the component name, server name, task ID, user ID, and flow ID with the SARM ID, business object, business component, and view name.

About Siebel Process Failure Diagnostics

Siebel Business Applications provide process failure diagnostics that help you to allocate system resources and maintain an efficient and reliable environment. Administrators can monitor KPIs (key performance indicators) of the various levels of the Siebel environment to do the following:

- Identify, isolate, and remedy adverse system and application conditions.
- Adjust the various settings and parameters of the Siebel environment to avert adverse conditions.
- Optimize available resources for improved performance.

When a Siebel process fails, the administrator can do the following:

- View all detectable failed processes for a given Siebel Enterprise by Siebel Server.
- Identify the probable cause and point of origin of the failure, including (but not limited to) user, process, thread, task, application, view, and activity immediately prior to the failure.
- Review content of the main process failure file (crash.txt) associated with a given failure including the call stack, the register contents, and the memory information.
- Review content of other failure information contained in the Siebel FDR (Flight Data Recorder), Siebel ARM (Application Response Measurement), and component log files.
- Determine other Siebel users directly affected by the failure.

Related Topics

How Siebel Process Failure Diagnostics Work on page 59

Scenario for Working with Siebel Process Failure Diagnostics on page 59

Investigating Failed Siebel Server Processes on page 60

Example of Investigating a Failed Siebel Server Process on page 61

How Siebel Process Failure Diagnostics Work

Siebel process failure diagnostics collects data for use by administrators to diagnose and troubleshoot a variety of failed Siebel Server processes.

The SvrTaskPersist component of the SystemAux component group handles the diagnostic data collection. This component uses the SIEBEL_DIAG_STORE environment variable as a location store to retrieve the diagnostic data (FDR file, crash.txt file, component log file, failure summary, and so on).

The administrator can use the siebprocdiag command-line utility to output the process failure data to any path you specify for future retrieval. The utility scans the *SIEBEL_ROOT*/bin directory and collects the various files for each process failure and copies them to the specified directory. For example, if you execute the following command for Windows, then the process failure files are copied to d:\temp:

```
siebprocdiag d:\temp
```

For information about configuring system environment variables and using server management utilities, see *Siebel System Administration Guide*.

Related Topics

About Siebel Process Failure Diagnostics on page 58

Scenario for Working with Siebel Process Failure Diagnostics on page 59

Investigating Failed Siebel Server Processes on page 60

Example of Investigating a Failed Siebel Server Process on page 61

Scenario for Working with Siebel Process Failure Diagnostics

This topic provides a scenario for working with Siebel Process Failure Diagnostics. You might use this feature differently, depending on your business needs.

An administrator is informed by a user that the Siebel application is unresponsive. The administrator navigates to the Process Failure Diagnostics view to investigate whether the cause might be a failed process. After identifying a failed process that is associated with that user, the administrator can view the details of the failure including:

- Siebel Server and server component names
- Time of failure
- Process, thread, and task IDs
- Number of affected tasks
- Last set of meaningful business processes that occurred at the time of the failure
- Whether a new process was created, and if so, the list of current users impacted by the failure

- A list of users whose sessions were lost following the failure
- Content of the actual crash.txt file that was logged, which provides the call stack, register contents, and memory information

The administrator then determines the cause and effect to address the issue or investigates further to resolve. When applicable, the administrator forwards the details to an internal technical support team to assist them in their troubleshooting activities. The administrator might also want to query the failed process data in the Siebel database to generate histograms or reports to initiate preventative measures.

Related Topics

About Siebel Process Failure Diagnostics on page 58

How Siebel Process Failure Diagnostics Work on page 59

Investigating Failed Siebel Server Processes on page 60

Example of Investigating a Failed Siebel Server Process on page 61

Investigating Failed Siebel Server Processes

The process failure feature in Siebel Business Applications provides administrators with the ability to analyze and diagnose various system failures. Administrators can identify the following:

- Siebel Server and component process or processes that have failed
- Users who have lost sessions as a result of the process failure
- User actions that might have resulted in the failure
- Dump files associated with the failed process

To investigate a failed Siebel Server process

1. Navigate to the Administration - Server Management screen, Diagnostics, and then the Process Failure Diagnostics view.
2. In the Failed Processes list, select a record to view the Siebel Server, server component, failure time, and other details about the process that failed.
3. In Affected Users on Failed Process, view the users who have lost the sessions as result of the process failure.
4. In Failed User Task, view the user actions that might have resulted in the failure.
5. In Failed Process Call Stack, view the content of the crash.txt file, which is the call stack information.

If you need further assistance with troubleshooting, create a service request (SR) on My Oracle Support. Alternatively, you can phone Oracle Global Customer Support directly to create a service request or get a status update on your current SR. Support phone numbers are listed on My Oracle Support.

Related Topics

About Siebel Process Failure Diagnostics on page 58

How Siebel Process Failure Diagnostics Work on page 59

Scenario for Working with Siebel Process Failure Diagnostics on page 59

Example of Investigating a Failed Siebel Server Process on page 61

Example of Investigating a Failed Siebel Server Process

This topic gives one example of how you might investigate a failed Siebel Server process. You might use this feature differently, depending on your business needs. When a Siebel Server process failure is suspected, administrators can diagnose the nature of the failure in detail.

To investigate a failed Siebel Server process

1. Navigate to the Administration - Server Management screen, then the Process Failure Diagnostics view.
2. In the Failed Process list, select the failed process for which you want to learn more, then review the details about that failure. Details include: the Siebel Server and server component on which the process failed; the time of failure; process, thread, and task IDs; the location of related failure dump files, and so on.
3. Identify all users associated with this task, and notify these users as appropriate, that a key process related to their current activity has failed. Both the administrator and users can then take appropriate action, such as restarting any operations that have halted because of the failed process.
4. Review the activity and events (user-initiated and otherwise) that occurred immediately prior to the process failure.

   Administrators can use this information to detect patterns in Siebel configurations, usage, and interdependencies of Siebel entities (components, attribute and parameter settings, hardware, and so on) that might lead to process failures.
5. Query the failed process data in the Siebel database to generate histograms and reports. If the data indicates that failures are related to resources or throughput, then administrators can use that data for potential preventative measures.
6. Forward details from failure-related files and detectable patterns to technical support to assist in their troubleshooting of code and configuration issues.

Related Topics

About Siebel Process Failure Diagnostics on page 58

How Siebel Process Failure Diagnostics Work on page 59

Scenario for Working with Siebel Process Failure Diagnostics on page 59

Investigating Failed Siebel Server Processes on page 60

4 Configuring Siebel Server and Component Logging

This chapter provides descriptions and examples of configuring Siebel Server and component logging using Siebel events. It includes the following topics:

About Configuring Siebel Server and Component Logging

Configuring Siebel Server and component logging captures the internal activity and behavior of Siebel Business Applications during operation. Siebel Server and component logging use event logging to collect data and write the information to a text log file. You can configure event logging to use system alerts, or you can use event logging with third-party system management applications to notify administrators of any significant or adverse conditions. For information about configuring server components to use system alerts, see *Siebel System Administration Guide*. You can monitor and manage most Siebel Business Applications products and functional areas with event logging.

The information collected by event logging can range from error messages to detailed diagnostic logs. Some of the application conditions and operations that result in data written to the log file include:

- Catastrophic or error conditions
- Change of status of a Siebel Server or server component
- Start or finish of a Siebel process or workflow
- Specific point in a Siebel process or workflow
- When measurable threshold values are reached or exceeded
- When operational conditions are met

About Events and Event Logging

The elements of event logging are defined in the following bullets:

- **Event.** An event is created each time you execute a program code (such as running a task).
- **Event Type.** Event types are categories of events.
  - For information on event types pertinent to a specific part of Siebel Business Applications, see product-specific documentation or details available on My Oracle Support.

    NOTE: The Siebel Bookshelf is available on Oracle Technology Network (OTN) and Oracle Software Delivery Cloud. It might also be installed locally on your intranet or on a network location.
  - For generic event types used in server component and Siebel Application Object Manager diagnostics, see "Common Event Types for Component Diagnostics" on page 81 and "Common Event Types for Siebel Application Object Manager Diagnostics" on page 82.
- **Event Subtype.** Event subtypes are code references that define the event.
- **Log Level.** The log level determines the amount of information that is written to the log file. Log levels are set for event types. Table 17 lists the log levels of event types.
- **Severity.** A severity level is associated with each event subtype. The severity level and log level share the same scale and are compared when writing events to the log file. Table 17 lists the severity of event subtypes.

Table 17. Severity and Log Levels

Log and Severity Level	Description
0	Fatal
1	Errors
2	Warnings
3	Informational
4	Details
5	Diagnostic

When an event occurs, the severity level of the event (as defined by the event subtype) is compared with the log level of the event type. If the numerical value of the event severity level is equal to or lower than the numerical value of the event type log level, then the event is written to the log file. If the numerical value of the event severity level is higher than the numerical value of the event type log level, then the event is ignored.

NOTE: Event subtypes with a lower numeric value have a higher severity. For example, a value of 0 indicates the event subtype is more severe than one with a value of 5. By setting the event log level to a low number such as 1, only the most severe events are logged, but if the event log level is set to a higher number such as 5, more information is captured including less severe event subtypes.

For example, the Siebel Server components in the Enterprise Application Integration component group (alias EAI) have an event type called EAI Siebel Wizard. Several event subtypes belong to the EAI Siebel Wizard event type, including:

- EAI Siebel Wizard Invalid Business Component with a severity level of 2
- EAI Siebel Wizard Invalid MVG with a severity level of 2
- EAI Siebel Wizard MVG with a severity level of 3

While the EAI component group is running, the process encounters a multi-value group (MVG). This encounter creates an event of the EAI Siebel Wizard MVG subtype. If the MVG is invalid, a second event of the EAI Siebel Wizard Invalid MVG subtype is created. If the log level of the EAI Siebel Wizard event type is set to 1, both events are ignored. If the log level is set to 3, both events are written to the log file.

Events are logged at the Siebel Server level and the component level. For details on Siebel Server events, see "Configuring Siebel Server Logging" on page 68. For information on component events, see "Configuring Siebel Server Component Logging" on page 71.

About Event Attributes and Log File Format

Each event within the log file contains information about the associated application condition, including:

- Event Identifier
  - Type (category)
  - Subtype
- Timestamp
- Severity Level
- Details (metrics) about the event

For examples of individual events and their attribute values, see "Examples of Siebel Server Log Files" on page 69 and "Examples of Component Log Files" on page 74. For an example of a group of events collected within a log file, see "Example of a Detailed Component Log File" on page 80.

Events are written to and collected in a log file in the order of their occurrence. Each log file contains a header that provides information on the individual log file. The following is an example of a log file header:

```
ï»¿2021 2009-05-07 21:02:06 0000-00-00 00:00:00 -0800 00000000 001 003f 0001 09
SiebSrvr 2049 1364 1548 d:\sba81\siebsrvr\log\siebel81.server1.log 8.1.1 [21111] ENU
```

Table 18 provides descriptions of the example log file header details.

Table 18. Example of a Log File Header Detail With Descriptions

Log File Header Detail	Description
ï»¿	Byte Order Marker (BOM). The BOM is a Unicode format instruction. If the log file header opens with similar characters to the left, it indicates that the text editor used to view the log file cannot interpret the Unicode instruction.
2009-05-07 21:02:06	Time stamp of log file creation.
-0800	Offset of the local time from the GMT in the format ±HHMM.
SiebSrvr	The Siebel Server or component alias to which this log file refers.
2049	Task ID.
1364	OS Process ID (PID).
1548	Thread ID.
d:\sba81\siebsrvr\log\ siebel81.server1.log	Log filename.
8.1.1	Version number.
[21111]	Build number.
ENU	Language code.

About Siebel Server Log Files

Siebel Server log files record data for each individual Siebel Server deployed as part of a Siebel Enterprise Server. The Siebel application stores Siebel Server log files in the log directory for each individual Siebel Server as shown in Table 19.

Table 19. Siebel Server Log Directories

Operating System	Log Directory
Windows	*SIEBSRVR_ROOT*\log
UNIX	*SIEBSRVR_ROOT*/enterprises/*EnterpriseServerName*/*SiebelServerName*/log

Server log files use the following name convention: *EnterpriseServerName*.*SiebelServerName*.log.

Information contained in the Siebel Server log file can be used to determine where to search and investigate component log files for further information. The task ID, which makes up a part of the component log filename, is referenced in messages written to the Siebel Server log file. Locate the appropriate component task ID in the Siebel Server log file and open the task-specific component log that has the task ID in the log filename. For an example of this relationship, see "Example of Component Startup Log File" on page 74.

For further information and examples of Siebel Server log files, see "Viewing Siebel Server Log Files" on page 69 and "Examples of Siebel Server Log Files" on page 69.

About Component Log Files

Siebel Server component log files record data for each individual component and task functioning on a particular Siebel Server. These component log files are stored in the Siebel Server log directory on the Siebel Server in which the components are active as shown in Table 20. Using event logging with individual components allows you to isolate portions of a Siebel application.

Table 20. Log Directories for Siebel Server Components

Operating System	Log Directory
Windows	*SIEBSRVR_ROOT*\log
UNIX	*SIEBSRVR_ROOT*/enterprises/*EnterpriseServerName*/*SiebelServerName*/log

Component log files use the following naming convention:

<component_alias_name>_<SISProcID>_<taskid>.log

where:

- *component_alias_name* is the name of the component running the task.

- *SISProcID* is an internal four-character, zero-padded process ID that is rotating and incremented as component processes are spawned. The minimum numeric value allowed for SISProcID is 1. The maximum value allowed is 2047.
- *taskid* is a 32-bit internal zero-padded task ID number. Internally, the task ID contains a SISProcId as well as a counter maintained in each component process.

There is one process ID counter for all processes, not for each component. Therefore, you can sort the log files of a particular component by the specific component process.

Individual component task log files can also be consolidated into a single log file by setting the Use Shared Log File (alias LogUseSharedFile) component parameter. For more information about this parameter and on administering the Siebel Server and server component parameters, see *Siebel System Administration Guide*. For further information about and examples of component log files, see "Viewing Component Log Files" on page 74 and "Examples of Component Log Files" on page 74.

Configuring Siebel Server Logging

Siebel Server logging use event types that relate to Siebel Servers. For example, the Server State event type is a Siebel Server-level event that logs changes to the state of the Siebel Server. This topic describes how to configure and view Siebel Server event types. For details, see:

Setting Log Levels for Siebel Server Event Types

This topic describes setting log levels for Siebel Server event types using the Server Manager GUI or Server Manager command-line interface program (srvrmgr). For background information on event logging and event types, see "About Configuring Siebel Server and Component Logging" on page 63. To see the resultant Siebel Server log files, see "Viewing Siebel Server Log Files" on page 69. For examples of Siebel Server log files, see "Examples of Siebel Server Log Files" on page 69.

NOTE: The log level setting takes place immediately.

To set log levels for a Siebel Server event type on Server Manager GUI

1. Navigate to the Administration - Server Configuration, then Servers view.
2. In the Siebel Servers list, select the Siebel Server of interest.
3. From the view tabs, click Events.
4. In the Event Type list, select the Siebel Server Event Type of interest.

   For information on event types pertinent to a specific part of Siebel Business Applications, see product-specific documentation or details available on My Oracle Support.

5 In the Log Level field, choose the log level that you want to set for this event type.

For a list of log levels, see Table 17 on page 64.

6 Click the menu button and then Save Record.

To set log levels for a Siebel Server event type on srvrmgr

- Enter:

```
change evtloglvl event_alias_name=level for server siebel_server_name
```

To list Siebel Server event types on srvrmgr

- Enter:

```
list evtloglvl for server siebel_server_name
```

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Viewing Siebel Server Log Files

Siebel Server-level events are written to the Siebel Server log file. The log directory location on Windows is *SIEBSRVR_ROOT*\log. The log directory location on UNIX is *SIEBSRVR_ROOT*/enterprises/*EnterpriseServerName*/*SiebelServerName*/log. For background information on event logging and event types, see "About Configuring Siebel Server and Component Logging" on page 63. For more information and file naming conventions, see "About Siebel Server Log Files" on page 67. For examples of Siebel Server log files, see "Examples of Siebel Server Log Files" on page 69.

You can also view Siebel Server event logs from the Server Manager GUI. For information on this task, see "Monitoring Siebel Server Log Files" on page 32.

To assist in analyzing Siebel Server event log files, use the Log File Analyzer (LFA) utility to query and isolate log files of interest. For information on this feature, see Chapter 6, "Querying System Log Files."

Examples of Siebel Server Log Files

This topic provides examples of Siebel Server event log files. The event log format and information are detailed and described with the examples.

Example of Siebel Server Startup Log File

The following log file samples display what is written to the server log file during a regular startup of a Siebel Server. In this example, events are created that are defined by the event subtypes LstnObjCreate, ProcessCreate, and Startup, all of which have a severity of 1. For a detailed description of the sample output, see Table 21 through Table 23 on page 71. These events belong to the event type Server Logging (alias ServerLog). If this event type is set to a log level between 1 and 5, the following information is a sample of what is recorded in the log file.

LstnObjCreate Event Subtype

Table 21 describes the output for a LstnObjCreate event subtype for the following entry:

```
ServerLog LstnObjCreate 1 0 2003-05-13 11:35:10Created port 49173 for Server
Request Processor
```

Table 21. Event Subtype LstnObjCreate

Log Detail	Description
ServerLog	Event Type alias
LstnObjCreate	Event Subtype
1	Event Severity
0	SARM ID
2003-05-13 11:35:10	Date and time of log
Created port 49173 for Server Request Processor	Log message

Startup Event Subtype

Table 22 deoxyribose the output of a Startup event subtype for the following entry:

```
ServerLog Startup 1 0 2003-05-13 11:35:10Siebel Application Server is ready and
awaiting requests
```

Table 22. Event Subtype Startup

Log Detail	Description
ServerLog	Event Type alias
Startup	Event Subtype
1	Event Severity
0	SARM ID
2003-05-13 11:35:10	Date and time of log
Siebel Application Server is ready and awaiting requests	Log message

ProcessCreate Event Subtype

Table 23 describes the output of a ProcessCreate event subtype for the following entry:

```
ServerLog ProcessCreate 1 0 2003-05-13 11:35:10Created multithreaded server process
(OS pid = 2756) for File System Manager with task id 4114
```

Table 23. Event Subtype ProcessCreate

Log Detail	Description
ServerLog	Event Type alias
ProcessCreate	Event Subtype
1	Event Severity
0	SARM ID
2003-05-13 11:35:10	Date and time of log
Created multithreaded server process	Log message
(OS pid = 2756)	Operating System Process ID number
for File System Manager	Siebel Server Component
with task id 4114	Task ID number referencing the Siebel Server task

Configuring Siebel Server Component Logging

Component logging uses event types that relate to a specific Siebel Server component. For example, the SQL Tracing event type is a component-level event that traces SQL statements for a particular server component. This topic describes how to configure and view server component event types. For details see the following topics:

- "Setting Log Levels for Component Event Types" on page 72
- "Viewing Component Log Files" on page 74
- "Examples of Component Log Files" on page 74
- "Common Event Types for Component Diagnostics" on page 81
- "Common Event Types for Siebel Application Object Manager Diagnostics" on page 82

Setting Log Levels for Component Event Types

This topic describes setting log levels for server component event types using the Server Manager GUI or Server Manager command-line interface program (srvrmgr). For background information on event logging and event types, see "About Configuring Siebel Server and Component Logging" on page 63. To see the resultant Siebel Server component log files, see "Viewing Component Log Files" on page 74. For examples of Siebel Server component log files, see "Examples of Component Log Files" on page 74.

NOTE: The log level setting takes place immediately.

Setting Log Levels for Siebel Server Component Event Types Using the Server Manager GUI

Use the following procedure to set log levels for Siebel Server component event types using the Server Manager GUI.

To set log levels for a Siebel Server component event type using the Server Manager GUI

1. Navigate to the Administration - Server Configuration screen, then the Servers view.
2. In the Siebel Servers list, select the Siebel Server of interest.
3. Click the Components view tab.
4. In the Components list, select the Siebel Server component of interest.

   For example, you might select Call Center Object Manager (ENU).
5. Click the Events subview tab.
6. Select the Siebel Server component event type of interest.
   - For information on event types pertinent to a specific part of Siebel Business Applications, see product-specific documentation or details available on My Oracle Support.
   - For generic event types used in server component and Siebel Application Object Manager diagnostics, see "Common Event Types for Component Diagnostics" on page 81 and "Common Event Types for Siebel Application Object Manager Diagnostics" on page 82.
7. In the Log Level field, type in the log level you want to set for this event type, and then step off the record to save it.

   For a list of log levels and descriptions, see Table 17 on page 64.

Setting Log Levels for Siebel Server Component Event Types Using srvrmgr

Use the following procedures to set log levels for Siebel Server component event types using the Server Manager command-line interface program (srvrmgr).

To configure a component event type using srvrmgr

- Enter:

  change evtloglvl *event_alias_name=level* for component *component_alias_name*

To configure a server-specific component event type using srvrmgr

- Enter:

  change evtloglvl *event_alias_name=level* for server *siebel_server_name* component *component_alias_name*

To list component event types using srvrmgr

- Enter:

  list evtloglvl for component *component_alias_name*

To set server component event log levels for all users using srvrmgr

1. Make sure the values for the List of users component parameter are blank.

   For information about setting log levels for Siebel Server component parameters, see "Setting Log Levels for Component Event Types" on page 72.

2. Enter:

   change evtloglvl *event_alias_name=level4* for component *component_alias_name*

   Detailed log events for all users are written to the log file.

To set server component event log levels for a specific user using srvrmgr

1. Make sure the values for the List of users component parameter are set to SADMIN.

   For information about setting log levels for Siebel Server component parameters, see "Setting Log Levels for Component Event Types" on page 72.

2. Enter:

   change evtloglvl *event_alias_name=level4* for component *component_alias_name*

   Detailed log events for only SADMIN users are written to the log file.

   NOTE: The log level for other users remains the same.

For details on starting, running, and configuring the srvrmgr program, see *Siebel System Administration Guide*.

Viewing Component Log Files

Component-level events are written to log files for each task based on the component. The log directory location on Windows is *SIEBSRVR_ROOT*\log. The log directory location on UNIX is *SIEBSRVR_ROOT*/enterprises/*EnterpriseServerName*/*SiebelServerName*/log. Portions of component task log files can be viewed from the Server Manager GUI. For details, see "Monitoring Server Component Task Log Files" on page 38. Individual component task log files can also be consolidated into a single log file. For more information and file naming conventions, see "About Component Log Files" on page 67.

To assist in analyzing Siebel Server component event log files, use the Log File Analyzer (LFA) utility to query and isolate log files of interest. For information on this feature, see Chapter 6, "Querying System Log Files."

Examples of Component Log Files

This topic provides excerpts and examples of component event log files. The event log format and information are described with each of the examples.

Example of Component Startup Log File

The following log file sample displays what is written to the individual Siebel Server component log files during a regular startup of components running on a Siebel Server. In the following example, an event is created for the File System Manager component that is defined by the event subtype LstnObjInherit. For a detailed description of this sample output, see Table 24. This event has a severity of 3 and events of this subtype belong to the event type ServerLog. If this event type is set to a log level between 1 and 5, the following information is recorded in the log file.

```
ServerLog LstnObjInherit 3 0 2003-05-13 11:35:10Inherited listening object for port
49172
```

Table 24. Event Subtype LstnObjInherit

Log Detail	Description
ServerLog	Event Type alias
LstnObjInherit	Event Subtype
3	Event Severity
0	SARM ID
2003-05-13 11:35:10	Date and time of log
Inherited listening object for port 49172	Log message

This sample log file extract is from the component log file named FSMSrvr_4114.log and is located in the log directory of the Siebel Server. The task ID, 4114, which defines this log file title, corresponds to the log message in the appropriate Siebel Server log file. For this message, see Table 23 on page 71.

Example of Server Request Broker Log File

The following examples display log file entries in a sample Server Request Broker log file. The name of this log file is SRBroker_*TaskID*.log and is found in the Siebel Server /log directory. The first sample captures an event defined by the event subtype GenericInfo, which belongs to the component event type General Events (alias GenericLog). For a detailed description of this sample output, see Table 25. This event has a severity of 3 and is recorded to the log file if the General Event log level is set between 3 and 5.

```
GenericLog GenericInfo 3 0 2003-05-13 14:07:31Set environment variable
DB2CODEPAGE=1252
```

Table 25. Event Subtype GenericInfo

Log Detail	Description
GenericLog	Event Type alias
GenericInfo	Event Subtype
3	Event Severity
0	SARM ID
2003-05-13 14:07:31	Date and time of log
Set environment variable DB2CODEPAGE=1252	Log message

The next two samples belong to the component event type SQL Parse and Execute. Events were recorded of the event subtype Statement and Prepare + Execute. For detailed descriptions of the sample output, see Table 26 and Table 27 on page 76, respectively. Both of these event subtypes have a severity of 4 and are recorded to the log file if the SQL Parse and Execute event type is set to either 4 or 5.

Statement Event Subtype

Table 26 describes the output for a Statement event subtype for the following entry:

```
SQLParseAndExecute Statement 4 0 2003-05-13 14:07:38 select ROW_ID, NEXT_SESSION,
MODIFICATION_NUM from dbo.S_SSA_ID
```

Table 26. Event Subtype Statement

Log Detail	Description
SQLParseAndExecute	Event Type alias
Statement	Event Subtype
4	Event Severity
0	SARM ID
2003-05-13 14:07:38	Date and time of log
select ROW_ID, NEXT_SESSION, MODIFICATION_NUM from dbo.S_SSA_ID	SQL statement

Prepare + Execute Event Subtype

Table 27 describes the output for a Prepare + Execute event subtype for the following entry:

```
SQLParseAndExecute Prepare + Execute4 0 2003-05-13 14:07:38Time: 0s, Rows: 0, Avg.
Time: 0s
```

Table 27. Event Subtype Prepare + Execute

Log Detail	Description
SQLParseAndExecute	Event Type alias
Prepare + Execute	Event Subtype
4	Event Severity
0	SARM ID
2003-05-13 14:07:38	Date and time of log
Time: 0s, Rows: 0, Avg. Time: 0s	SQL Execution statistics

Example of a Log File for a Server Request Processor

The following code displays a log file entry in a sample server request processor log file, which can help you troubleshoot why a component request might not run. The name of this log file is SRProc_TaskID.log and is found in the SIEBELSRVR_ROOT\log directory. This code captures an event defined by the event SrmRouting subtype, which belongs to the SRMRouting component event type. Table 28 provides a detailed description of the sample output. This event has a log level of 4.

```
2021 2009-03-11 11:54:24 2009-03-11 12:50:55 +0530 000002d4 001 003f 0001 09
TestMTSBound 4194307 332 2976 m:\siebel\log\TestMTSBound_0004_4194307.log 8.1.1
[21102] ENU
```

Table 28. Event Subtype SrmRouting

Log Header Detail	Description
2021	Indicates the values of the LogEol and LogXlateMsgs parameters, as well as the file version and file completion indicators. The first number represents the LogEol parameter. This parameter can take values CRLF, LF, CR, or a custom value where: ■ CRLF represents \r\n and is shown as 2 in the log file. ■ LF represents \n and is shown as 1 in the log file. ■ CR represents \r and is shown as 0 in the log file. ■ A custom value is shown as 3 in the log file. The second number represents the file completion indicator as zero and does not change. The third number represents the file version indicator as 2 and does not change. The fourth number represents the value of the LogXlateMsgs (translate log file) parameter in the log file. The LogXlateMsgs parameter can take the values true or false where: ■ *True* indicates that log files are translated. It is shown as *1* in the log file. ■ *False* indicates log files are translated. It is shown as *2* in the log file. In this example, 2021 indicates the following: ■ LogEol = 2 (new line character is \r\n) ■ File completion error is 0 ■ File version is 2 ■ LogXlateMsgs = 1 (translate)
2009-03-11 11:54:24	Log file creation timestamp.

Table 28. Event Subtype SrmRouting

Log Header Detail	Description
2009-03-11 12:50:55	Log file completion timestamp.
+0530	Indicates the time difference of the Siebel Server time zone from GMT in the format ±HHMM, where *HH* represents hours, and *MM* represents minutes.
000002d4	The number of lines in the log file in hexadecimal format.
001	The segment number in decimal format.
003f	The LogEntryFlgs parameter value is applicable only if the LogEntryFmt parameter is set to *delimited*. If the LogEntryFmt parameter is set to *fixed*, then all the fields are logged. When the LogEntryFmt is set to delimited, the LogEntryFlgs parameter takes values in decimal format and then internally converts the values to binary format. For example, if the LogEntryFlgs parameter is set to the value of 63, the value is converted to its binary equivalent of 00111111, and is then processed further. The bits are numbered from right to left, starting from 0 to 7 (the seventh bit is 0, and the zero-position bit is 1). Given the following log file entry: `SisnTcpIp SisnSockDetail 4 00000cd049bd8290:0 42009-03-16 10:57:02 35928:LOCALTRANS-server] accept() timeout during get conn request` If the bit in the: ■ Zero position is set, this least significant bit (LSB) logs the main event type, which in the log file entry is *SisnTcpIp.* ■ First position is set, the subevent type is logged, which in the log file entry is *SisnSockDetail.* ■ Second position is set, the event severity is logged, which in the log file entry is *4.* ■ Third position is set, the timestamp is logged, which in the log file entry is *42009-03-16 10:57:02.* ■ Fourth position is set, the details are logged, which in the log file entry is *35928:LOCALTRANS-server] accept() timeout during get conn request.* ■ Fifth position is set, the flow ID and the SARM ID are logged, which in the log file entry is *00000cd049bd8290.*
0001	The number of characters in the LogFieldDelim parameter. The default value is \t.

Table 28. Event Subtype SrmRouting

Log Header Detail	Description
09	Represents the characters in the LogFieldDelim parameter. The default value is \t. For example, 09 represents the character \t, and 09 is the hexadecimal representation of the ASCII value of the character \t.
TestMTSBound	The name of the component.
4194307	The task ID.
332	The process ID of the component process.
2976	The thread ID.
m:\siebel\log\TestMTSBound_0004_4194307.log	The log file name.
8.1.1 [21102] ENU	The product version, including the language code.

Example of Component Error Log File

This example displays an error entry from a sample Assignment Manager component log file. The log file is located in the *SIEBSRVR_ROOT*\log directory and is named, AsgnSrvr_TaskID.log. The log message details an event defined by the event subtype GenericError, which belongs to the component event type General Events (alias GenericLog). For a detailed description of the sample output, see Table 29. An error event has a severity of 1 and is recorded to the log file if the General Event log level is set between 1 and 5.

```
GenericLog GenericError 1 0 2003-04-03 01:02:12[MERANT][ODBC Oracle 8
driver][Oracle 8]ORA-12541: TNS:no listener
```

Table 29. Event Subtype GenericError

Log Detail	Description
GenericLog	Event Type alias
GenericError	Event Subtype
1	Event Severity
0	SARM ID
2003-04-03 01:02:12	Date and time of log
MERANT][ODBC Oracle 8 driver][Oracle 8]ORA-12541: TNS:no listener	Error message

Example of a Detailed Component Log File

The previous log file examples are sample extracts from various component log files. As a final example, the following collection of log file messages display the output recorded to a log file after a successful task run by the Document Server component. This log file information is recorded when the appropriate event type log levels are set.

```
2021 2009-03-16 23:28:38 0000-00-00 00:00:00 -0600 00000000 001 003f 0001 09 SiebSrvr
0 5956 3856 d:\21112\ses\siebsrvr\log\siebel.sdc78275svqe.log 8.1.1 [21112] ENU

ServerLogServerStartup10000622e49ba14c8:02009-03-16 23:28:38Siebel Enterprise
Applications Server is starting up

ServerLogLstnObjCreate1000062d549ba14c8:02009-03-16 23:28:38Created port 49156 for
Workflow Process Batch Manager

ServerLogLstnObjCreate1000062d549ba14c8:02009-03-16 23:28:38Created port 49157 for
Workflow Recovery Manager

ServerLogLstnObjCreate1000062d649ba14c8:02009-03-16 23:28:38Created port 49158 for
Workflow Process Manager

ServerLogLstnObjCreate1000062d649ba14c8:02009-03-16 23:28:38Created port 49159 for File
System Manager

ServerLogLstnObjCreate1000062d649ba14c8:02009-03-16 23:28:38Created port 49160 for
Server Request Processor

ServerLogLstnObjCreate1000062d649ba14c8:02009-03-16 23:28:38Created port 49161 for
Siebel Administrator Notification Component

…

ServerLogProcessExit10000651d49ba14c8:02009-03-16 23:30:03SmartAnswer 6612 TERMINATED
Process 6612 was terminated

ServerLogComponentUpdate2000013f949bf1744:02009-03-16 23:30:07CommOutboundMgr
INITIALIZEDComponent has initialized.

ServerLogProcessCreate1000013f949bf1744:02009-03-16 23:30:15Created server process (OS
pid = 2660) for ServerMgr

ServerLogProcessCreate1000013f949bf1744:02009-03-17 00:45:51Created server process (OS
pid = 7624) for ServerMgr

ServerLogProcessCreate1000013f949bf1744:02009-03-17 03:43:39Created server process (OS
pid = 3236) for ServerMgr

ServerLogProcessExit10000651d49ba14c8:02009-03-17 03:53:25ServerMgr 2660 SUCCESS
Process 2660 completed Successfully

ServerLogProcessExit10000651d49ba14c8:02009-03-17 03:58:35ServerMgr 3236 SUCCESS
Process 3236 completed Successfully

ServerLogProcessCreate1000013f949bf1744:02009-03-17 03:58:48Created server process (OS
pid = 5816) for ServerMgr
```

```
ServerLogProcessExit10000651d49ba14c8:02009-03-17 03:59:13ServerMgr 5816 SUCCESS
Process 5816 completed Successfully

ServerLogProcessCreate1000013f949bf1744:02009-03-17 03:59:29Created server process (OS
pid = 5976) for ServerMgr

ServerLogProcessExit10000651d49ba14c8:02009-03-17 04:34:25ServerMgr 7624 SUCCESS
Process 7624 completed Successfully
```

Common Event Types for Component Diagnostics

Set the event types in Table 30 to the indicated log levels for general server component diagnostic purposes. The increased log levels either create log files for the server component of interest or increase the amount of logging information contained in the component log files. For a description on how to set log levels for component event types, see "Setting Log Levels for Component Event Types" on page 72.

CAUTION: Increased log levels require more memory and system resources. Make sure to return the event types to their previous values after completing diagnostics.

Table 30. Common Event Types for Component Diagnostics

Event Type Name	Event Type Alias	Log Level Setting
Component Tracing	Trace	4
General Events	GenericLog	4
Task Configuration	TaskConfig	4
SQL Tracing	SQL	4
SQL Error	SQLError	4
SQL Parse and Execute	SQLParseAndExecute	4

Common Event Types for Siebel Application Object Manager Diagnostics

Set the event types in Table 31 to the indicated log levels for general Siebel Application Object Manager (AOM) diagnostic purposes. The increased log levels either create log files for the AOM of interest or increase the amount of logging information contained in the AOM component log files. Increasing the event logging provides information about the individual processes and steps that are part of the AOM task.

For a description on how to set log levels for AOM component event types, see "Setting Log Levels for Component Event Types" on page 72.

CAUTION: Increased log levels require more memory and system resources. Make sure to return the event types to their previous values after completing diagnostics.

Table 31. Common Event Types for Siebel Application Object Manager Diagnostics

Event Type Name	Event Type Alias	Log Level Setting	Description
Event to track the flow of a message	MessageFlow	4	Captures messages exchanged between the Siebel Application Object Manager (AOM) and Siebel Web Server Extension (SWSE).
Object Manager Session Operation and SetErrorMsg Log	ObjMgrSessionLog	4	Captures user session login, logout, and timeout information.
		5	Captures user name and IP address when the session completes.
Event Context	EventContext	4	Captures applet and method executed, view names, and screen names that the user navigates to.
Object Manager Data Object Log	ObjMgrDataObjLog	5	Captures data manager object tracking; that is, the creation, use, and deletion of database connections, search specifications, sort specifications, and cursors.
General Object Manager Log	ObjMgrMiscLog	5	Captures general AOM events: load license, open SRF, errors, and so on.
Object Manager Business Component Operation and SetErrorMsg Log	ObjMgrBusCompLog	4	Captures Business Component-related events: create and delete.
Object Manager Business Service Log	ObjMgrBusServiceLog	4	Captures business service-related events: create, delete, methods invoked, and so on.
Main Thread Events	MainThread	4	Captures task counter, task creates, and task exits (in main Multithreaded Server log).
Task Related Events	TaskEvents	4	Captures task creation, context, session timeout, and close info.

Table 31. Common Event Types for Siebel Application Object Manager Diagnostics

Event Type Name	Event Type Alias	Log Level Setting	Description
SQL Parse and Execute	SQLParseAndExecute	4	Captures the SQL insert, update, and delete statements processed by the database connector. It includes the SQL statement and bind variables. The content is similar to the ObjMgrSqlLog event; however, the select statement is not captured by the SQLParseAndExecute event.
Object Manager SQL Log	ObjMgrSqlLog	4	Captures the SQL select, insert, update, and delete statements processed by the AOM data object layer. Includes the SQL statement and bind variables. It also captures the preparation, execute, and fetch time for the SQL cursor.
		5	Captures internal and customer-defined search and sort specifications, the joins processed for queries, as well as a call stack of the operation performed. Setting this event to log level 5 incurs a significant performance impact because a callstack is generated. Only set this event to log level 5 in consultation with Oracle Global Customer Support.
SQL Profiling	SQLProfiling	4	Captures SQL Profiling information. Helps aid in the diagnosis of a poorly performing component.
SQL Summary	SQLSummary	4	Captures SQL prepare, fetch, and execute times. Provides detailed information regarding the execution of an SQL statement.
SQL Slow Query	SQLSlowQuery	4	Captures SQL Performance— lists ten slowest performing queries.
Security Adapter Log	SecAdptLog	5	Captures security adapter tracing information to the AOM log file.
Security Manager Log	SecMgrLog	5	Captures security manager tracing information to the AOM log file.

5 Configuring Additional System Logging

This chapter describes other system logging configurations and information that can be used to uncover errors or improper application behavior in addition to Siebel Server and component event logging. It includes the following topics:

About Environment Variables for System Logging

The following system environment variables can be set to assist with logging other aspects of the Siebel application deployment. For information on configuring these environment variables on both Microsoft Windows and UNIX, see *Siebel System Administration Guide* or review the documentation specific to your operating system for details on changing these variables.

- **SIEBEL_LOG_EVENTS.** The SIEBEL_LOG_EVENTS environment variable sets the event logging level, which determines the extent of information captured in the log file. For level settings and descriptions of information captured, see Table 17 on page 64. More information is captured when the environment variable is set to a higher numeric value, and less information is captured when the variable is set to a lower numeric value. The numeric value is inversely proportional to the severity of the information—0 is more severe than 5 for instance. More disk space is consumed and performance is hindered when the value is set to a value of 5 than a value of 0.
- **SIEBEL_LOG_ARCHIVES**. The SIEBEL_LOG_ARCHIVES environment variable determines the number of log files archived. Set this value to a positive integer; this value indicates the number of files that are saved. For example, if the value is 3, then only the three most recent log files are retained, any additional log files are deleted. When a new log is created, `program.log`, the previous versions are archived as `program_1.log`, `program_2.log`, and so on. The numbers in the filename increase as the file becomes Statistics older. The oldest log file that numbers past the integer setting is deleted. The default value of this variable is ten.
- **SIEBEL_LOG_DIR.** The SIEBEL_LOG_DIR environment variable determines the log file location. Set this variable to change the location from the default directory. Make sure this directory already exists, access permission to write a file in that location is available, and sufficient space is free to support the log file.

- **SIEBEL_CRASH_HANDLER.** The SIEBEL_CRASH_HANDLER environment variable enables the creation of files when there is a malfunction or failure. For information on these files, see "About Other Siebel Server Log Files" on page 88. The default setting is 1, which enables the creation of such files. Setting this variable to 0 disables this function. Only set this variable in consultation with Oracle Global Customer Support. For help with setting this variable, create a service request (SR) on My Oracle Support.
- **SIEBEL_ASSERT_MODE.** The SIEBEL_ASSERT_MODE environment variable enables the creation of assert files. For information on assert files, see "About Other Siebel Server Log Files" on page 88. The default setting is 0, which disables the creation of assert files. Only set this variable in consultation with Oracle Global Customer Support. For help with setting this variable, create a service request (SR) on My Oracle Support
- **SIEBEL_SESSMGR_TRACE.** The SIEBEL_SESSMGR_TRACE environment variable enables tracing for session manager, which is part of the Siebel Web Server Extension (SWSE). By default, this variable is set to 0, which logs fatal and error events to the SWSE log file. For information on SWSE log files, see "About SWSE Logging" on page 11. To enable detailed logging of session manager, set this variable to 1. For further information on configuring logging for SWSE, see "Configuring SWSE Logging" on page 12.
- **SIEBEL_SISNAPI_TRACE.** The SIEBEL_SISNAPI_TRACE environment variable enables tracing for SISNAPI, which is a Siebel-proprietary communication protocol between the Web server and the Siebel Servers. By default, this variable is set to 0, which logs fatal and error events to the SWSE log file. For information on SWSE log files, see "About SWSE Logging" on page 11. To enable detailed logging of SISNAPI, set this variable to 1. For further information on configuring logging for SWSE, see "Configuring SWSE Logging" on page 12.
- **SIEBEL_STDERROUT.** The SIEBEL_STDERROUT environment variable enables logging of the standard error files. For further information on standard error files, see "Configuring Standard Error Files" on page 87. By default, this variable is set to 0, which disables standard error file logging. To enable logging of standard error files, set this variable to 1.

For information about environment variables for client-side logging for high interactivity, see "About Enabling and Disabling Client-Side Logging" on page 127.

Configuring Siebel Gateway Name Server Log Files

The Siebel Gateway Name Server log file, NameSrvr.log, is located in the LOG folder of the Siebel Gateway Name Server root directory. This file captures operational information when the Siebel Gateway Name Server System Service is started manually or when Siebel Gateway Name Server errors occur. For further details on the Siebel Gateway Name Server, see *Siebel System Administration Guide*.

To configure Siebel Gateway Name Server logging

1 On the computer running the Siebel Gateway Name Server, set the following environment variable to the given value:

   - SIEBEL_LOG_EVENTS = 3 (or higher)

   For further information on this variable, see "About Environment Variables for System Logging" on page 85.

   NOTE: If this value is set to 2 or lower, a Siebel Gateway Name Server log file is not created.

   For further information on setting environment variables, see *Siebel System Administration Guide*.

2 Stop and restart the computer running the Siebel Gateway Name Server for the environment variable to take effect.

NOTE: If the Siebel Gateway Name Server does not create log files, the log details might still reside in the operating system buffer. Shut down the name server to flush the logging information to the log file.

Configuring Standard Error Files

Standard error files contain process messages that are directed to standard error and standard out. These messages come from Siebel Server or third-party components and contain important information to help diagnose Siebel Server functionality issues. For example, the information contained in a Siebel Server process message can help identify instances where `siebmtshmw`, the process shell in which the Siebel Application Object Manager (AOM) component runs, is unable to start up due to problems like incorrect LIBPATH setting or a corrupt registry. For further information on Siebel Server processes, see *Siebel System Administration Guide*.

When configured, process messages are saved to file in the directory labeled *`SIEBSRVR_ROOT`*`/log/StdErrOut`. The format of the standard error files is as follows:

`stderrout_$`*`Process_ID`*`_$`*`Time_stamp`*`.log`

where:

- *`Process_ID`* is the operating system process ID number (PID).
- *`Time_stamp`* is the log file creation time in YYYY-MM-DD HH:MM:SS format.

Standard error file logging is not enabled by default.

To configure standard error file logging

1 On the computer running the Siebel Server, set the following environment variable to the given value:

- SIEBEL_STDERROUT = 1

For further information on this variable, see "About Environment Variables for System Logging" on page 85.

For further information on setting environment variables, see *Siebel System Administration Guide*.

2 Stop and restart the computer running the Siebel Server for the environment variable to take effect.

About Other Siebel Server Log Files

Siebel Business Applications generate other text log files in the binary (bin) subdirectory of the Siebel Server root directory. These files record conditional responses when certain portions of code are executed during the operation of the application. They appear in the following form listed in Table 32.

Table 32. Other Siebel Server Log Files

Log Filename	Description
siebel_assert*.txt	Indicates a fatal condition that might have led to a failure or data corruption.
siebel_crash*.txt	Indicates a process has malfunctioned or failed. These files are produced only on Windows and HP-UX platforms.
siebel_prefer*.txt	Indicate a less critical error condition that arises but did not lead to a failure, malfunction, or data corruption.

If these files are generated during the normal running of processes when no errors occur, they can be ignored (or deleted as they can become very large). However, if these files are generated when errors occur (especially failures), you can send these files to Oracle Global Customer Support for investigation by creating a service request (SR) on My Oracle Support.

About Flight Data Recorder Log Files

Siebel flight data recorder (FDR) files are records of system and server component behavior at run time. In the event of a system or server component failure, the settings and events leading up to the failure are captured and logged. You can send the Siebel flight data recorder log file to Oracle Global Customer Support by creating a service request (SR) on My Oracle Support. The log file is used to troubleshoot and analyze the specific settings and events that occurred before the failure.

The Siebel flight data recorder log files are stored in the Binary subdirectory of the Siebel Server root directory. They appear in the following form:

- T<YYYYMMDDHHMM>_P<process id value>.fdr

  where:

  - *YYYYMMDDHHMM* is the timestamp
  - *process id value* is the identification number of the process that failed or was stopped.

For example:

T200503181601_P001376.fdr

is a filename that is based on a component that was started on March 18, 2005 at 4:01 PM where the process id value was 1376.

The Siebel flight data recorder feature is enabled by default. However, FDR activation requires the execution of at least one instrumentation point to generate a log file. If a failure happens before execution of the first instrumentation point, no log file is generated. Instrumentation points are embedded in some workflow business services to provide capture-processing details in case of a system failure or server component failure. For more information about instrumentation and instrumentation points, see *Siebel Performance Tuning Guide* and *Siebel Business Process Framework: Workflow Guide*, respectively.

NOTE: FDR files are stored in binary format and cannot be read with a text editor.

Setting the environment variable SIEBEL_CRASH_HANDLER to 0 disables the creation of FDR files, in addition to several other logging functions. Only set this variable to 0 in consultation with Oracle Siebel Global Customer Support by creating a service request (SR) on My Oracle Support.

About Java EE Connector Architecture Logging

The Java EE Connector Architecture (JCA) provides a Java interface solution between application servers and Enterprise Information Systems (EIS). Siebel Business Applications support JCA with the Siebel Resource Adapter. The Siebel Resource Adapter supports the invocation of business services to perform operations, such as pooling connections and managing security. JCA allows you to keep logs for such operations. For more information about JCA logging, see *Transports and Interfaces: Siebel Enterprise Application Integration*. For more information about JCA, see:

http://java.sun.com/j2ee/connector

6 Querying System Log Files

Querying log files produced by a Siebel application is a useful diagnostic task to resolve problems that occur during any stage of operation. The Log File Analyzer (LFA) is a command-line utility that assists with this analysis. It includes the following topics:

About the Log File Analyzer

The Siebel Log File Analyzer (LFA) is a command-line utility designed to search through Siebel log files and isolate information of interest. Use the LFA to analyze and review the content of log files and to compile analysis information from these files.

Run the LFA to query log files across Siebel Servers and Siebel Web Server Extensions (SWSE) while filtering on one or more of the following items:

- User name
- Literal values
- Session IDs
- Component
- Log levels
- Events or subevents
- Time and date of log files

The LFA creates analysis output, which can be reviewed from the command-line or saved to text files.

For details on the process to run the LFA, see "Process for Analyzing Log Files with LFA" on page 93.

LFA Language Considerations

The LFA uses information in the events of the main Siebel Server log file to determine what components are available. The events in this log file are translated for different languages. To understand the format of the events for different languages, the LFA reads information in the language files located in the locale subdirectory of the Siebel Server root directory (for example, `/siebsrvr/locale`).

If the language files are changed, the LFA might not be able to recognize certain key events in the main Siebel Server log file, which lead to run-time errors.

Strategy for Analyzing Log Files

The strategy for analyzing log files depends on the type of issues encountered. Identify whether the issue of interest is related to a particular user or the application system in general. Run the Log File Analyzer (LFA) using the strategy applicable to the identified issue.

- For a strategy to use the LFA to examine user issues, see "Analyzing User Issues."
- For a strategy to use the LFA to examine system issues, see "Analyzing System Issues."

For information and details on the process of using the LFA, see "Process for Analyzing Log Files with LFA."

Analyzing User Issues

For user issues that are not immediately resolvable, log files provide additional information logged by the application regarding a user's time spent accessing and using the application.

The LFA gives the administrator the capability of querying across numerous log files for log events that were pertinent to the user's session. For example, in a situation where a user named Casey Smith reports an issue with her application at approximately 13:00, use the LFA to query events pertinent to Casey that occurred between 12:30 and 14:00. To refine the results, include the condition that the log level must be greater than or equal to one, which represents an error condition.

The LFA output includes information as to which file each log event came from. The administrator can, after finding an error or other log event of interest, check back in the original log file and look for events nearby that might give additional context useful for troubleshooting the issue.

NOTE: To query log files for users, make sure the environment variable SIEBEL_LOG_EVENTS is set to 4. For further information on environment variables, see "Common Event Types for Component Diagnostics" on page 81.

Analyzing System Issues

For general system issues not involving user issues (for example, a problem with a workflow), the LFA assists the administrator in isolating and resolving issues relating to general system usage.

For example, if the workflow processor is known to have failed within a particular time frame, use the LFA to search for log events that occurred during that time frame, and then look at the log files in which the events are contained for more specific detail.

As a preventative measure, the LFA is also useful to periodically check log files for any errors even if no system issue is apparent at that time.

Process for Analyzing Log Files with LFA

To analyze log files with the Log File Analyzer (LFA), perform the following tasks:

1. Configure the LFA to access the appropriate Siebel Server and Siebel Web Server Extension (SWSE) log files, if necessary. For further information on this task, see "Configuring the Log File Analyzer" on page 93.
2. Start the LFA. For further information on this task, see "Starting the Log File Analyzer" on page 97.
3. Query the log files using LFA. For information on this task, see "Creating and Saving LFA Queries" on page 99. For general information on running the LFA, see "About Running Log File Analyzer Commands" on page 99.

For strategies on analyzing log files using the LFA, see "Strategy for Analyzing Log Files" on page 92.

Configuring the Log File Analyzer

Configure the Log File Analyzer (LFA) by accessing and editing the LFA configuration file, which has the default name, logreader.cfg. The LFA uses the LFA configuration file when started to reference Siebel Server locations, Siebel Web Server Extension (SWSE) plug-in locations, and other run-time details.

This task is the first step in "Process for Analyzing Log Files with LFA" on page 93. Once the LFA is configured, this step is optional unless further changes are necessary.

The default location for the LFA configuration file is the binary subdirectory of the Siebel Server root directory (for example, `/siebsrvr/bin`).

The LFA configuration file contains sections that configure which log files are analyzed by the utility and what content is reviewed. Edit the appropriate sections in the configuration file with a text editor. For LFA configuration file parameters and their descriptions, see Table 33. For an example of a typical configuration file, see "Example of a Log File Analyzer Configuration File" on page 96.

Table 33. Log File Analyzer Configuration File Sections and Parameters

Section	Parameter	Description

[elements]	Siebel Server Identification Tag	Under the [elements] section, list Siebel Servers searchable by the LFA using the following format:
		Siebel Server Identification Tag = server
		where *Siebel Server Identification Tag* is a unique tag name identifying the Siebel Server of interest.
		This tag can be the Siebel Server name, but can also be any other configurable value, for example:
		`[elements]`
		`SiebelServer1=server`

	Siebel Web Server Extension Identification Tag	Under the [elements] section, list SWSE plug-ins searchable by the LFA using the following format:
		Siebel Web Server Extension Identification Tag = plug-in
		where *Siebel Web Server Extension Identification Tag* is a unique tag identifying the SWSE plug-in of interest.
		This tag can be the SWSE plug-in name, but can also be any other configurable value, for example:
		`[elements]`
		`SiebelSWSE1=plugin`

[Siebel Server Identification Tag]	Path	Each Siebel Server identification parameter listed in the [elements] section has a respective section of its own with its name in square brackets. The path parameter of each Siebel Server section denotes the location of the associated log files for that Siebel Server. For example:
		`[SiebelServer1]`
		`Path = //SiebSrv1/siebsrvr/log`

Table 33. Log File Analyzer Configuration File Sections and Parameters

Section	Parameter	Description

[Siebel Server Identification Tag.Siebel Server Component Name]	shortname	List Siebel Server component display names in square brackets to allow the LFA to search for component references in log files specific to a Siebel Server. Add the Siebel Server component alias as the value for the short name parameter. For example:
		`[SiebelServer1.Server Request Broker]`
		`shortName=SRBroker`
		For a listing of Siebel Server components and their aliases, see *Siebel System Administration Guide*.

[Siebel Web Server Extension Identification Tag]	Path	Each SWSE plug-in identification parameter listed in the [elements] section has a section of its own with its name in brackets ([ ]). The path parameter of each SWSE plug-in section denotes the location of the associated log files for that SWSE plug-in. For example:
		`[SiebelSWSE1]`
		`Path = //SWSE1/eappweb/log`

[Render]	event	Displays information on log events, if enabled. Set to 1 to enable; set to 0 to disable. The parameter information in the [render] section is also controlled by using commands during the running of the LFA. For more information, see "About Running Log File Analyzer Commands" on page 99.
	subevent	Displays information on log sub events if enabled. Set to 1 to enable; set to 0 to disable.
	loglevel	Displays information on log level of event subtypes. Set to 1 to enable; set to 0 to disable.
	time	Displays log timing information in enabled.Set to 1 to enable; set to 0 to disable.
	file	Displays log file path information if enbabled.Set to 1 to enable; set to 0 to disable.

NOTE: Do not modify the sections entitled [schemes], [user], and [session].

Example of a Log File Analyzer Configuration File

The following example Log File Analyzer (LFA) configuration file is intended for a Siebel application with two Siebel Servers, named SiebSrv1 and SiebSrv2, and three Web servers with three Siebel Web Server Extensions (SWSE), named SWSE1, SWSE2, and SWSE3. The LFA configuration file also contains alias information on two Siebel Server components, Server Request Broker and Call Center Object Manager. Using this configuration file, the LFA searches all Siebel Server and SWSE log files, has the ability to search on the two Siebel Server components listed, and displays all information except log level and the log file path.

For descriptions of the individual sections and parameters, see "Configuring the Log File Analyzer" on page 93.

```
[elements]

SiebSrv1=server

SiebSrv2=server

SWSE1=plugin

SWSE2=plugin

SWSE3=plugin

[SiebSrv1]

Path = //SiebSrv1/siebsrvr/log

[SiebSrv2]

Path = //SiebSrv2/siebsrvr/log

[SiebSrv1.Server Request Broker]

shortName=SRBroker

[SiebSrv2.Call Center Object Manager (ENU)]

shortName=SCCObjMgr

[SWSE1]

Path = //SWSE1/eappweb/log

[SWSE2]

Path = //SWSE2/eappweb/log

[SWSE3]

Path = //SWSE3/eappweb/log

[Render]

event=1

subevent=1
```

```
loglevel=0

time=1

file = 0
```

Starting the Log File Analyzer

Starting the Log File Analyzer (LFA) is the second step in the "Process for Analyzing Log Files with LFA" on page 93. For background information on the LFA, see "About the Log File Analyzer" on page 91.

The LFA utility resides in the binary subdirectory of Siebel Server root directory under Microsoft Windows as the executable logreader.exe or as binaries under UNIX.

The procedure for starting the LFA under Microsoft Windows is available in "Starting the Log File Analyzer Under Microsoft Windows" on page 97.

The procedure for starting the LFA under UNIX is available in "Starting the Log File Analyzer Under UNIX" on page 98.

Starting the Log File Analyzer Under Microsoft Windows

Use the following command to start the Log File Analyzer (LFA) command-line utility under Microsoft Windows.

To start the Log File Analyzer under Microsoft Windows

1. Navigate to the binary subdirectory within the Siebel Server root directory (for example, `/siebesrvr/bin`).
2. Make sure the LFA configuration file (logreader.cfg) is present in the same directory as the utility. If this file is located in another directory, or has another name, use the `/f` parameter described in Table 34.

   For further information on the configuration file, see "Configuring the Log File Analyzer" on page 93.
3. At the Windows command prompt, enter `logreader.exe` using, as necessary, parameters listed in Table 34.

   The log reader command prompt appears after a successful start as follows:

   ```
   logreader>
   ```
4. Run the LFA by using the commands described in "About Running Log File Analyzer Commands" on page 99.

NOTE: Make sure the DLL files `MSVCR70D.dll` and `MSVCP70D.dll` are present in the LFA directory.

Table 34 describes the parameters available for use during the starting of the LFA.

Table 34. Log File Analyzer Parameters

Parameter	Description	Example
/h	Lists the parameters available for use with the LFA utility.	`logreader /h`

/f	Locates the LFA configuration file if not present in LFA utility directory or if the configuration file is named differently than logreader.cfg. Include the path or new configuration filename after the /f parameter. If the configuration filename includes a space, enclose the argument with quotation marks.	`logreader /f abc.cfg`
		or
		`log reader /f g:\abc\abc.cfg`

/i	Specifies an input file that contains LFA commands. At startup, the LFA provides output from the commands listed in the input file. Include the filename and path, if necessary, after the `/i` parameter.	`logreader /i g:\abc\abc.txt`

NOTE: Use the `/f` and `/i` parameters independently or together.

Starting the Log File Analyzer Under UNIX

Use the following command to start the Log File Analyzer (LFA) command-line utility under UNIX.

To start the Log File Analyzer under UNIX

1. Make sure the LD_LIBRARY_PATH (Oracle Solaris), SHLIB_PATH (HP-UX), or LIBPATH (AIX) environment variable contains the full pathname for your database client library directory. For more information on these variables, see *Siebel Installation Guide for UNIX*.
2. Make sure the LFA configuration file (logreader.cfg) is present in the same directory as the utility. If this file is located in another directory, or has another name, use the `/f` parameter described in Table 34. For further information on the configuration file, see "Configuring the Log File Analyzer" on page 93.
3. Enter `logreader` using, as necessary, other parameters listed in Table 34.

   The log reader command prompt appears after a successful start as follows:

   ```
   logreader>
   ```

4. Run the LFA by using the commands described in "About Running Log File Analyzer Commands" on page 99.

About Running Log File Analyzer Commands

Running the Log File Analyzer (LFA) allows you to search and filter information contained in Siebel application log files. For overall strategy on running the LFA, see "Strategy for Analyzing Log Files" on page 92.

Make sure when running the LFA that you enter commands and parameters correctly. The following information is common to all LFA commands:

- The LFA is case sensitive.
- Enclose any parameters that contain spaces with quotation marks.

The following topics list instructions for running the LFA:

- "Creating and Saving LFA Queries" on page 99. Creating and executing a query is the fundamental task associated with the LFA.
- "Filtering LFA Queries" on page 106. Filtering queries assists the user to isolate diagnostic information of interest.

NOTE: Move log files to a nonproduction environment before querying them with the LFA. As the LFA parses through potentially large and numerous log files, using the LFA in a production environment might reduce overall system performance.

Creating and Saving LFA Queries

Creating and executing a query is the fundamental task associated with the Log File Analyzer (LFA). Creating saved queries is a task in the "Process for Analyzing Log Files with LFA" on page 93.

Run queries using the LFA query command to search log files based on users, literal values, sessions, severity, events, subevents, log times, or combinations of these items.

For descriptions on running these commands, see the following topics.

The LFA saves the results of each query to memory or saves it to a text file. For details on displaying saved queries, see "Displaying Saved Query Output" on page 107. For details on saving output to a text file, see "Saving Log File Analyzer Output to Text Files" on page 107.

To stop a query before it finishes, see "Interrupting Log File Analyzer Queries" on page 108.

Querying Log Files for Users

To query log files for users, you must first set an environment variable log level, and add sections to the logreader.cfg file.

NOTE: To run queries for a specific user using the option *user*, you must run the query against the Siebel Web Server log files.

Use the following procedures to query log files for users.

To prepare for querying events associated with users

1. Make sure the SIEBEL_LOG_EVENTS environment variable is set to 4.

   For more information on setting environment variables, see "Setting Log Levels for Component Event Types" on page 72.

2. Add new sections to the logreader.cfg file by doing the following:

   a. Add a SWE=plugin section under the [elements] section.

   b. Create a new [SWE] section with the following:

      Path = <*swe log files directory location*>.

Use the following procedure to search log files for events associated with individual users.

To query for events associated with a particular user

- Using the command-line interface, enter:

  query *query_name* where user = *user_name*

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *user_name* is the user of interest in log files.

An example of this query command is as follows:

```
query asqry where user = asmith
```

This command queries log files for events associated with user asmith and saves the output to memory under the name asqry.

For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

Querying Log Files for Literal Values

Use the following procedure to search log files for specific literal values. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

To query for a literal value

- Enter:

  query *query_name* where literal = *literal_value*

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *literal_value* is the literal value of interest in log files.

An example of this query command is as follows:

```
query litqry where literal = Parameter
```

This command queries log files for events associated with literal `Parameter` and saves the output to memory under the name `litqry`.

Querying Log Files for Error Messages

Use the following procedure to search log files for error messages. This command is an application of querying for literal values. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

To query for an error message

- Enter:

  query *query_name* where literal = *error_message_number*

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *error_message_number* is the error message number of interest in log files.

An example of this query command is as follows:

```
query errorqry where literal = SBL-ASG-00001
```

This command queries log files for events associated with error message number `SBL-ASG-00001` and saves the output to memory under the name `errorqry`.

Querying Log Files for Sessions

Use the following procedure to search log files for specific sessions. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

To query for events associated with a particular session

- Enter:

  query *query_name* where session = *session_ID*

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *session_ID* is the session ID of interest in log files.

An example of this query command is as follows:

```
query sesqry where session = !1.15bc.c425.3f302b17
```

This command queries log files for events associated with session ID `!1.15bc.c425.3f302b17` and saves the output to memory under the name `sesqry`.

Querying Log Files of a Particular Severity

Use the following procedure to search log files for events of a specific severity. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

Events are categorized from 0 to 5, 0 being the most severe or critical. For further information on event severity and event logging, see *Siebel System Administration Guide*.

This command includes events of the indicated severity as well as events of a greater severity. For example, if you query for a severity of 2, events of severity 0 and 1 are also included in the output.

To query for events associated with a particular severity

- Enter:

  `query query_name where loglevel = severity_value`

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *severity_value* is the severity value of interest (integer value from 0 to 5).

An example of this query command is as follows:

```
query svtqry where loglevel = 1
```

This command queries log files for events associated with a severity of 0 and 1 and saves the output to memory under the name `svtqry`.

Querying Log Files for a Particular Log Event

Use the following procedure to search log files for a specific log event. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

For a partial listing of log events and for further information on event logging, see *Siebel System Administration Guide*.

To query for events associated with a particular log event

- Enter:

  `query query_name where event = event_name`

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *event_name* is the log event name of interest.

An example of this query command is as follows:

```
query evtqry where event = SessMgr
```

This command queries log files for log events named `SessMgr` and saves the output to memory under the name *evtqry*.

Querying Log Files with a Particular Log Subevent

Use the following procedure to search log files for a specific log subevent. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

For a partial listing of log subevents and for further information on event logging, see *Siebel System Administration Guide*.

To query log entries associated with a particular log subevent

- Enter:

  query *query_name* where subevent = *subevent_name*

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *subevent_name* is the log subevent name of interest.

An example of this query command is as follows:

```
query subevtqry where subevent = SisnNetGeneric
```

This command queries log files for log subevents named `SisnNetGeneric` and saves the output to memory under the name subevtqry.

Querying Log Files After a Particular Time

Use the following procedure to search log files created after a specific time. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

To query events logged after a certain time

- Enter:

  query *query_name* where time from "YYYY-MM-DD HH:MM:SS"

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *"YYYY-MM-DD HH:MM:SS"* is the date and time of interest.

  NOTE: The exact time portion of the date and time parameter, `HH:MM:SS`, can be omitted. In this case, the date's base time defaults to `00:00:00`.

An example of this query command is as follows:

```
query timeqry where time from "2003-10-01 16:30:00"
```

This command queries log files created after October 1, 2003 at 4:30 PM, and saves the output to memory under the name `timeqry`.

This command is useful in combination with other parameters to filter results. For further information, see "Querying Log Files Using Multiple Conditions" on page 105.

Querying Log Files Within a Time Interval

Use the following procedure to search log files created within a specific time interval. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

To query events logged within a certain time interval

- Enter:

  query *query_name* where time from "YYYY-MM-DD HH:MM:SS" to "YYYY-MM-DD HH:MM:SS"

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *"YYYY-MM-DD HH:MM:SS"* is the date and time of interest.

  NOTE: The exact time portion of the date and time parameter, `HH:MM:SS`, can be omitted. In this case, the date's from-time defaults to `00:00:00` and the to-time defaults to `23:59:59`.

An example of this query command is as follows:

```
query timeintqry where time from "2003-10-01 16:30:00" to "2003-10-05"
```

This command queries log files created between October 1, 2003 at 4:30 PM and October 5, 2003 at 11:59 PM, and saves the output to memory under the name `timeintqry`.

This command is useful in combination with other parameters to filter results. For further information, see "Querying Log Files Using Multiple Conditions" on page 105.

Querying Log Files for Components

Use the following procedure to search log files for a specific Siebel Server component. For other options of the Log File Analyzer (LFA) query command, see "Creating and Saving LFA Queries" on page 99.

Make sure the LFA configuration file contains information on the Siebel Server component of interest. For further information, see "Configuring the Log File Analyzer" on page 93.

For further information on Siebel Server components, see *Siebel System Administration Guide*.

To query log entries for a particular Siebel Server component

- Enter:

  query *query_name* where component = *component__name*

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *component__name* is the Siebel Server component name of interest.

  NOTE: The *component__name* parameter takes either the long form or alias form of the Siebel Server component name. For a list of component names and aliases, see *Siebel System Administration Guide*.

An example of this query command is as follows:

```
query compqry where component = SCCObjMgr
```

This command queries log files for the Call Center Object Manager (alias SCCObjMgr) and saves the output to memory under the name `compqry`.

Querying Log Files Using Multiple Conditions

This topic provides examples of combination query commands using multiple conditions. For a list of individual query command conditions and their use, see "Creating and Saving LFA Queries" on page 99.

The logical AND and OR operators are also applicable to the Log File Analyzer (LFA) query command. To add clarity to multiple condition commands, group condition sets in parentheses.

- `query litasqry where (literal = Parameter) or (user = asmith)`

  This command queries log files for the literal `Parameter` or the user `asmith`. It saves the output to memory under the name `litasqry`.

- `query aqry where literal = Parameter and literal = SBL-GEN`

  This command queries log files for the literal `Parameter` and the literal `SBL-GEN`. It saves the output to memory under the name `aqry`.

- `query asaugqry where user = asmith time from 2003-08-05`

  This command queries log files for the user `asmith` after August 05, 2003. It saves the output to memory under the name `asaugqry`.

- `query asaugqry where user = asmith time from "2003-08-05 15:20:00" to "2003-08-05 15:30:00"`

  This command queries log files for the user `asmith` during the ten-minute period between 3:20 PM and 3:30 PM on August 05, 2003. It saves the output to memory under the name `asaugqry`.

Filtering LFA Queries

Use the show command to further refine the output of saved queries. For information on querying log files and creating saved queries, see "Creating and Saving LFA Queries" on page 99.

For information on displaying a saved query or multiple saved queries, see "Displaying Saved Query Output" on page 107.

To filter saved query information

- Enter:

  show *query_name where_clause*

  where:

  - *query_name* is the query command output stored in memory under this name.
  - *where_clause* is the WHERE clause used to filter display results using key words.

For a list of key words available for use with the Log File Analyzer (LFA), see "Listing Query Command Key Words" on page 108. The syntax of where clauses used with the show command are similar to those used with the query commands. Review "Creating and Saving LFA Queries" on page 99 for further information.

Use multiple where clause conditions and the logical operators AND and OR to further filter an individual or multiple saved queries. For examples of these types of commands, see "Examples of Filtered Saved Queries" on page 106.

To save filtered output from the show command, save the results to a text file. For description of this task, see "Saving Log File Analyzer Output to Text Files" on page 107. Filtered output from the show command cannot be saved in memory.

Examples of Filtered Saved Queries

The following examples display the type of filtering available on saved queries using the show command.

- `show aquery where user = asmith`

  This command filters the saved query aquery for information specific to user asmith.

- `show aquery where user = asmith and literal = Parameter time from "2003-08-05 15:20:20" to "2003-08-05 15:30:00" > out.dat`

  This command filters the saved query aquery for information on user asmith and the literal value parameter between the time of 3:20 and 3:30 PM on August 05, 2003. The command also stores the results of the filtered query to a text file named out.dat.

- `show aquery, bquery where user = asmith and literal = Parameter time from "2003-08-05 15:20:20" to "2003-08-05 15:30:00" > out.dat`

  This command filters the saved queries aquery and bquery based on the same conditions in the previous bullet.

Saving Log File Analyzer Output to Text Files

Use the following procedure to save the results of a Log File Analyzer (LFA) command to a text file. For information on running the LFA, see "About Running Log File Analyzer Commands" on page 99. Any LFA command that creates output can have the output channeled to a file.

To save Log File Analyzer output to text files

- Enter:

  log_file_analyzer_command > file_name.txt

  where:

  - *log_file_analyzer_command* is the LFA command.
  - *file_name.txt* is the name of the output text file.

  Make sure to:

  - Include the > character when saving output to a text file.
  - Specify a path name with the text filename if you want to save the log file to another directory, and not the Log File Analyzer (LFA) directory.

Example:

```
query litqry where literal = Parameter > output1.txt
```

This command saves the output from the litqry saved query to the text file named `output1.txt`. The LFA stores this output text file in the save directory as the Log File Analyzer directory.

Displaying Saved Query Output

Use the following procedures to display results of one or more saved query commands to the screen. For a listing of saved queries, see "Listing Log File Analyzer Queries and Run-time Details" on page 111.

For more information on the query command, see "Creating and Saving LFA Queries" on page 99.

The Log File Analyzer (LFA) also saves query command output to text files. For further information on this task, see "Saving Log File Analyzer Output to Text Files" on page 107.

To show saved query output to the screen

- Enter:

  show *query_name*

  where:

  query_name is the query command output stored in memory under this name.

Example:

```
show evtqry
```

This example displays the output from a previous query command named evtqry.

NOTE: The LFA only displays queries saved to memory during a given session.

To show multiple saved query output to the screen

- Enter:

  show *query_name_1, query_name_2, ... , query_name_N*

  where:

  query_name_N is the query command output stored in memory under this name.

Example:

```
show evtqry1, evtqry2
```

This example displays the output from two previous query commands named evtqry1 and evtqry2.

Interrupting Log File Analyzer Queries

Use the following procedure to interrupt a query command. For more information on the query command, see "Creating and Saving LFA Queries" on page 99.

To interrupt a query command in operation

- Press CTRL-C during the operation of the command.

Listing Query Command Key Words

Use the following procedure to list the key words available for use with the query command where clause. For detailed descriptions of use for each key word, see "Creating and Saving LFA Queries" on page 99.

To list the query command key words

- Enter:

  keys

  The key words are output to the screen.

Listing Log Event Fields Display Status

Use the following procedure to list the display status for log event fields. The value 1 indicates the log event field is set to display. The value 0 indicates the log event field is set to hide.

To list log event fields display status

- Enter:

  `fields`

To change the display status at run-time, see the task "Showing Log Event Fields in LFA Results" on page 109 or "Hiding Log Event Fields in LFA Results" on page 110 for further information.

Set the default display status of the event log fields by modifying the Log File Analyzer (LFA) configuration file. For more information on the LFA configuration file, see "Configuring the Log File Analyzer" on page 93.

Showing Log Event Fields in LFA Results

Use the following procedures to show log file fields in the output from the Log File Analyzer (LFA) during an individual LFA session. You can also set this information in the LFA configuration file, which is applicable to all LFA sessions. For further information, see "Configuring the Log File Analyzer" on page 93.

To list the current event log field display status, see "Listing Log Event Fields Display Status" on page 109.

To show log file fields in the LFA output

- Enter:

  `showfield` *`log_field_name`*

  where:

  `log_field_name` is the name of the log field name for display.

For a list of the available display fields, see Table 35.

Set multiple log file fields to show on a single `showfield` command by separating each log file field with a space or comma.

Table 35. Configurable Log File Fields

Log File Field	Description
event	Name of the event.
subevent	Name of the subevent.
loglevel	Severity of the log file event.

Table 35. Configurable Log File Fields

Log File Field	Description
file	File and path name of the log file.
time	Date and time of the log file.

Hiding Log Event Fields in LFA Results

Use the following procedures to hide log file fields in the output from the Log File Analyzer (LFA) during an individual LFA session. You can also set this information in the LFA configuration file, which is applicable to all LFA sessions. For further information, see "Configuring the Log File Analyzer" on page 93.

To list the current event log field display status, see "Listing Log Event Fields Display Status" on page 109.

To hide log file fields in the LFA output

- Enter:

  `hidefield` *`log_field_name`*

  where:

  `log_field_name` is the name of the log field name for display. For a list of the available display fields, see Table 35 on page 109.

Set multiple log file fields to hide on a single `showfield` command by separating each log file field with a space or comma.

Deleting Log File Analyzer Saved Query Results

Use the following procedure to delete saved queries. For further information on querying log files, see "Creating and Saving LFA Queries" on page 99.

NOTE: Deleting saved queries does not delete queries saved as text files.

To delete Log File Analyzer query results

- Enter:

  `delete` *`query_name`*

  where:

  `query_name` is the query command output stored in memory under this name.

Delete multiple saved queries by separating each query name with a space or comma when using the `delete` command.

Listing Log File Analyzer Queries and Run-time Details

Use the `list` command in the following procedure to list saved queries and run-time details to the screen. For information on running the Log File Analyzer (LFA), see "About Running Log File Analyzer Commands" on page 99. For information on creating saved queries, see "Creating and Saving LFA Queries" on page 99.

For information on each list item, see "Listing Log File Information Using Log File Analyzer" on page 112 for details.

To list Log File Analyzer queries and run-time details

- Enter:

  `list` *`list_item`*

  where:

  `list_item` is the list item of interest.

For items available for listing, see Table 36.

Table 36. Log File Analyzer Items Available for Listing

Item	Description
all	Lists all LFA items available for listing. **NOTE:** The LFA does not list users or sessions until you perform at least one user query.
queries	Lists LFA queries saved in the current session.
servers	Lists servers searched by LFA.
sessions	Lists sessions found in the log files searched by LFA.
plugins	Lists plug-ins searched by LFA.
components	Lists components with information in log files searched by LFA.
processes	Lists processes with information in log files searched by LFA.
users	Lists users with information in the log files searched by LFA.

NOTE: If the LFA is not searching the appropriate server or plug-in, see "Configuring the Log File Analyzer," for details on configuring the LFA to search the server and plug-in of interest.

Listing Log File Information Using Log File Analyzer

Use the info command in the following procedure to list detailed information on the values of the run-time details. For a list of items available for use with the info command, see "Listing Log File Analyzer Queries and Run-time Details" on page 111.

For information on running the Log File Analyzer (LFA), see "About Running Log File Analyzer Commands" on page 99. For information on creating saved queries, see "Creating and Saving LFA Queries" on page 99.

To list information on values for Log File Analyzer run-time details

- Enter:

  info *info_item*

  where:

  info_item is the value of a list item of interest.

For items available for listing (with the exception of list item all and queries), see Table 36 on page 111.

List information on multiple list values by separating values with a comma or space for the *info_item* parameter.

For example, using the list command for users revealed an entry named asmith. Use the following command to list information on asmith:

```
info asmith
```

Exiting Log File Analyzer

Use the following command to exit the log file analyzer. Exiting the log file analyzer deletes saved queries for that session unless query output is saved to text files. For information on this task, see "Saving Log File Analyzer Output to Text Files" on page 107.

To exit the Log File Analyzer

- Enter:

  ```
  exit
  ```

Troubleshooting Log File Analyzer Errors

This topic provides guidelines for resolving errors and problems that the Log File Analyzer (LFA) might generate during processing. To resolve the problem, look for it in the Symptom/Error Code and Message column in Table 37.

Table 37. Resolving Log File Analyzer Problems

System or Error Code and Message	Diagnostic Steps or Cause	Solution
SBL-LFA-00100 Section [%s] in configuration file is empty.	The section indicated in the error message is blank. LFA requires content for this section.	For the correct specification of the configuration file, see "Configuring the Log File Analyzer" on page 93.
SBL-LFA-00101 Rule "%s" appears in the configuration file but is not registered.	A rule has been added to the LFA configuration file but not registered with the utility. Therefore, the rule is not recognized.	At this time, it is not possible to create customized rules for the LFA. Remove this rule from the configuration file.
SBL-LFA-00102 Cannot find section [%s] in the configuration file.	Though it is a required section, the section of the LFA configuration file indicated in the error message text is missing.	For the correct specification of the configuration file, see "Configuring the Log File Analyzer" on page 93.
SBL-LFA-00103 There is a format problem in section [%s] of the configuration file.	There is a formatting error in the LFA configuration file section indicated in the error message text.	For the correct specification of the configuration file, see "Configuring the Log File Analyzer" on page 93.
SBL-LFA-00104 Value "%s" in the section is invalid or missing.	There is a missing value in the LFA configuration file section indicated in the error message text.	For the correct specification of the configuration file, see "Configuring the Log File Analyzer" on page 93.
SBL-LFA-00105 Time filters are invalid or have contradictory values.	The time filter you are trying to use in your query is invalid. It is possible that the *To* time is before the *From* time.	For information on using time filters correctly, see "Querying Log Files Within a Time Interval" on page 104.
SBL-LFA-00106 Value or Name for "%s" is a negative number.	This value is not expected to be negative.	Provide a positive value.
SBL-LFA-00107 Cannot open file: "%s".	The LFA cannot write output to the given file.	Check your permissions to the file and directory. Make sure the file is not read only.

Table 37. Resolving Log File Analyzer Problems

System or Error Code and Message	Diagnostic Steps or Cause	Solution
SBL-LFA-00108 File "%s" is already in use.	This file might be locked by another running application.	Shut down applications that might be accessing the file and try again.
SBL-LFA-00109 Cannot create pipe for command \"%s\".	Pipe is not supported.	This functionality is not supported.
SBL-LFA-00110 OUT OF MEMORY !!!!!!	The computer on which you are using the LFA has run out of memory.	Shut down some of your applications and try again.
SBL-LFA-00112 Query's "where" clause is invalid.	The where clause in the query is not correctly specified.	For information on correct application of the WHERE clause, see "Creating and Saving LFA Queries" on page 99.
SBL-LFA-00113 Query with name "%s" does not exist.	You have tried to reference a query that does not exist.	Type list queries to see existing queries. If your query does not exist, you must create it before trying to reference it. For information on creating queries, see "Creating and Saving LFA Queries" on page 99.
SBL-LFA-00114 Filter for "%s" does not exist.	The specified parameter cannot be used as a filter.	Do not use this item as a query parameter.
SBL-LFA-00115 Category "%s" does not exist.	You tried to use the specified word, but only key words are expected	Fix the command and try again. For information on key words, see "Listing Query Command Key Words" on page 108.
SBL-LFA-00116 Object "%s" does not exist.	The object (that is, Siebel Server, plug-in, query, user, component, or session) that you are trying to reference is unavailable.	Make sure the object is available for reference. For information on listing existing objects, see "Listing Log File Analyzer Queries and Run-time Details" on page 111.
SBL-LFA-00117 Object "%s" already exists. Please use another name.	An object by that name already exists.	Use another name for your object.

Table 37. Resolving Log File Analyzer Problems

System or Error Code and Message	Diagnostic Steps or Cause	Solution
SBL-LFA-00118 Query "%s" finished abnormally.	The query finished abnormally, possibly due to corrupt log files or user intervention.	Re-run the query. If that does not work and the query is complex, try simplifying it.
SBL-LFA-00119 "%s" should not be used for naming.	The name you have specified cannot be used.	Use another combination of characters.
SBL-LFA-00120 Cannot interpret: "%s"	The name you have specified cannot be used in this place.	The LFA identified an error in your command syntax. For information on valid LFA commands, see "About Running Log File Analyzer Commands" on page 99.
SBL-LFA-00121 Token has a wrong value: "%s"	The specified value is invalid.	For information on valid LFA commands, see "About Running Log File Analyzer Commands" on page 99.
SBL-LFA-00122 Unknown issue.	There is an error in the command that you have entered.	For information on valid LFA commands, see "About Running Log File Analyzer Commands" on page 99.
SBL-LFA-00123 There is no file "%s".	The input file that you specified when starting the LFA does not exist.	Make sure the file exists and the filename and path is correct.
SBL-LFA-00124 Wrong format of the string: "%s".	The specified string is formatted incorrectly.	For information on valid LFA commands, see "About Running Log File Analyzer Commands" on page 99.
SBL-LFA-00125 Error parsing configuration file "%s".	The Log File Analyzer configuration file specified in the message text is missing.	Restart the LFA with another configuration file, or make sure the specified configuration file is available.
SBL-LFA-00126 Too many unrelated files are found following main server log file pattern: "%s".	The log files in the server log directory are inconsistent. More than one unrelated file fits the main server log file pattern that is used by the LFA to initialize the server model.	Remove all unrelated files and try again.

Table 37. Resolving Log File Analyzer Problems

System or Error Code and Message	Diagnostic Steps or Cause	Solution
SBL-LFA-00127 Invalid usage of the command.	You have used the command incorrectly.	For information and links to the correct usage of LFA commands, see "About Running Log File Analyzer Commands" on page 99.
SBL-LFA-00128 Component with name "%s" could not be found.	The Log File Analyzer cannot translate the component name you entered into a component short name.	If this is a valid component, specify its short name in the LFA configuration file. For further information, see "Configuring the Log File Analyzer" on page 93.
SBL-LFA-00130 Language "%s" could not be initialized. Please see Log File Analyzer documentation for more information.	The language files in the locale directory on the Siebel Server might be missing or corrupt.	Review information on LFA log file language considerations. For further information, see "About the Log File Analyzer" on page 91.
SBL-LFA-00131 String with code "%s" could not be loaded. Please see Log File Analyzer documentation for more information.	The language files in the locale directory on the Siebel Server might be missing or corrupt.	Review information on LFA log file language considerations. For further information, see "About the Log File Analyzer" on page 91.
SBL-LFA-00132 Formatting string "%s" is not supported. Parameters for this string could not be extracted.	An error in the string makes it impossible for the Log File Analyzer to parse it properly.	If you cannot resolve the underlying issue that caused this error, contact Oracle Global Customer Support by creating a service request (SR) on My Oracle Support.

7 Configuring Client-Side Logging

This chapter describes how to configure client-side logging for Siebel CRM. It also describes the log file information that you can use to diagnose and troubleshoot session and browser issues for Siebel Web Clients. It includes the following topics:

For more information on the following:

- **General high-interactivity features.** See *Siebel System Administration Guide*.
- **Siebel Server and component logging.** See Chapter 4, "Configuring Siebel Server and Component Logging."
- **Client-side logging for Mobile Web Clients.** See *Siebel Remote and Replication Manager Administration Guide*.

NOTE: Only Siebel Web Client applications running in high interactivity and Siebel Open UI support client-side logging. For applications running in standard interactivity, client-side logging is not supported. For more information about standard and high interactivity, see *Siebel System Administration Guide*. or more information about Siebel Open UI, see *Configuring Siebel Open UI*.

About Client-Side Logging

Client-side logging allows you to diagnose and troubleshoot session and browser issues for Siebel Web Clients.

The options for deploying Siebel Web Clients are:

- Siebel Open UI
- High interactivity

- Standard interactivity

Siebel employee applications are typically deployed in high interactivity or Siebel Open UI, whereas customer-facing applications are not. For more information about standard and high interactivity, see *Siebel System Administration Guide*. For more information about Siebel Open UI, see *Configuring Siebel Open UI*.

Client-side logging is supported only for client computers running Microsoft Windows and Internet Explorer (IE). UNIX is not supported. For more information about which versions are supported, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

NOTE: For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certification tab on My Oracle Support. For information about the Certification application, see article 1492194.1 (Article ID) on My Oracle Support.

Client-side logging uses the Siebel event logging system to collect data and write the information to a text log file. The log file resides in the `C:\SiebelLogs` directory on the computer running the application.

Client-side logging allows you to:

- Capture browser activity data for troubleshooting, such as when a Siebel Web Client stops responding or fails
- Log individual user or global session information for a specific Siebel Server
- Debug the source code using JavaScript
- Trace the sequences of operations

For examples of client-side logging files, see "Examples of Log Files for Client-Side Logging" on page 134.

Use the client-side logging feature as a complement to other system management tools.

How Client-Side Logging Works

When a Siebel Web Client starts, the client-side environment variables, if specified, are read from the client, and then the server-side parameters are read. The log engine then initializes and generates the retrieved log file data to the C:\SiebelLogs directory as shown in Figure 1.

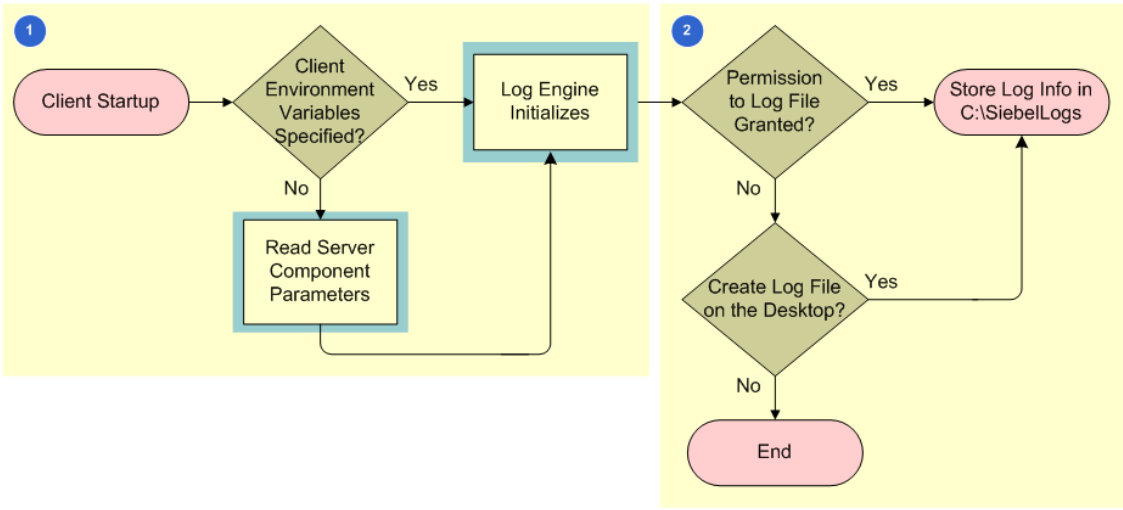

Figure 1. How Client-Side Logging Works

Figure 1 illustrates the following flow:

1. When the client starts up, the client-side environment variables are read from the client and the log engine initializes in one of two ways:
   - If no client environment variables are specified, the log engine initializes after reading the server component parameter values.
   - If there are no server component parameter values specified, the log engine initializes directly from the client.

   NOTE: Client-side settings take precedence over server-side settings.
2. The log file stores the data retrieved in the C:\SiebelLogs directory. However, if a user does not have permission for the C:\SiebelLogs directory, the log file creation fails, and logging is terminated.

Also, consider the following with regard to client-side logging (not shown in Figure 1):

- If the disk is full, logging is terminated, and the log file is not created.
- The minimum log file size is 50 megabytes (MB).

For more information about the following:

- The log file size and specifying the server component parameters, see "Configuring Server Component Parameters for Client-Side Logging" on page 131.

- Specifying user environment variables, see "Configuring Client User Environment Variables for Client-Side Logging" on page 133.

About SiebelLogs Log Files

Log files for Siebel Business Applications record data for individual user or global session information for a specific Siebel Server and capture browser activity data that you can use for troubleshooting. The Siebel application stores these client-side logging files in a separate `C:\SiebelLogs` directory for each session.

The naming convention for client-side log files is:

SiebelCL.<*session_id*><*number*>.log

where:

- *session_id* is the unique session number created for the user
- *number* is an incremental integer that is dependent on a preset log file size

For example, a resulting log filename might be:

```
SiebelCL.ynFc7ujPpnG4.N.8tbIiLKBa7t3fSDpG-1GNdBJgNZw_.log
```

where:

- ynFc7ujPpnG4.N.8tbIiLKBa7t3fSDpG-1GNdBJgNZw is the *session_id.*
- _ is an incremental integer.

If the size of a log file exceeds the preset log file size (as specified either at the server or client level), a new log file is created by adding an incremental integer to the log filename as follows:

```
SiebelCL.J1CWMStoyHrjkydKbJ2JmX2Zf32YnZa2Ep8wBE5i.jo_.03
SiebelCL.J1CWMStoyHrjkydKbJ2JmX2Zf32YnZa2Ep8wBE5i.jo_.02
SiebelCL.J1CWMStoyHrjkydKbJ2JmX2Zf32YnZa2Ep8wBE5i.jo_.01
```

where:

- The file with the 03 suffix is the most current.
- The file with the 01 suffix is the oldest.

For more information about:

- Setting the log file size for a Siebel Server, see "Configuring Server Component Parameters for Client-Side Logging" on page 131
- Setting the log file size for a Siebel Web Client, see "Configuring Client User Environment Variables for Client-Side Logging" on page 133

Each SiebelLogs log file consists of a log file header and a log file detail as shown in Figure 2.

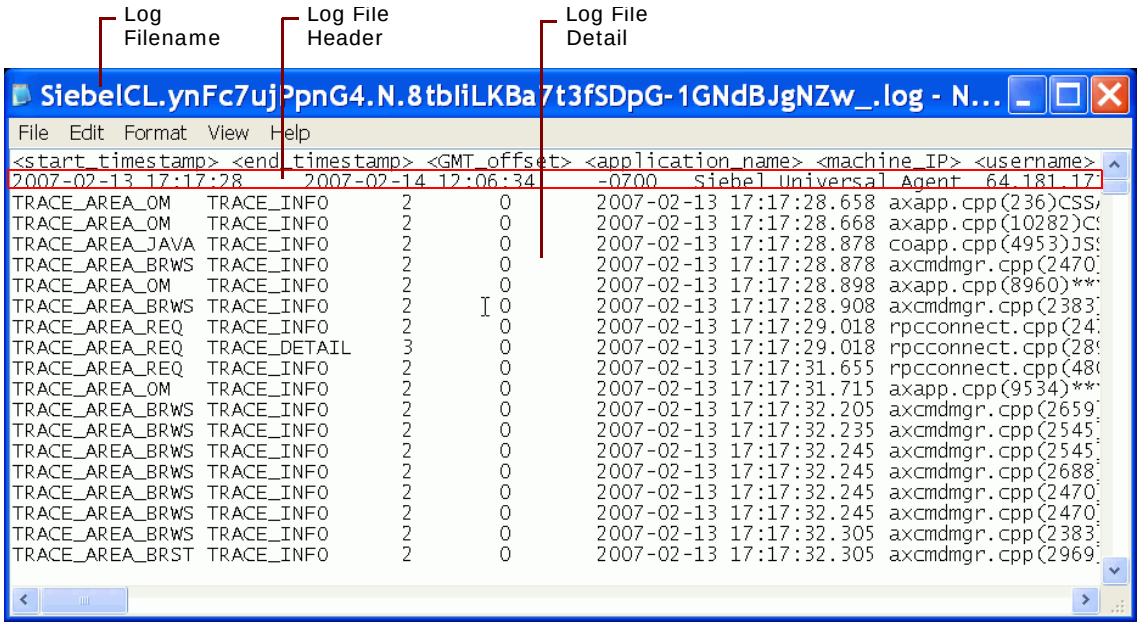

Figure 2. Sample SiebelLogs Log File

The SiebelLogs Log File Header

The format of the SiebelLogs log file header is:

<start_timestamp> <end_timestamp> <GMT_offset> <application_name> <computer_IP> <username> <session_id> <IE_process_id> <IE_thread_id> <log_file_path> <product_version_build_lang>

A sample log file header (as shown in Figure 2 on page 121) might be:

```
2007-02-13 17:17:282007-02-14 12:06:34-0700Siebel Universal Agent64.181.171.9BCOOK
ynFc7ujPpnG4.N.8tbIiLKBa7t3fSDpG-1GNdBJgNZw_4196 6080
d:\mySiebelLog\SiebelCL.ynFc7ujPpnG4.N.8tbIiLKBa7t3fSDpG-1GNdBJgNZw_.log8.0 [20405....] LANG_INDEPENDENT
```

Table 38 provides sample data and a description of each element in the sample log file header.

Table 38. Sample Data in a SiebelLogs Log File Header

Header Element	Sample Data	Description
<start_timestamp>	2007-02-13 17:17:28	The time the log file is created in YYYY-MM-DD HH:MM:SS format.
<end_timestamp>	2007-02-14 12:06:34	The time the client session ends and the log file stops being written in YYYY-MM-DD HH:MM:SS format.

Table 38. Sample Data in a SiebelLogs Log File Header

Header Element	Sample Data	Description
<*GMT_offset*>	-0700	Offset of the local time from Greenwich Mean Time (GMT) in the format ±HHMM.
<*application_name*>	Siebel Universal Agent	The application to which this log file refers, in this instance, Siebel Call Center.
<*computer_IP*>	64.181.171.9	The IP address for the client computer.
<*username*>	BCOOK	The name of the user logged in to the application.
<*session_id*>	ynFc7ujPpnG4.N.8tbIiLKBa7t3fSDpG-1GNdBJgNZw_	A unique session number created for the user for this event.
<*IE_process_id*>	4196	The operating system process Id for the Internet Explorer browser hosting the Siebel application. **NOTE:** Client-side logging is supported only for Microsoft Internet Explorer (IE) browsers. For information about which versions of IE are supported, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network. **NOTE:** For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certification tab on My Oracle Support. For information about the Certification application, see article 1492194.1 (Article ID) on My Oracle Support.
<*IE_thread_id*>	6080	The operating system Id for the Internet Explorer browser. **NOTE:** Client-side logging is supported only for Microsoft Internet Explorer (IE) browsers. For information about which versions of IE are supported, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network. **NOTE:** For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certification tab on My Oracle Support. For information about the Certification application, see article 1492194.1 (Article ID) on My Oracle Support.

Table 38. Sample Data in a SiebelLogs Log File Header

Header Element	Sample Data	Description
<log_file_path>	`d:\mySiebelLog\SiebelCL.ynFc7ujPpnG4.N.8tbIiLKBa7t3fSDpG-1GNdBJgNZw_.log`	The directory path of the log file.
<product_version_build_lang>	`8.0 [20405....] LANG_INDEPENDENT`	Product version and language code that the client is running.

SiebelLogs Log File Information

Figure 3 shows elements in the log file for the same sample log file (SiebelCL.ynFc7ujPpnG4.N.8tbIiLKBa7t3fSDpG-1GNdBJgNZw_.log).

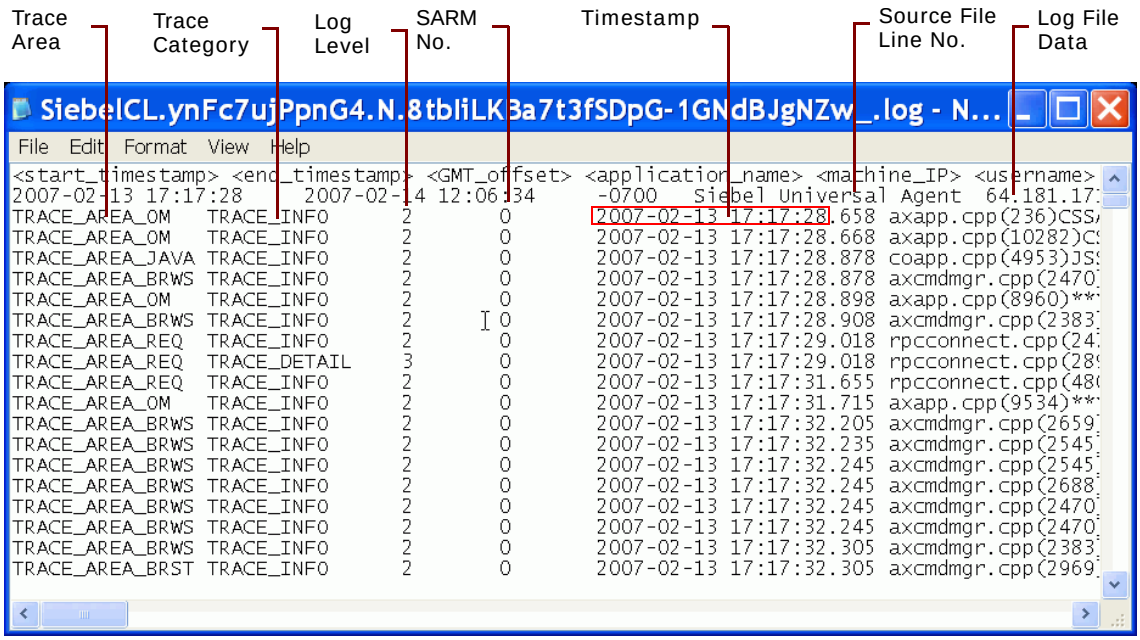

Figure 3. Sample SiebelLogs Log File Detail

The format for the SiebelLogs log file detail is:

<*trace_area*><TAB><*trace_category*><TAB><*numeric_category_level*><TAB>
<*SARM_operation_number*><*timestamp*><TAB><*source_file(line_number)*>
<SPACE><*logged_data*><CRLF>

Table 39 provides sample data and a description of each element in the log file. The term *CRLF* is an abbreviation for a line terminator (carriage return and line feed).

Table 39. Sample SiebelLogs Log File Detail

Detail Element	Sample Data	Description
<*trace_area*>	TRACE_AREA_OM	The functional area of the application writing the log file data.
<*trace_category*>	TRACE_INFO	The textual name given to a log level.
<*numeric_category_level*>	2	The numeric log level value, where the lower values indicate more significant log content. For detailed information about these log levels, see "Trace Categories with Log Levels" on page 126.
<*SARM_operation_number*>	0	The click ID value that is captured by the SARM (Siebel Application Response Management) infrastructure. This value is the same value as the value captured in the component log files in the SARM files. You can use this value to compare between client and other log files.
<*timestamp*>	2007-02-13 17:17:28	The time the log data is written to the log file in YYYY-MM-DD HH:MM:SS format.
<*source_file(line_number)*>	658	The source code filename and line number where the log is initiated.
<*logged_data*>	axapp.cpp(236)CSS...	The information written to the log file that provides contextual information about the application behavior, parameter values, and so on.

About Trace Areas and Trace Categories with Log Levels

Each log entry in a SiebelLogs log file captures a trace area and trace category with the log level. The log entry identifies the functional area of the application writing the log file data and how detailed that data is.

Trace Areas

Table 40 provides a description of each of the trace areas.

Table 40. Trace Areas for Client-Side Logging

Trace Area	Description
TRCAREA_ATL	Intercepted ATL (ActiveX template library) trace statements
TRACEAREA_BRST	Browser state (such as busy)
TRACEAREA_BRWS	Browser operations (such as minimum or maximum)
TRACEAREA_CACHE	Client-side string operation on a cache
TRACEAREA_CLNT	Messages from the client
TRCAREA_CMDMGR	Command manager
TRCAREA_CUSTCTL	Applications, custom controls
TRCAREA_GENCTL	Edit boxes, combination boxes
TRACEAREA_JAVA	Messages from JavaScript
TRACEAREA_LAYOUT	View layout operations
TRCAREA_OM	Area client object manager
TRCAREA_POPUP	Popup
TRACEAREA_REQ	Request sent to server
TRACEAREA_RESP	Response as a property set
TRCAREA_UICOMP	User interface components

Trace Categories with Log Levels

Table 41 provides the various trace categories with the log-level values and a description of each. The higher the log level setting, the more detailed the information that is recorded as well as the larger the log file size. However, higher log-level settings might result in slower application performance.

NOTE: Log levels are set internally in the application and cannot be configured.

Table 41. Trace Categories with Log Level Values for Client-Side Logging

Trace Category	Log Level	Description
TRACE_ERROR	0	Describes the operations that fail.
TRACE_WARNING	1	Describes the operations that do not fail but might in the future if some condition is not changed.
TRACE_INFO	2	Describes the operations of interest to the person reviewing the log file, such as an interesting operation the application has performed.
TRACE_DETAIL	3	Describes the detailed operations performed. This event provides a larger volume of content when compared with the content provided by any of the other trace events.

Viewing SiebelLogs Log Files

This topic describes how to view the SiebelLogs log file and the log file information it provides. Each event within the SiebelLogs log file contains information about client events and the associated server condition, including:

- Log file pathname
- Start and end timestamps
- Application name
- Computer IP address
- Username
- Session Id
- Trace area
- Trace category with log level

The session Id, which makes up part of the log filename, is referenced in messages written to the log file. Locate the appropriate session Id in the log file header and open the specific log that has the session Id in the log filename. Information contained in the log file detail can be used to determine where to search and investigate for further information.

Events are written to and collected in the log file in the order of their occurrence. Each log file contains a header that provides information on the individual log file followed by the actual log detail information as shown in Figure 2 on page 121.

To view a SiebelLogs log file

1. Navigate to the SiebelLogs directory. By default, this is `C:\SiebelLogs`.
2. Using any text editor, open the SiebelLogs log file. Alternatively, you can open this file using Microsoft Excel.

For more information about SiebelLogs log files and to view a sample file, see "About SiebelLogs Log Files" on page 120.

About SiebelLogs Log File Archives

The archive folder for client-side logging holds historic log data up to a preset number of files. You set the maximum number of files to be held in the archive folder in the SEBLCL_LOGARCHIVECOUNT user environment variable. A new directory, `SEBLCL_LOGDIR\logarchive`, is created in the directory where the current log file is being created, and the log files are archived in that directory. The Siebel Web Client checks for any log files to be archived when a user logs in.

The current log file is typically identified as SiebelCL.*session_id*.log.

where:

session_id is the unique session number created for the user.

When this log file reaches a preset file size limit (configured in the SEBLCL_LogFileSize environment variable or in the ClntLogFileSize server component parameter), another log file is created in the same folder (up to the limit specified in the ClntLogArchiveCount server component parameter) with the following convention:

SiebelCL. *session_id*.0n.log, SiebelCL.*session_id*.0(n+1).log

NOTE: Recall that client-side settings take precedence over server-side settings.

For more information about configuring the log archive settings, see "Configuring Server Component Parameters for Client-Side Logging" on page 131 and "Configuring Client User Environment Variables for Client-Side Logging" on page 133.

About Enabling and Disabling Client-Side Logging

When you enable or disable client-side logging for Siebel Business Applications, the settings you configure on the Siebel Web Client and the corresponding Siebel Server control the output of the client. For information about how to enable or disable client-side logging, see "Process of Configuring Client-Side Logging" on page 129.

NOTE: The following subtopics assume that you have enabled the ClientSideLogging server component parameter at the object manager level and have set the tracemode settings appropriately for both the Siebel Web Client and the Siebel Server. For more information, see "Enabling or Disabling Client-Side Logging" on page 129.

How the Server and Client Trace-Mode Settings Affect Client Output

You enable or disable client-side logging by setting the client-side SEBLCL_TRACEMODE environment variable or the server-side ClntTraceMode component parameter for the Siebel application that you are running.

NOTE: The values defined on the client take precedence over the values defined on the server.

When a client is started (as shown in Figure 1 on page 119), the client-side environment variables are read from the client and the log engine initializes in one of two ways:

- If no client environment variables are specified, the log engine initializes after reading the server component parameter values.
- If there are no server component parameter values specified, the log engine initializes directly from the client.

Table 42 shows how the trace mode settings on the client and on the server affect client output. A value of 1 enables high-interactivity mode, whereas a value of 0 disables.

Table 42. How the Server and Client Trace Mode Settings Affect Client Output

Value on Client (SEBLCL_TRACEMODE environment variable)	Value on Server (ClntTraceMode parameter)	Client Output
Not applicable	0	No log file
Not applicable	1	Log file for all users
0	0	No log file
0	1	No log file for the current user
1	1	Log file for the current user and all users
1	0	Log file for the current user only

For information about setting the trace mode settings, see the following:

- “Configuring Server Component Parameters for Client-Side Logging” on page 131
- “Configuring Client User Environment Variables for Client-Side Logging” on page 133

About Enabling Client-Side Logging for a Siebel Server and a Siebel Web Client

If you want to enable client-side logging for a Siebel Server, you define the server component parameters, but you do not define the client environment user variables. Conversely, if you want to enable client-side logging for one or more users on a Siebel Web Client, you define the client environment user variables, but you do not define the server component parameters. Table 43 shows how to enable client-side logging for a Siebel Server compared to enabling client-side logging for one or more users on a Siebel Web Client.

NOTE: Client-side settings take precedence over server-side settings.

Table 43. How to Enable Client-Side Logging for a Siebel Server or for One or More Users on a Siebel Web Client

To Enable Client-Side Logging for...	Define Server Component Parameters	Define Client User Environment Variables
A Siebel Server	Yes	No
One or more users on a Siebel Web Client	No	Yes

NOTE: For the settings in Table 43 to take affect, you must first set the Value on Restart field for the ClntTraceMode server component parameter. For information on setting this parameter, see "Enabling or Disabling Client-Side Logging" on page 129.

Process of Configuring Client-Side Logging

To set up client-side logging, perform the following tasks:

1. "Enabling or Disabling Client-Side Logging" on page 129
2. "Configuring Server Component Parameters for Client-Side Logging" on page 131
3. "Configuring Client User Environment Variables for Client-Side Logging" on page 133

It is recommended that the first time you configure client-side logging that you follow the steps in this process in the order provided. Thereafter, you can perform a single step or multiple steps at any time.

Enabling or Disabling Client-Side Logging

By default, client-side logging is not enabled. You enable or disable client-side logging by setting the ClientSideLogging server component parameter on a Siebel Server.

NOTE: For Siebel version 8.0 or higher, you no longer set the server parameter values for the Siebel Web Client in the [SWE] section of the application configuration file.

This task is a step in "Process of Configuring Client-Side Logging" on page 129.

You can enable or disable client-side logging using either the Server Manager GUI or the Server Manager command-line interface program (srvrmgr).

To enable or disable client-side logging using Server Manager GUI

1. Log in to the application for which you want to set client-side logging.
2. Navigate to the Administration - Server Configuration screen, then Server Components view.
3. In the Components list, select the object manager (component) for which you want to enable client-side logging.

   For example, if you want to enable logging for the Call Center application for the English language, select the Call Center Object Manager (ENU) component.
4. Click the Parameters view tab.
5. In the Parameter field in the Component Parameters list, query for ClientSideLogging, and then perform one of the following:
   - If you want to enable client-side logging, set the Value on Restart field to True.
   - If you want to disable client-side logging, set the Value on Restart field to False.

To enable or disable client-side logging using srvrmgr

1. Log in to the application for which you want to turn on client-side logging.
2. From the Server Manager command-line interface, type the following:

   ```
   change param clientsidelogging = TRUE for comp compname server servername
   ```

   where:

   - *compname* is the name of the component
   - *servername* is the name of the Siebel Server you want to enable

For example, if you want to enable client-side logging for Siebel Call Center for the English language on the Siebel Server named sdchs20n518, enter the following command:

```
change param clientsidelogging = True for comp SCCObjMgr_enu server sdchs20n518
```

Next, you configure the trace-mode settings and other parameters for the Siebel Server and the Siebel Web Client. For information about this configuration, see:

- “Configuring Server Component Parameters for Client-Side Logging” on page 131
- “Configuring Client User Environment Variables for Client-Side Logging” on page 133

Configuring Server Component Parameters for Client-Side Logging

You can enable the client-side logging feature for any server component. This topic describes how to configure the server-side parameters. For information about configuring the client-side environment variables, see "Configuring Client User Environment Variables for Client-Side Logging" on page 133.

This task is a step in "Process of Configuring Client-Side Logging" on page 129.

You can configure the server component parameters for client-side logging applications using either the Server Manager GUI or the Server Manager command-line interface program (srvrmgr).

NOTE: You must first enable the ClientSideLogging server component parameter before performing the following procedure. For more information, see "Enabling or Disabling Client-Side Logging" on page 129.

To configure server component parameters for client-side logging using Server Manager GUI

1. Log in to the application for which you want to set client-side logging.
2. Navigate to the Administration - Server Configuration screen, then Server Components view.
3. In the Components list, select the object manager (component) for which you want to enable client-side logging.

   For example, if you want to enable logging for the Call Center application for the English language, select Call Center Object Manager (ENU).
4. Click the Parameters view tab and in the Parameter field, query for *Clnt**.

5 For each server component parameter, set the Value on Restart field appropriately:

a For the ClntTraceMode parameter, set the Value on Restart field = 1.

b Set the other parameters as necessary.

NOTE: You must set the ClntTraceMode parameter to enable or disable client-side logging for a Siebel Server. The other parameters automatically set to their default values, which means you change them only as needed.

Parameter	Description	Possible Values	Default Value
ClntTraceMode	Set this parameter to enable or disable logging.	1 - Enable 0 - Disable	0
ClntLogFileSize	Size of the log file in megabytes (MB). Minimum log file size is 50 MB. **NOTE:** If you attempt to enter a value less than 50 MB, the log size resets to 50.	50 MB or more	50
ClntLogArchiveCount	Number of files to be archived.	2, 5, 10, and so on.	5
ClntLogDirectory	Directory where the log file is created.	D:\SiebelLogs	C:\SiebelLogs
ClntTraceUnicode	Set this parameter to turn Unicode trace on or off.	Y indicates Unicode trace is on N indicates Unicode trace is off	N

6 Perform one of the following:

- Restart the server component
- Restart the Siebel Server

For more information about restarting the Siebel Server and server components, see *Siebel System Administration Guide*.

To enable client-side logging using srvrmgr

1 Log in to the application for which you want to turn on client-side logging.

2 From the Server Manager command-line interface, type the following:

```
change param ClntTraceMode = 1 for compdef component
```

where *component* is the server component that you want to enable.

For example, if you want to enable client-side logging for Siebel Call Center, enter the following command:

```
change param ClntTraceMode = 1 for compdef SCCObjMgr_enu
```

To configure the client-side settings, see "Configuring Client User Environment Variables for Client-Side Logging" on page 133.

Configuring Client User Environment Variables for Client-Side Logging

This topic describes how to configure the client-side environment variables for client-side logging. For information about configuring server-side component parameters, see "Configuring Server Component Parameters for Client-Side Logging" on page 131.

NOTE: For Siebel version 8.0 or higher, you no longer enter values for the Siebel Web Client in the [SWE] section of application configuration file.

This task is a step in "Process of Configuring Client-Side Logging" on page 129.

To configure client user environment variables for client-side logging for a Siebel Web Client, you must set the SEBLCL_TRACEMODE user environment variable = 1. Setting the other environment variables is optional.

NOTE: You must first enable the ClientSideLogging server component parameter before performing the following procedure. For more information, see "Enabling or Disabling Client-Side Logging" on page 129.

To configure client user environment variables for client-side logging

1. Right-click My Computer on your desktop, and then select Properties.
2. In the System Properties window, click the Advanced tab, and then select Environment Variables.
3. Create and set new environment variables appropriately as shown in the following table:
   a. For the SEBLCL_TRACEMODE environment variable, set the value to 1.
   b. Set the other parameters as necessary.

   NOTE: You must set the following user environment variables to enable or disable client-side logging for a Siebel Web Client. Settings on the client override the settings on the server.

Environment Variable	Possible Values
SEBLCL_TRACEMODE	1—Enable 0—Disable
SEBLCL_LOGDIR	Directory where the log file is created

Environment Variable	Possible Values
SEBLCL_LOGFILESIZE	Size of the log file in megabytes (MB) **NOTE:** If a log file size exceeds the value in the SEBLCL_LOGFILESIZE environment variable, a new log file is created.
SEBLCL_LOGARCHIVECOUNT	Maximum number of files to be archived
SEBLCL_TRACEUNICODE	Y—Unicode trace is on N—Unicode trace is off

4 After setting the user environment variables, click OK.

To configure the server-side settings, see "Configuring Server Component Parameters for Client-Side Logging" on page 131.

Examples of Log Files for Client-Side Logging

This topic provides excerpts of sample log files and examples of log file information for client-side logging for Siebel Business Applications. For more information about the elements in these log files, see "About SiebelLogs Log Files" on page 120. To view a sample log file, see Figure 3 on page 123.

Example of a Log File When a Client Stops Responding

The following log file information is an example of what is written to the SiebelLogs log file when a client stops responding. For a detailed description of this sample output, see Table 44. This trace event has a category of TRACE_ERROR with a log level value of 0.

```
TRACE_AREA_OM   TRACE_ERROR   0   14 2007-04-12 14:40:07.738
axapp.cpp(4028)CSSAxApp::CallRPC - User session has timed out.
```

Table 44. Sample Log File When a Client Stops Responding

Log Detail	Description
TRACE_AREA_OM	Event trace area
TRACE_ERROR	Event trace category
0	Event category level
14	SARM ID
2007-04-12 14:40:07.738	Date and time of log file
axapp.cpp(4028)CSSAxApp::CallRPC - User session has timed out.	Log file message

Example of a Log File When a Required Field on an Applet Is Not Provided

The following log file is an example of the information that is written to the SiebelLogs log file when a user does not enter a value in the required field on an applet. For a detailed description of this sample output, see Table 45. This trace event has a category of TRACE_INFO with a log level value of 2.

```
TRACE_AREA_OM   TRACE_INFO    2  17 2007-04-12 15:14:45.946
axobjectbase.cpp(431)&#039;Name&#039; is a required field.  Please enter a value for
the field.(SBL-DAT-00498)
```

Table 45. Sample Log File When a Required Field on an Applet Is Not Provided

Log Detail	Description
TRACE_AREA_OM	Event trace area
TRACE_INFO	Event trace category
2	Event category level
17	SARM ID
2007-04-12 15:14:45.946	Date and time of log file
axobjectbase.cpp(431)'Name' is a required field. Please enter a value for the field.(SBL-DAT-00498)	Log file message

Example of a Log File When an eScript Business Service Is Called on a View

The following log file information is an example of what is written to the SiebelLogs log file when an eScript business service is called on a view. For a detailed description of this sample output, see Table 46. This trace event has a category of TRACE_DETAIL with a log level value of 3.

```
TRACE_AREA_CACHE  TRACE_DETAIL 3   10   2007-04-12 15:14:00.822
axapp.cpp(2214)Queried view layout cache for view Call_JS_Acc
```

Table 46. Sample Log File When an eScript Business Service Is Called on a View

Log Detail	Description
TRACE_AREA_CACHE	Event trace area
TRACE_DETAIL	Event trace category
3	Event category level
10	SARM ID
2007-04-12 15:14:00.822	Date and time of log file
axapp.cpp(2214)Queried view layout cache for view Call_JS_Acc	Log file message

Example of a Trace Event Log Sequence

The following log file is an example of the information written to the SiebelLogs log file to show the state of an application before it goes into a busy state, thus allowing you to track the sequence of events and diagnose any errors:

```
TRACE_AREA_OM  TRACE_INFO      2  0  2007-04-12 14:21:18.514 axapp.cpp(9534)***
CSSAxApp::ProcessObjectInfo() ends! ***

TRACE_AREA_BRWS   TRACE_INFO   2   0 2007-04-12 14:21:19.706
axcmdmgr.cpp(2470)Browser DocumentComplete is called

TRACE_AREA_BRWS   TRACE_INFO   2   0  2007-04-12 14:21:19.716
axcmdmgr.cpp(2659)Browser DownloadBegin is called

TRACE_AREA_BRWS   TRACE_INFO   2   0  2007-04-12 14:21:19.856
axcmdmgr.cpp(2545)Browser NavigateComplete2 is called

TRACE_AREA_BRWS   TRACE_INFO   2  0  2007-04-12 14:21:20.047
axcmdmgr.cpp(2383)Browser BeforeNavigate2 is called with URL http://sdchs21n110/
sales_enu/
start.swe?SWENeedContext=false&SWECmd=GetCachedFrame&SWEACn=1912&SWEC=3&SWEFrame=t
op._sweclient._sweappmenu&SWEBID=-1&SWETS=

TRACE_AREA_BRST   TRACE_INFO   2  0   2007-04-12 14:21:20.047
axcmdmgr.cpp(2969)Entering Busy state
```

Example of a Control Log File

The following log file displays the information that is written to the SiebelLogs log file when a control is accessed by way of the user interface. In this example, a TRACE_AREA_GENCTL event is created when a user right-clicks using a mouse. For a detailed description of this sample output, see Table 47. This trace event has a category of TRACE_INFO with a log level value of 3.

```
TRACE_AREA_GENCTL TRACE_INFO 3 0 2004-06-10 16:28:01.121 listctrl.cpp(1957): User
right-clicked.
```

Table 47. Sample Control Log File

Log Detail	Description
TRACE_AREA_GENCTL	Event trace area
TRACE_INFO	Event trace category
3	Event category level
0	SARM ID
2004-06-10 16:28:01.121	Date and time of log file
listctrl.cpp(1957): User right-clicked	Log file message

Example of Client Startup Log File

The following log file displays the information that is written to the SiebelLogs log file when a client starts up. In this example, two TRACE_AREA_REQ trace areas are created: one for the start of operation, and one for the end of operation. For a detailed description of this sample output, see Table 48. These trace events have a category of TRACE_INFO with a log level value of 3.

Start of operations

The following indicates the start of operations:

```
TRACE_AREA_REQ TRACE_INFO 3 0 2004-06-10 16:28:01.123 axappp.cpp(9231): Begin
sending RPC
```

NOTE: RPC stands for remote procedure call. This call enhances performance between the application server and the client. RPC is supported only for applications running in high-interactivity mode.

End of operations

The following indicates the end of operations:

```
TRACE_AREA_REQ TRACE_INFO 3 0 2004-06-10 16:28:05.234 axappp.cpp(9232): Completed
RPC request.
```

Table 48. Sample Client Startup Log File

Log Detail	Description
TRACE_AREA_REQ	Event trace area
TRACE_INFO	Event trace category
3	Event category level
0	SARM ID
2004-06-10 16:28:05.234	Date and time of log
axappp.cpp(9232): Completed RPC request.	Log message

Example of a GET Request Log File

The following log file information is an example of a get request to the server and the information that is written to the SiebelLogs log file when a user navigates from one view to another within an application. For a detailed description of this sample output, see Table 49. This trace event has a category of TRACE_DETAIL with a log level value of 4:

```
TRACE_AREA_REQ TRACE_DETAIL 4 0 2004-06-10 16:28:01:235 axapp.cpp(9236):[155]GET
query string: SWECmd=GetCachedFrame&SWEACn=5114&SWEC=3&SWEFrame= top._swe.
```

Table 49. Sample Get Request Log File

Log Detail	Description
TRACE_AREA_REQ	Event trace area
TRACE_DETAIL	Event trace category
4	Event category level
0	SARM ID
2004-06-10 16:28:01:235	Date and
axapp.cpp(9236):[155]GET query string: SWECmd=GetCachedFrame&SWEACn=5114&SWEC=3&SWEFrame= top._swe	Log file message

Example of a POST Request Log File

The following log file is an example of the information that is written to the SiebelLogs log file when a post request is sent to a server. In this example, a TRACE_AREA_REQ trace area is created for the post request. For a detailed description of this sample output, see Table 50. This trace event has a category of TRACE_DETAIL with a log level value of 4.

```
TRACE_AREA_REQ TRACE_DETAIL 4 0 2004-06-10 16:28:01:235 axapp.cpp(9236):[155] POST:
data=field1=<data>&field2=<data>...
```

Table 50. Sample Post Request Log File

Log Detail	Description
TRACE_AREA_REQ	Event trace area
TRACE_DETAIL	Event trace category
4	Event category level
0	SARM ID
2004-06-10 16:28:01:235	Date and time of log file
axapp.cpp(9236):[155] POST:data=field1=<data>&field2=<data>…	Log file message

Example of a String Cache Operations Log File

The following log file displays the information that is written to the SiebelLogs log file when view layout caching is complete. In this example, a TRACE_AREA_CACHE trace area is created for the view layout cache. For a detailed description of this sample output, see Table 51. This trace event has a category of TRACE_DETAIL with a log level value of 4:

```
TRACE_AREA_CACHE TRACE_DETAIL 4 0 2004-06-10 16:28:01:435 axapp.cpp(9252): Queried
view for layout cache for view Order Entry List View
```

Table 51. Sample String Cache Operations Log File

Log Detail	Description
TRACE_AREA_CACHE	Event trace area
TRACE_DETAIL	Event trace category
4	Event category level
0	SARM ID
2004-06-10 16:28:01:435	Date and time of log file
axapp.cpp(9252): Queried view for layout cache for view Order Entry List View	Log file message

Example of Busy State Log File

The following log file information displays what is written to the SiebelLogs log file when the browser enters a busy state. In this example, a TRACE_AREA_BRST trace area is created for the busy state. For a detailed description of this sample output, see Table 52. This trace event has a category of TRACE_DETAIL with a log level value of 4:

```
TRACE_AREA_BRST TRACE_DETAIL 4 0 2004-06-10 16:28:435 axapp.cpp(5122): Entering busy
state
```

Table 52. Sample Busy State Log File

Log Detail	Description
TRACE_AREA_BRST	Event trace area
TRACE_DETAIL	Event trace category
4	Event category level
0	SARM ID
2004-06-10 16:28:435	Date and time of log file
axapp.cpp(5122): Entering busy state	Log file message

Example of a JavaScript Error Message Log File

The following log file is an example of a message from JavaScript code and the information that is written to the SiebelLogs log file when the object manager fails, a session has timed out, or a request could not be processed. For a detailed description of this sample output, see Table 53. This trace event has a log level value of 1:

```
TRACE_AREA_JAVA 1 0 2004-06-10 16:28:09:435 axapp.cpp(5129):SBL-UIF-00335: We are unable to process your request. (This is most likely because you used the browser BACK or REFRESH button to get to this point)
```

Table 53. Sample JavaScript Error Message Log File

Log Detail	Description
TRACE_AREA_JAVA	Event trace area
1	Event category level
0	SARM ID
2004-06-10 16:28:09:435	Date and time of log file
axapp.cpp(5129):SBL-UIF-00335: We are unable to process your request. (This is most likely because you used the browser BACK or REFRESH button to get to this point)	Log file message

Example of a Browser Operation Log File

The following log file is an example of a browser event handler and the information that is written to the SiebelLogs log file when the browser is maximized, minimized, or closed. For a detailed description of this sample output, see Table 54. This trace event has a category of TRACE_INFO with a log level value of 3.

```
TRACE_AREA_BRWS TRACE_INFO 3 0 2004-06-10 16:28:09:435 axapp.cpp(5102): Browser minimized.
```

Table 54. Sample Browser Operation Log File

Log Detail	Description
TRACE_AREA_BRWS	Event trace area
TRACE_INFO	Event trace category
3	Event category level
0	SARM ID
2004-06-10 16:28:09:435	Date and time of log file
axapp.cpp(5102): Browser minimized.	Log file message

Example of a View Layout Log File

The following log file is an example of the information that is written to the SiebelLogs log file when a view layout is rendered from memory, the hard disk, or the server. For a detailed description of this sample output, see Table 55. This trace event has a category of TRACE_DETAIL with a log level value of 4:

```
TRACE_AREA_LAYOUT TRACE_Detail 4 0 2004-06-10 16:29:22:578 axapp.cpp(1223):
Accessing view layout for Account List View from disk
```

Table 55. Sample View Layout Log File

Log Detail	Description
TRACE_AREA_LAYOUT	Event trace area
TRACE_Detail	Event trace category
4	Event category level
0	SARM ID
2004-06-10 16:29:22:578	Date and time of log file
axapp.cpp(1223): Accessing view layout for Account List View from disk	Log file message

Example of Propertyset Info for GotoView Log File

The following log file displays the information that is written to the SiebelLogs log file when a property set response comes back from the server. In this example, a TRACE_AREA_RESP trace area is created for the property set response. For a detailed description of this sample output, see Table 56. This trace event has a category of TRACE_INFO with a log level value of 3:

```
TRACE_AREA_RESP TRACE_INFO 3 0 2004-06-10 16:28:09:435 axapp.cpp(5102): Propertyset
for gotoview

@0*0*6*3*0*3*0*6*Target35*top._sweclient._swecontent._sweview4*SWEC1*74*View17*Acc
ount List View6*ViewId0*6*Status9*NewLayout2*UC1*10*6*3*api3*0*2*0*2*
sc3*0*3*rqc1*71*v3034*8*Accounts5*Query6*Cancel15*Query

Assistant6*Delete4*Save13*PositionOnRow18*ToggleListRowCount8*
Location4*Site17*Main Phone Number12*Main Phone#14*Account status23*PickList
Account Status9*Home

Page3*URL7*GotoUrl12*SynchAccount10*BlankLine110*Blankline213*OwnerInstance19*Labe
lACCOUNTDETAILS10*LabelAUDIT13* LabelAsterisk13*LabelDUN
```

Table 56. Sample Propertyset Info for GotoView Log File

Log Detail	Description
TRACE_AREA_RESP	Event trace area
TRACE_INFO	Event trace category

Table 56. Sample Propertyset Info for GotoView Log File

Log Detail	Description
3	Event category level
0	SARM ID
2004-06-10 16:28:09:435	Date and time of log
axapp.cpp(5102): Propertyset for gotoview @0*0*6*3*0*3*0*6*Target35*top._sweclient._swecontent._sweview 4*SWEC1*74*View17*Account List View6*ViewId0*6*Status9*NewLayout2*UC1*10*6*3*api3*0*2*0*2 * sc3*0*3*rqc1*71*v3034*8*Accounts5*Query6*Cancel15*Query Assistant6*Delete4*Save13*PositionOnRow18*ToggleListRowCount8* Location4*Site17*Main Phone Number12*Main Phone#14*Account status23*PickList Account Status9*Home Page3*URL7*GotoUrl12*SynchAccount10*BlankLine110*Blankline213 *OwnerInstance19*LabelACCOUNTDETAILS10*LabelAUDIT13* LabelAsterisk13*LabelDUN	Log file message

Example of JavaScript Log File

Logging using JavaScript is supported through an exposed API in the application object. You use the line number and filename provided in the log file to debug the source code.

The log engine is called as follows:

JavaScript => theApplication().SeblTrace("*level*,*logmessage*");

where:

- *level* is 0, 1, 2, or 3 (For more information on log levels, see Table 41 on page 126)
- *logmessage* is the string message

A possible example is:

JavaScript => theApplication().SeblTrace(*3,"JSSApplicationShadow Initialized."*);

where the resultant Java trace log file appears as follows:

```
TRACE_AREA_JAVA TRACE_INFO 4 0 2004-06-10 16:29:09:358 JSSApplicationShadow
Initialized.
```

NOTE: Auto logging is not supported from a browser script.

8 Collecting Siebel Environment Data

This chapter describes how to collect Siebel environment information (such as setup, configuration,and logging information) for diagnostic and troubleshooting purposes using the Siebel Diagnostic Collector (SDDC) command-line utility. This chapter includes the following topics:

About Siebel Diagnostic Data Collector

The Siebel Diagnostic Data Collector (SDDC) is a command-line utility that resides in the binary subdirectory of the Siebel Server, Siebel Gateway Name Server, and Siebel Web Server Extension (SWSE) root directory as the executable siebsnap.exe under Microsoft Windows or as binaries under UNIX. When run, the Siebel Diagnostic Data Collector (SDDC) utility collects information individually for Siebel Servers, the Siebel Gateway Name Server, and the Siebel Web Server Extension. The utility stores the collected data in output files. These files are available for immediate review, or can be sent to Oracle Global Customer Support if required by creating a service request (SR) on My Oracle Support. For information on using SDDC to collect environment data, see "Process of Collecting Siebel Environment Data Using SDDC" on page 145.

SDDC creates output files after each execution. These files document environment information for each specific entity. For details on the location and type of collected information, see "About Reviewing Siebel Environment Data" on page 150.

NOTE: To run an environment data collection, make sure you have all necessary executables or binaries available. For further information, see "About SDDC Executables and Binaries" on page 143.

About SDDC Executables and Binaries

The Siebel Diagnostic Data Collector (SDDC) utility uses the following executables or binaries to collect environment data comprehensively. The SDDC does not require all executables and binaries to run; however, the SDDC collects the most information when all are present.

The executables and binaries are divided based on operating system and platform.

Windows Executables

- odbcsql
- netstat
- db2level (if using db2)
- osql (if using MS SQL)
- sqlplus (if using Oracle)

UNIX Binaries (Common)

The SDDC uses the following 31 binaries on all UNIX platforms.

- /usr/bin/cp
- /usr/bin/ls
- /bin/tar
- /bin/mv
- /bin/compress
- /bin/mkdir
- /bin/rm
- /bin/chmod
- /bin/grep
- /bin/cat
- /bin/find
- /bin/touch
- /bin/echo
- /bin/sum
- /bin/wc
- /bin/head
- /bin/coreadm
- /bin/sed
- /bin/awk
- /bin/date
- /bin/hostname
- /bin/uname
- /bin/netstat
- /etc/system
- /usr/sbin/ndd
- /dev/tcp
- db2level
- /usr/bin/ipcs
- db2
- sqlplus
- what

UNIX Binaries for Oracle Solaris

- psrinfo
- sysdef
- prtconf
- ifconfig
- CC
- /bin/isainfo
- /bin/ulimit -a
- /sbin/prtdiag

UNIX Binaries for AIX

- lscfg
- instfix
- lsattr
- lsps
- lsfs
- lspv
- lsvg
- no
- ifconfig
- /bin/oslevel
- /bin/getconf
- /bin/lslpp
- /etc/security/limits
- /bin/errpt
- /etc/inittab

UNIX Binaries for HP-UX

- sysdef
- aCC
- swlist
- ioscan
- /bin/getconf
- /usr/lib/libCsup2
- /etc/system

UNIX Binaries for Linux

- gcc
- getconf
- /sbin/ifconfig
- /proc/cpuinfo
- /sbin/sysctl

Process of Collecting Siebel Environment Data Using SDDC

You run Siebel Diagnostic Data Collector (SDDC) to collect environment setup, configuration settings, and logging information about the system. For more information about SDDC, see "About Siebel Diagnostic Data Collector" on page 143.

Run SDDC separately for the Siebel Servers, the Siebel Gateway Name Server, and the Siebel Web Server Extension (SWSE) to collect information specific to that entity. The kinds of tasks you can perform are similar for both Microsoft Windows and UNIX, but there is additional preparation for UNIX before you can begin collecting data.

To collect Siebel environment data using SDDC, perform the following tasks:

1. (UNIX only) Do "Preparing the UNIX Environment to Run SDDC" on page 145.
2. Do one of the following:

- Run SDDC to collect data for the Siebel Server.

  For more information, see "Running SDDC to Collect Siebel Server Data" on page 146.

- Run SDDC to collect data for the Siebel Gateway Name Server.

  For more information, see "Running SDDC to Collect Siebel Gateway Name Server Data" on page 146.

- Run SDDC to collect data for the Siebel Web Server and SWSE.

  For more information, see "Running SDDC to Collect Siebel Web Server and SWSE Data" on page 147.

Preparing the UNIX Environment to Run SDDC

Perform the following procedure to prepare the UNIX environment to run Siebel Diagnostic Data Collector (SDDC).

To prepare the UNIX environment to run SDDC

1. Run a database-specific script to set database environment variables.
2. Run the `siebenv.sh` or `siebenv.csh` scripts to set Siebel environment variables.

   For more information on these scripts, see *Siebel Installation Guide for UNIX*.

3. Change the permissions to execute SDDC.

You are now ready to collect data using SDDC using one of the following procedures:

- "Running SDDC to Collect Siebel Server Data" on page 146
- "Running SDDC to Collect Siebel Gateway Name Server Data" on page 146
- "Running SDDC to Collect Siebel Web Server and SWSE Data" on page 147

Running SDDC to Collect Siebel Server Data

This topic provides procedures for collecting Siebel Server data using Siebel Diagnostic Data Collector (SDDC) for Microsoft Windows and UNIX.

Running SDDC to Collect Siebel Server Data for Microsoft Windows

Use the following procedure to collect Siebel Server data for Microsoft Windows.

To run SDDC to collect Siebel Server data for Microsoft Windows

1. Navigate to the binary subdirectory within the Siebel Server root directory.
2. Run `siebsnap.exe` using the `/s` flag and, as necessary, the parameters listed in Table 57 on page 148 as shown in the following example:

   ```
   siebsnap.exe /s
   ```

3. Review the collected information in the `siebsnap` output directory, which is created by the SDDC utility under the *SIEBSRVR_ROOT* directory.

Running SDDC to Collect Siebel Server Data for UNIX

Use the following procedure to collect Siebel Server data for UNIX.

To run SDDC to collect Siebel Server data for UNIX

1. Do "Preparing the UNIX Environment to Run SDDC" on page 145.
2. Enter the siebsnap command using the `-s` flag and, as necessary, parameters listed in Table 58 on page 149 as shown in the following example:

   ```
   siebsnap -s siebel_server_name
   ```

3. Review the collected information in the `siebsrvr_computer-name_server-name` output directory.

Running SDDC to Collect Siebel Gateway Name Server Data

This topic provides procedures for collecting Siebel Gateway Name Server data using Siebel Diagnostic Data Collector (SDDC) for Microsoft Windows and UNIX.

Running SDDC to Collect Siebel Gateway Name Server Data for Microsoft Windows

Use the following procedure to collect Siebel Gateway Name Server data for Microsoft Windows.

To run SDDC to collect Siebel Gateway Name Server data for Microsoft Windows

1. Navigate to the binary subdirectory within the Siebel Gateway Name Server root directory.
2. Run `siebsnap.exe` using the `/g` flag and, as necessary, the parameters listed in Table 57 on page 148 as shown in the following example:

   ```
   siebsnap.exe /g
   ```

3. Review the collected information in the `siebsnap` output directory, which is created by the SDDC utility under the `gtwysrvr` directory.

Running SDDC to Collect Siebel Gateway Name Server Data for UNIX

Use the following procedure to collect Siebel Gateway Name Server data for UNIX.

To run SDDC to collect Siebel Gateway Name Server data for UNIX

1. Do "Preparing the UNIX Environment to Run SDDC" on page 145.
2. Enter the `siebsnap` command using the `-g` flag and, as necessary, the parameters listed in Table 58 on page 149 as shown in the following example:

   ```
   siebsnap -g siebel_gateway_name
   ```

3. Review the collected information in the *computer-name*_gateway output director.

Running SDDC to Collect Siebel Web Server and SWSE Data

This topic provides procedures for collecting Siebel Web Server and Siebel Web Server Extension (SWSE) data using Siebel Diagnostic Data Collector (SDDC) for Microsoft Windows and UNIX.

Running SDDC to Collect Siebel Web Server and SWSE Data for Microsoft Windows

Use the following procedure to collect Siebel Web Server and SWSE data for Microsoft Windows.

Table 57 shows the parameters available for use in collecting Siebel Server and SWSE data for Microsoft Windows.

Table 57. SDDC Parameters under Microsoft Windows

Parameter	Description	Required?
/g	Append this parameter to the siebsnap.exe command to collect information on the Siebel Gateway Name Server.	Yes
/s	Append this parameter to the siebsnap.exe command to collect information on the Siebel Server.	Yes
/w	Append this parameter to the siebsnap.exe command to collect information on the Web server and SWSE.	Yes
/c siebsnap.cfg	Include this parameter to reference a particular configuration file. Use this parameter if Oracle Global Customer Support provides a configuration file. For more information, see "Configuring SDDC Content Under Microsoft Windows" on page 155.	No
/h	Use this parameter with the siebsnap.exe command to list information on SDDC and its parameters.	No
/l	Indicates the preferred language.	No

NOTE: Use only one of the /g, /s, and /w parameters during a single SDDC execution.

To run SDDC to collect Siebel Web Server and SWSE data for Microsoft Windows

1. Navigate to the binary subdirectory within the Siebel *SWSE_ROOT* directory.
2. Run siebsnap.exe using the /w flag and, as necessary, parameters listed in Table 57 on page 148 as shown in the following example:

   ```
   siebsnap.exe /w
   ```

3. Review the collected information in the siebsnap output directory, which is created by the SDDC utility under the *SWSE_ROOT* output directory.

Running SDDC to Collect Siebel Web Server and SWSE Data for UNIX

This topic provides procedures for collecting Siebel Web Server and Siebel Web Server Extension (SWSE) data using Siebel Diagnostic Data Collector (SDDC) for UNIX.

The parameters available for use in collecting Siebel Server and SWSE data for UNIX are provided in Table 58.

Table 58. SDDC Parameters for UNIX

Parameter	Description
-g *siebel_gateway_name*	Append the -g parameter with the name of the Siebel Gateway Name Server to collect information on the Siebel Gateway Name Server. Alternatively, use -g this_server.
-s *siebel_server_name*	Append the -s parameter with the name of the Siebel Server to collect information on a Siebel Server. Alternatively, use -s this_server.
-w *webserver_root*	Append the -w parameter with the path of the Web server root to collect information on the SWSE and Web server. Alternatively, use -w this_server.
-c siebsnap.ini	Include this parameter to reference a particular configuration INI file. For more details, see "Configuring SDDC Content Under UNIX" on page 160.
-help	Use this parameter with the siebsnap command to list information on SDDC and its parameters.

To run SDDC to collect Siebel Web Server and SWSE data for UNIX

1. Do "Preparing the UNIX Environment to Run SDDC" on page 145.
2. Enter the siebsnap command using the -w flag and, as necessary, parameters listed in Table 58 on page 149 as shown in the following example:

   ```
   siebsnap -w webserver_root
   ```

3. Review the collected information in the *computer-name_webserver-name* output directory.

NOTE: Alternatively, use this_server in place of the Siebel Gateway Name Server name, Siebel Server name, or the Web Server name when using SDDC under UNIX.

Examples of SDDC Commands

This topic provides examples of Siebel Diagnostic Data Collector (SDDC) commands for Microsoft Windows and UNIX.

Examples of SDDC Commands for Microsoft Windows

Some examples of Microsoft Windows SDDC commands are:

- `siebsnap.exe /c siebsnapw32.cfg -g`

  This command retrieves Siebel Gateway Name Server information using a configuration file named siebsnapw32.cfg.

- `siebsnap.exe /s`

  This command retrieves Siebel Server information.

- `siebsnap.exe /c siebsnapw32.cfg /w`

  This command retrieves Web server and SWSE information using a configuration file named siebsnapw32.cfg.

Examples of SDDC Commands for UNIX

Some samples of UNIX SDDC commands are:

- `siebsnap -s this_server -u sadmin -p sadmin`

  This command retrieves Siebel Server information using a username and password.

- `siebsnap -g gtway1`

  This command retrieves Siebel Gateway Name Server information with a Siebel Gateway Name Server name of gtway1.

- `siebsnap -w this_server`

  This command retrieves Web server and SWSE information.

About Reviewing Siebel Environment Data

The Siebel Diagnostic Data Collector (SDDC) utility creates output files and directories, as necessary, after each execution of the utility. Manually access these files to review the Siebel environment data, or send the output files to Oracle Global Customer Support for review by creating a service request (SR) on My Oracle Support.

The output files document the environmental setup information, application configurations, and log files if specified. For further information on collecting data using SDDC, see “Process of Collecting Siebel Environment Data Using SDDC” on page 145.

About SDDC Output

The SDDC Microsoft Windows utility creates output in the format of a root directory with additional subdirectories and files. For details on SDDC Microsoft Windows output file information and locations, see "SDDC Output Under Microsoft Windows" on page 152.

The SDDC UNIX utility creates output in the format of compressed files. For details on SDDC UNIX output file information and locations, see "SDDC Output Under UNIX" on page 154.

SDDC uses the following naming convention for the creation of the root directory and filenames:

ss_{GS|SS|WS}*yyyy-mm-dd_hh_mm_ss*

where:

- *ss* is siebsnap.
- *GS*/*SS*/*WS* is the Siebel Gateway Name Server, Siebel Server, or Web server .
- *yyyy-mm-dd* is year, month, and day.
- *hh_mm_ss* is hour, minute, and second based on a 24-hour clock.

For example, the directory or filename ss_SS2003-04-08_17_10_30 represents information collected for a Siebel Server on 8 April at approximately 5:00 P.M. The directory or filename ss_GS2003-04-07_14_18_58 represents information collected for the Siebel Gateway Name Server on 7 April at approximately 2:00 P.M.

Common SDDC Output Files and Folders

The output from a Siebel Diagnostic Data Collector (SDDC) execution for a Siebel Server, the Siebel Gateway Name Server, and Siebel Web Server Extension (SWSE) contains common folders and files. Table 59 provides further descriptions of the information collected in these files and folders.

Table 59. Common Files and Folders

Files and Subfolders	Description
ReadMe file	Provides a snapshot of the files copied and directories created during the SDDC execution.
siebsnap log file	Provides a detailed log file of information collected during the SDDC execution. This file is available for SDDC only under Microsoft Windows.
Configuration file	Copies the configuration file used if one is specified during the SDDC execution. This file is only available for SDDC under Microsoft Windows.
siebel_info directory	Directory for Siebel environment information. This folder contains further subfolders, which contain log files and details on the Siebel environment.
system_info directory	Directory for system information. This folder contains text files containing information on hardware, network statistics, operating system, and registry keys.

Table 59. Common Files and Folders

Files and Subfolders	Description
db_info directory	Directory for database version information. This folder contains text files containing details on the database version.
WebserverLogs directory	Directory for Web server log information. This folder contains log files for the Web server.

SDDC Output Under Microsoft Windows

Siebel Diagnostic Data Collector (SDDC) output under Microsoft Windows consists of files stored within a directory structure created by the utility. If a configuration file is not specified, the default directory for the SDDC output under Microsoft Windows is the siebsnap directory under the Siebel Server root. To configure a different SDDC output location, update the OutputDirectory parameter in the SDDC configuration file. For information on configuring this parameter and other parameters in the SDDC configuration file, see "Configuring SDDC Content Under Microsoft Windows" on page 155.

SDDC creates additional folders within the siebsnap directory (or the configured output directory) based on whether SDDC collects data for a Siebel Server, the Siebel Gateway Name Server, or the Web server and SWSE. For details on the time-sensitive directory naming convention for these root folders, see "About Reviewing Siebel Environment Data" on page 150.

For locations of the output contents produced for these entities, see:

- "Siebel Server SDDC Output Under Microsoft Windows" on page 152
- "Siebel Gateway Name Server SDDC Output Under Microsoft Windows" on page 153
- "Web Server SDDC Output Under Microsoft Windows" on page 153

For descriptions of the files and directory content of the SDDC output, some of which are common between each entity, see "Common SDDC Output Files and Folders" on page 151.

Siebel Server SDDC Output Under Microsoft Windows

With a Siebel Server Siebel Diagnostic Data Collector (SDDC) execution, the utility creates the root Siebel Server output folder, in the format ss_SSyyyy-mm-dd_hh_mm_ss, within the siebsnap directory (or configured output directory). Within this folder, the utility creates a folder of the format, siebesrvr_*server_name*, where *server_name* represents the name of the Siebel Server profiled by the utility. The directory structure and contents appear as follows:

```
ss_SSyyyy-mm-dd_hh_mm_ss\

   siebesrvr_enterprise-name_server-name\

      Readme file

      Siebsnap log file

      Configuration file
```

```
        system_info\
        siebel_info\
        db_info\
```

Siebel Gateway Name Server SDDC Output Under Microsoft Windows

With a Siebel Gateway Name Server Siebel Diagnostic Data Collector (SDDC) execution, the utility creates the root Siebel Gateway Name Server output folder in the format ss_GSyyyy-mm-dd_hh_mm_ss within the siebsnap directory (or configured output directory). Within this folder, the utility creates a folder named gateway, which collects information on the Siebel Gateway Name Server. The directory structure and contents appear as follows:

```
ss_GSyyyy-mm-dd_hh_mm_ss\
    gateway\
        Readme file
        Siebsnap log file
        Configuration file
        system_info\
        siebel_info\
```

Web Server SDDC Output Under Microsoft Windows

With a Web server Siebel Diagnostic Data Collector (SDDC) execution, the utility creates the root Web server output folder in the format ss_WSyyyy-mm-dd_hh_mm_ss within the siebsnap directory (or configured output directory). Within this folder, the utility creates a folder named webserver, which collects information on the Web server and SWSE. The directory structure and contents appear as follows:

```
ss_WSyyyy-mm-dd_hh_mm_ss\
    webserver\
        Readme file
        Siebsnap log file
        Configuration file
        system_info\
        siebel_info\
        WebserverLogs\
```

SDDC Output Under UNIX

Siebel Diagnostic Data Collector (SDDC) output under UNIX consists of files compressed within a directory structure created by the utility. The default directory for the compressed files is the directory from which SDDC is run. To configure a different SDDC output location, use the -o parameter during the SDDC execution. For details on running the SDDC utility under UNIX, see "Process of Collecting Siebel Environment Data Using SDDC" on page 145.

The compressed output files have the extension .tar.z appended to the filename created by SDDC using the SDDC output naming convention. For a description of this naming convention, see "About Reviewing Siebel Environment Data" on page 150. The extensions .logarchive.tar.z, asserts.tar.z, and logarchive_asserts.tar.z also apply based on the log parameters specified during execution.

For descriptions of the output for each entity see:

- "Siebel Server SDDC Collector Output Under UNIX" on page 154
- "Siebel Server SDDC Collector Output Under UNIX" on page 154
- "Web Server SDDC Output Under UNIX" on page 155

For descriptions of the files and directory content of the SDDC output, some of which are common between each entity, see "Common SDDC Output Files and Folders" on page 151.

Siebel Server SDDC Collector Output Under UNIX

With a Siebel Server Siebel Diagnostic Data Collector (SDDC) execution, the utility creates the compressed file in the format ss_SS_yyyy-mm-dd_hh_mm_ss.tar.Z in the default output directory (or configured output directory). The information collected by the SDDC utility varies based on the parameter settings in the siebsnap.ini file. For information on configuring the siebsnap.ini file, see "Configuring SDDC Content Under Microsoft Windows" on page 155.

By default, the Siebel Server SDDC execution collects system information (system_info), database version information (database_info), and Siebel environment information (Siebel_info). For descriptions of the files and directory content, see "Common SDDC Output Files and Folders" on page 151.

Siebel Gateway Name Server SDDC Output Under UNIX

When Siebel Diagnostic Data Collector (SDDC) executes on a Siebel Gateway Name Server, it creates the compressed file in the format ss_GS_yyyy-mm-dd_hh_mm_ss.tar.Z in the default output directory (or configured output directory). The information collected by the SDDC utility varies based on the parameter settings in the siebsnap.ini file. For information on configuring the siebsnap.ini file, see "Configuring SDDC Content Under Microsoft Windows" on page 155.

By default, when SDDC executes on a Siebel Gateway Name Server it collects system information (system_info) and Siebel environment information (Siebel_info). For descriptions of the files and directory content, see "Common SDDC Output Files and Folders" on page 151.

Web Server SDDC Output Under UNIX

With a Siebel Web server Siebel Diagnostic Data Collector (SDDC) execution, the utility creates the compressed file in the format ss_WS_yyyy-mm-dd_hh_mm_ss.tar.Z in the default output directory (or configured output directory). The information collected by the SDDC utility varies based on the parameter settings in the siebsnap.ini file. For information on configuring the siebsnap.ini file, see "Configuring SDDC Content Under Microsoft Windows" on page 155.

By default, when SDDC executes on a Web server it collects system information (system_info), Siebel environment information (Siebel_info), and Web server log information (WebserverLogs). For descriptions of the files and directory content, see "Common SDDC Output Files and Folders" on page 151.

Configuring SDDC Content Under Microsoft Windows

The Microsoft Windows Siebel Diagnostic Data Collector (SDDC) can be configured to modify or enhance the amount of information collected during an SDDC execution. A Microsoft Windows SDDC configuration file is required by SDDC to modify any configurations to the output. The configuration file is referenced during the SDDC execution. By default, a configuration file is not included with the SDDC utility. It is recommended that you contact Oracle Global Customer Support before using configuration files by creating a service request (SR) on My Oracle Support. Oracle Global Customer Support provides configuration files based on the specific information you require. For information on SDDC configurations under UNIX, see "Configuring SDDC Content Under UNIX" on page 160.

The SDDC configuration file is divided into sections that can be used to configure the type of information and log files collected by the utility. Edit the configuration file with a text editor. Table 60 provides the SDDC configuration file parameters.

Table 60. SDDC Configuration File and Parameters

Section	Parameter	Specifies
[Main]	OutputDirectory	Specifies the directory location for the creation of the SDDC directory and output files.
	CollectLog	Specifies whether log files are collected.
	CollectLogArchive	Specifies whether log archive files are collected.
	CollectCrash	Specifies whether failure files are collected.
	CollectStderrFiles	Specifies whether standard error files are collected.
	CollectDump	Specifies whether back-up system files are collected.
	CollectAssert	Specifies whether assert and prefer files are collected.
	SiebelBinDir	Specifies the directory location of the *SIEBSRVR_ROOT* binary folder.

Table 60. SDDC Configuration File and Parameters

Section	Parameter	Specifies
[Registry]	Key01	Specifies a registry key for collection.
	Key02	Specifies a registry key for collection.
	Key03	Specifies a registry key for collection.
[CrashFiles]	StartDate	Specifies the start date for a range of process failure files to collect.
	EndDate	Specifies the end date for a range of process failure files to collect.
	MatchingFiles	Specifies the process failure file extensions to collect. You can specify the collection of Siebel Flight Data Recorder (FDR) files in this section by identifying the extension FDR (for example, `*.fdr`).
[StderrFiles]	StartDate	Specifies the start date for a range of standard error files to collect.
	EndDate	Specifies the end date for a range of standard error files to collect.
	MatchingFiles	Specifies the standard error file extensions to collect.
[ProcessDump]	StartDate	Specifies the start date for a range of dump files to collect.
	EndDate	Specifies the end date for a range of dump files to collect.
	MatchingFiles	Specifies the dump file extensions to collect.
[AssertFiles]	StartDate	Specifies the start date for a range of assert files to collect.
	EndDate	Specifies the end date for a range of assert files to collect.
	MatchingFiles	Specifies the assert file extensions to collect.
[LogFiles]	StartDate	Specifies the start date for a range of log files to collect.
	EndDate	Specifies the end date for a range of log files to collect.
	MatchingFiles	Specifies the log file extensions to collect. You can specify the collection of Siebel Application Response Measurement (Siebel ARM) files in this section by identifying the file extension SARM (for example, `*.sarm`).
[LogArchive]	NumArchives	Specifies that SDDC collects log archive files from the `NumArchives` directory.
	MatchingArchiveDir	Specifies the archive directories for collection.

Table 60. SDDC Configuration File and Parameters

Section	Parameter	Specifies
[SiebelServer]	LogDir	Specifies the Siebel Server log directory in the case of not being able to connect to the Siebel Gateway Name Server.
	LogArchiveDir	Specifies the Siebel Server log archive directory in the case of not being able to connect to the Siebel Gateway Name Server.
[GatewayServer]	LogDir	Specifies the Siebel Gateway Name Server in the case the directory name is different than the default.
[WebServer]	SiebelRoot	Specifies the Siebel Server root directory in the case the directory name is different than the default.

About SDDC Parameter Configuration

The StartDate, EndDate, and MatchingFiles parameters, which appear in several Siebel Diagnostic Data Collector (SDDC) configuration file sections, have common configuration details as shown in Table 61.

Table 61. Common Parameter Configuration Details

Common Parameters	Configuration Details
StartDate, EndDate	Set these parameters to specify collection of data between the two dates. If StartDate and EndDate are set, do not set the MaxNumFiles parameter. Configure the dates in the following format: `dd-Month_Acronym-yyyy` where: ■ *dd* is the integer of the date ranging from 01 to 31. ■ *Month_Acronym* is the three-letter month acronym as follows: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec. ■ *yyyy* is the integer of the year. Another valid configuration selection for the StartDate and EndDate parameters is NONE. If NONE is entered for StartDate and a valid date is entered for EndDate, files prior to the end date are collected. If NONE is entered for EndDate and a valid date is entered forStartDate, files from the start date to the current date are collected.
MatchingFiles	Set this parameter to collect multiple file formats using a comma-delimited list. Wildcard characters are also applicable. For example, to collect files containing siebmtsh in the filename with the extension .dmp and files of the type siebmtshmw5409.dmp, enter: `MatchingFiles=siebmtsh*.dmp,siebmtshmw5409.dmp`

Example of a Microsoft Windows SDDC Configuration File

The following listing is an example of a Microsoft Windows Siebel Diagnostic Data Collector (SDDC) configuration file. For parameter descriptions and configuration details, see "Configuring SDDC Content Under Microsoft Windows" on page 155.

```
[Main]

OutputDirectory=D:\s\752-15051\SWEApp\siebsnap

CollectLog=TRUE

CollectLogArchive=TRUE

CollectCrash=TRUE

CollectStderrFiles=TRUE
```

```
CollectDump=TRUE

CollectAssert=TRUE

SiebelBinDir = D:\s\752-15051\SWEApp\bin


[Registry]

Key01 = HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Tag

Key02 = HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Version

Key02 =
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\MaxHashTableSize

Key03 =
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\MaxFreeTcbs

Key04 =
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\MaxUserPort


[CrashFiles]

StartDate=05-Jan-2002

EndDate=10-Feb-2004

MatchingFiles = crash*.txt


[StderrFiles]

StartDate=05-Jan-2003

EndDate=10-Jun-2004

MatchingFiles = stderrout_*.txt


[ProcessDump]

StartDate=05-Jan-2002

EndDate=10-Dec-2004

MatchingFiles = *.dmp


[AssertFiles]

StartDate=05-Dec-2002
```

```
EndDate=10-Dec-2003
MatchingFiles=siebel_prefer*,siebel_assert*

[LogFiles]
StartDate=05-Dec-2002
EndDate=10-Dec-2003
MatchingFiles=*.log

[LogArchiveFiles]
StartDate=05-Dec-2002
EndDate=24-Feb-2003
MatchingFiles=*.log

[SiebelServer]
LogDir=M:\siebel\log
LogArchiveDir=M:\siebel\logarchive

[GatewayServer]
LogDir=M:\siebel\log

[WebServer]
SiebelRoot=M:\siebel
```

Configuring SDDC Content Under UNIX

You can configure the execution of Siebel Diagnostic Data Collector (SDDC) under UNIX to enhance the amount of information collected during an SDDC execution. Modify the SDDC INI file to record any configuration changes to the SDDC UNIX output.

The SDDC INI file, siebsnap.ini, is in the binary (bin) subdirectory of the Siebel Server root directory. To modify this file, open with a UNIX text editor.

For information on SDDC configurations under Microsoft Windows, see “Configuring SDDC Content Under Microsoft Windows” on page 155.

To configure SDDC to collect enhanced diagnostic information

1. With a text editor, open the siebsnap.ini file located in the binary (bin) subdirectory of the Siebel Server root directory.
2. Set specific parameters in the siebsnap.ini file based on how much information you require.

   For details and descriptions of SDDC INI file parameters, see Table 62.
3. Save the siebsnap.ini file.

Table 62. UNIX Configuration Parameters for siebsnap.ini file

INI File Parameter	Description	Default
OutputDirectory	Set this parameter to send the SDDC output to a different file location than the default.	The directory from which SDDC runs.
CollectLog	Set this parameter to TRUE to collect log file information. For further information on log files, see Chapter 4, "Configuring Siebel Server and Component Logging."	TRUE
CollectLogArchive	Set this parameter to TRUE to collect log archive information.	TRUE
CollectCrash	Set this parameter to TRUE to collect process failure file information.	TRUE
CollectDump	Set this parameter to TRUE to collect dump file information.	TRUE
CollectAssert	Set this parameter to TRUE to collect assert file information. For further information on assert files, see "About Other Siebel Server Log Files" on page 88.	TRUE
CollectFDR	Set this parameter to TRUE to collect Flight Data Recorder (FDR) file information. For further information on these log files, see "About Flight Data Recorder Log Files" on page 89.	TRUE
CollectSARM	Set this parameter to TRUE to collect Siebel Application Response Measurement (Siebel ARM) information. For further information on these Siebel ARM files, see *Siebel Performance Tuning Guide*.	FALSE
CollectQuickFix	Set this parameter to TRUE to collect the following quick fix files if they are present: upgrade.txt, obsolete.txt, incompatible.txt, and log.txt.	TRUE
FileRetention	Set this parameter to the number of .tar.z files that you want to retain. It is useful to retain snapshots of the system in regular intervals and compare them. Once SDDC reaches the value set by the FileRetention parameter, it overwrites the oldest file.	2

Table 62. UNIX Configuration Parameters for siebsnap.ini file

INI File Parameter	Description	Default
StartDate, EndDate	Set these parameters to allow the SDDC utility to collect files between a range of dates. Configure the date values in the following format: `dd-Month_Acronym-yy` where: ■ *dd* is the integer of the date ranging from 01 to 31. ■ *Month_Acronym* is a three-letter month acronym as follows: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec. ■ *yy* is the integer of the last two digits of the year. If no value is set for EndDate, then all files are collected for the current date.	EndDate =*Current Date*
StartTime, EndTime	Set these parameters in conjunction with the StartDate and EndDate parameters to further refine the range of files collected by the SDDC utility. Configure the time values in the 24-hour clock format. If no values are set, the default start time is 00:00 and the default endtime is 23:59.	StartTime =00:00, EndTime =23:59

Example of a UNIX SDDC Configuration INI File

The following listing is an example of the contents of a UNIX Siebel Diagnostic Data Collector (SDDC) configuration INI file. For parameter descriptions and configuration details, see "Configuring SDDC Content Under UNIX" on page 160.

```
OutputDirectory=

CollectLog=TRUE

CollectLogArchive=TRUE

CollectCrash=TRUE

CollectDump=TRUE

CollectAssert=TRUE

CollectFDR=TRUE

CollectSARM=FALSE

CollectQuickFix=TRUE

FileRetention=2
```

```
StartDate=01-Jan-03
StartTime=00:00

EndDate=20-Jan-03
EndTime=12:59
```

A List of Statistics and State Values

This appendix contains listings and brief descriptions of Siebel application statistics and state values. It includes the following topics:

List of Siebel Server Infrastructure Statistics

Table 63 lists the statistics defined for the Siebel Server infrastructure. For background information on Siebel application statistics, see "About Siebel Application Statistics" on page 28.

Table 63. List of Siebel Server Infrastructure Statistics

Statistic Name	Alias	Description
Avg. Transfer Time	SCBAvgTransferTime	Average time for transferring connection to component
Component Maxed Out Error	SCBCompMaxeOut	Number of times connection transfer failed because component is busy
Component Unavailable Error	SCBCompOffline	Failed to transfer connection due to Component Unavailable
Successful Connections	SCBFwdConn	Connection successfully forwarded

Table 63. List of Siebel Server Infrastructure Statistics

Statistic Name	Alias	Description
Total Connections	SCBTotalConn	Total number of connection attempts
Total Transfer Time	SCBTotalTransferTime	Total time spent transferring connections to component
FDR Buffer Wraps	FDRWraps	Number of buffer wraps
FDR Buffer Life in seconds	FDRBufferLife	Seconds since buffer was created
FDR Avg time between aging	FDRAgingRate	Avg seconds per buffer wrap
CPU Time	CPUTime	Total CPU time for component tasks (in seconds)
Elapsed Time	ElapsedTime	Total elapsed (running) time for component tasks (in seconds)
Maximum Peak Memory Usage	MaxPeakMemory	Peak Mem used by task. Rolls up differently from MinPeakMemory
Minimum Peak Memory Usage	MinPeakMemory	Peak Mem used by task. Rolls up differently than MaxPeakMemory
Sleep Time	SleepTime	Total amount of sleep time for component tasks (in seconds)
Number of Sleeps	Sleeps	Total number of sleeps for component tasks
Total Tasks	TotalTasks	Total number of tasks completed for server components
Tasks Exceeding Configured Capacity	TskXcdCfgCpt	Number of tasks stated that exceeded configured capacity
Num of DBConn Retries	NumDBConnRtrs	Number of retries due to database connection loss
Num of DLRbk Retries	NumDLRbkRtrs	Number of retries due to deadlock rollbacks
Num of Exhausted Retries	NumExhstRtrs	Number of Times All Retries are Exhausted

List of Siebel Application Object Manager Statistics

Table 64 describes the statistics specific to the Siebel Application Object Manager (AOM). For background information on Siebel application statistics, see “About Siebel Application Statistics” on page 28.

In Table 64, Siebel Application Object Manager session refers to a session between a client and an AOM. A session begins when the client connects to the AOM, and ends when the connection is terminated. A session starts a task on the AOM. If the AOM's Multithreaded parameter is set to TRUE, tasks are implemented as threads.

NOTE: Disregard the following statistics, which are not AOM-specific but appear in the component statistics view: Avg SQL Execute Time, Number of SQL Executes, Number of SQL Fetches, and Number of SQL Parses.

Table 64. List of AOM Statistics

Statistics Name	Alias	Description
Average Connect Time	AvgConnTime	Average connect time for Object Manager sessions
Average Reply Size	AvgRepSize	Average size of reply messages (in bytes)
Average Request Size	AvgReqSize	Average size of request messages (in bytes)
Average Requests Per Session	AvgReqs	Average number of requests per Object Manager session
Average Response Time	AvgRespTime	Average Object Manager response time
Average Think Time	AvgThinkTime	Average end-user think time between requests
Total Database Response Time	DBRespTime	Total database response and processing time in milliseconds
Object Manager Errors	Errors	Number of errors encountered during Object Manager session
Reply Messages	RepMsgs	Number of reply messages sent by the server
Total Reply Size	RepSize	Total size (in bytes) of reply messages
Request Messages	ReqMsgs	Number of request message received by the server
Total Request Size	ReqSize	Total size (in bytes) of request messages
Total Response Time	RespTime	Total Object Manager response time (in seconds)
Total Think Time	ThinkTime	Total end-user think time (in seconds)

List of Database Infrastructure Statistics

Table 65 describes the statistics specific to the database infrastructure. For background information on Siebel application statistics, see "About Siebel Application Statistics" on page 28.

Table 65. List of Database Infrastructure Statistics

Statistic Name	Alias	Description
Avg SQL Execute Time	AvgSQLExecTime	Average time for SQL execute operations (in seconds)
Avg SQL Fetch Time	AvgSQLFetchTime	Average time for SQL fetch operations (in seconds)
Avg SQL Parse Time	AvgSQLParseTime	Average time for SQL parse operations (in seconds)
SQL Execute Time	SQLExecTime	Total elapsed time for SQL execute operations (in seconds)
Number of SQL Executes	SQLExecs	Total number of SQL execute operations
SQL Fetch Time	SQLFetchTime	Total elapsed time for SQL fetch operations (in seconds)
Number of SQL Fetches	SQLFetches	Total number of SQL fetch operations
SQL Parse Time	SQLParseTime	Total elapsed time for SQL parse operations (in seconds)
Number of SQL Parses	SQLParses	Total number of SQL parse operations
SQL Statement Prepare Time	OM SQL Prepare Time	Total elapsed time for SQL preparation operations. This statistic is used to optimize performance of SQL statements. Time is calculated as follows: sum of connector prepare logic time + network time + database SQL parse time **NOTE:** It is recommended that you seek the assistance of your database administrator when optimizing the performance of SQL statements.

List of Siebel EAI Statistics

Table 66 describes the statistics specific to Siebel EAI. For background information on Siebel application statistics, see "About Siebel Application Statistics" on page 28.

Table 66. List of Siebel EAI Statistics

Statistic Name	Alias	Description
Siebel Adapter Total Query Calls	SiebAdptTotQueryCalls	Total number of query calls made to the Siebel Adapter
Siebel Adapter Total Query Size	SiebAdptTotQuerySize	Total cumulative size of output property sets (in KB) for all queries
Siebel Adapter Total Sync/Upsert Calls	SiebAdptTotSyncCalls	Total Number of non-query (synchronize, upsert, update or insert) calls made to Siebel Adapter
Siebel Adapter Total Sync Size	SiebAdptTotSyncSize	Total cumulative size of input property sets (in KB) for all non-query calls (synchronize, upsert, update or insert)
EAI Receiver Total Messages Processed	EAIRcvrMsgsProcessed	Total number of messages processed by the EAI Receiver
Total XML Generator Calls	XMLGenTotCalls	Total number of XML generator calls
Total XML Converter Size of Input Buffer	XMLParseTotSize	Total Cumulative Size of Input Buffer (in KB)
Total XML Converter Size of Output Buffer	XMLGenTotSize	Total Cumulative Size of Output Buffer (in KB)
Total XML Parser Calls	XMLParseTotCalls	Total number of XML Parser Calls

List of Siebel Remote Statistics

Table 67 describes the statistics specific to Siebel Remote. For background information on Siebel application statistics, see "About Siebel Application Statistics" on page 28.

Table 67. List of Siebel Remote Statistics

Statistic Name	Alias	Statistics Description
Avg node extracted time	AvgTime	Average time per node extracted (in seconds)
Total nodes extracted	TotNodes	Total number of nodes extracted
Total time processing nodes	TotTime	Total time consumed to extract the latest node (in seconds)
Avg node processing time	AvgTime	Average time per node processed (in milliseconds)
Total nodes processed	TotNodes	Total number of nodes processed
Total time processing nodes	TotTime	Total time consumed to process the current node in the current iteration (in milliseconds)
Monitor Period (in seconds)	MonitorPeriod	Advanced: Time duration for which all monitor data are collected and calculated (in seconds)
Current Operation Processing Rate	OperProcessRate	Advanced: Current operations processed per second
Current Position-Rule Operation Processing Rate	PostnOperProcessRate	Advanced: Current Position-Rule operations processed per second
Current Related Visibility-Event Operation Processing Rate	RelVisOperProcessRate	Advanced: Current Related Visibility-Event operations processed per second.
Current Visibility-Event Operation Processing Rate	VisOperProcessRate	Advanced: Current Visibility-Event operations processed per second
Total Operations Processed	TotOper	Advanced: Total operations processed during the monitored period
Total Vis-Event Operations Processed	TotVisOper	Advanced: Total Vis-Event operations processed during the monitored period
Total RelVisEvent Operations Processed	TotRelVisOper	Advanced: Total related Vis-Event operations processed during the monitored period

Table 67. List of Siebel Remote Statistics

Statistic Name	Alias	Statistics Description
Total Postn Related Operations Processed	TotPostnOper	Advanced: Total position rule related operations processed during the monitored period
Average Time for Processing a Node	AvgTimePerNode	Average time for processing one node (in milliseconds)
Total nodes processed	TotNodes	Total number of nodes processed
Total time processing nodes	TotTime	Total time consumed to process the current node in the current iteration (in milliseconds)
Average Number of Rows Downloaded	AvgDownloadRows	Advanced: Average number of downloaded records routed during the monitored period
Total Number of Removed Records	TotRecRemove	Advanced: Total number of removed records routed during the last monitored period.
Average Number of Removed Records	AvgRemoveRows	Advanced: Average number of removed records routed during the monitored period
Total Time for Loading Visdata	TotVisdataLoadTime	Advanced: Total time for loading Visdata during the monitored period (in millisecond).
Average Time for Loading Visdata	AvgVisdataLoadTime	Advanced: Average time for loading Visdata during the monitored period
Total Time for Visdata Load SQL	TotVisdataLoadSqlTime	Advanced: Total time for SQL execution for loading Visdata during the monitored period
Average Time for Visdata Load SQL	AvgVisdataLoadSqlTime	Advanced: Average time for SQL execution for loading Visdata during the monitored period
Total Visibility Check SQL Statements Executed	TotalVisCheckSQLExe	Advanced: Total number of Visibility Check SQLs executed for loading Visibility Data database during the last monitored period
Average Time for Waiting Visdata	AvgVisdataWaitTime	Advanced: Average time for waiting Visdata during the monitored period
Total Time for Waiting Visdata	TotVisdataWaitTime	Advanced: Total time for waiting Visdata during the monitored period (in millisecond).
Average Number of VisCheck Load SQL	AvgVisCheckLoadSql	Advanced: Average number of VisCheck SQLs executed for loading Visdata during the monitored period

Table 67. List of Siebel Remote Statistics

Statistic Name	Alias	Statistics Description
Total Records Fetched by Visibility Check	TotRecFetchVisCheck	Advanced: Total number of records fetched by Visibility Checks for loading Visibility Data database during the last monitored period
Average Number of VisCheck Load Rows	AvgVisCheckLoadRow	Advanced: Average number of VisCheck SQL records fetched for loading Visdata during the monitored period
Total Number of Visdata Loading	TotVisdataLoads	Advanced: Total numbers of Visdata loading during the monitored period
Total Number of VisData VisChecks	TotvisdataHit	Advanced: Total number of VisChecks that used VisData during the monitored period
Total Number of Visdata Access	TotVisdataAcc	Advanced: Total numbers of Visdata access during the monitored period
Number of Visibility Data Garbage Collection	NumVisDataGC	Advanced: Total numbers of garbage-collection performed on the Visibility Data database during the last monitored period.
Total Number of Visdata FSGC	TotVisdataFSGC	Advanced: Total Number of Visdata Full Scan Garbage Collection during the monitored period
Total Number of Visdata RKGC	TotVisdataRKGC	Advanced: Total Number of Visdata Random Kill Garbage Collection during the monitored period
Hit Ratio of Visibility Data Cache	HitRatioVisData	Advanced: Hit ratio of the Visibility Data cache during the last monitored period
Reconcile-Operations Routed per Period	ReconcileOperRoute	Advanced: Total number of reconcile-operations routed in the last monitored period
Download-Operations Routed per Period	DownloadOperRoute	Advanced: Total number of Download-operations routed during the last monitored period
Remove-Operations Routed per Period	RemoveOperRoute	Advanced: Total number of Remove-operations routed during the last monitored period
Number of Nodes Routed per Second	NumNodeRoute	Advanced: Number of nodes routed per second during the last monitored period.
Total Number of Opers Processed	TotOpers	Advanced: Total number of operations routed during the monitored period

Table 67. List of Siebel Remote Statistics

Statistic Name	Alias	Statistics Description
Monitor Period (in Seconds)	MonitorPeriod	Advanced: Time duration for which all monitor data are collected and calculated (in seconds)
Total Number of Nodes Processed	TotNumNode	Advanced: Total number of nodes routed during the monitored period
Operations Routed per Second	OperRoute	Advanced: Number of operations routed per second during the last monitored period
Total Time for TS I/O	TotTSTime	Advanced: Total time for tall/skinny file I/O during the monitored period
Total Number of TS I/O	TotTSAccess	Advanced: Total number of tall/skinny file I/O during the monitored period
Average I/O Time for Tall-Skinny File	AvgIOTSFile	Advanced: Average I/O time for tall-skinny file during the monitored period (in milliseconds).
Total Time for VisData I/O	TotVisdataTime	Advanced: Total time for visdata I/O during the monitored period (in millisecond).
Total Number of VisData I/O	TotVisdataAccess	Advanced: Total number of Visdata I/O during the monitored period
Average I/O Time for Visibility Data File	AvgIOVisDataFile	Advanced: Average I/O time for Visibility Data file during the monitored period (in millisecond).
Total Time for DX File I/O	TotDXFileTime	Advanced: Total time for DX File I/O during the monitored period
Total Number of DX File I/O	TotDXFileAccess	Advanced: Total number of DX File I/O during the monitored period
Average I/O Time for DX File	AvgIODXFile	Advanced: Average I/O time for DX file during the last monitored period (in millisecond).
Total Number of SQLs	TotNumSQLs	Advanced: Total number of SQLs executed during the monitored period
Average Number of SQLs	AvgNumSqls	Advanced: Average number of SQLs executed per operation routed during the monitored period
Total Time for Visibility Check	TotTimeVisCheck	Advanced: Total time spent for Visibility Check during the last monitored period (in millisecond).

Table 67. List of Siebel Remote Statistics

Statistic Name	Alias	Statistics Description
Average Time for Vis-Check	AvgVisCheckTime	Advanced: Average time for Vis-Check per operation routed during the monitored period
Total Time for Reconcile	TotReconcileTime	Advanced: Total time needed for reconcile during the monitored period
Average Time for Reconcile	AvgReconcileTime	Advanced: Average time needed for reconcile during the monitored period
Total Time for Performing Related Visibility Check	TotTimeRelVisCheck	Advanced: Total time spent for performing Related Visibility-Check during the last monitored period (in millisecond).
Average Time for Related Vis-Check	AvgRelVisCheckTime	Advanced: Average time needed for Related Vis-Check during the monitored period
Total Time for Download	TotTimeDownload	Advanced: Total time spent on downloading records the last monitored period (in millisecond).
Average Time for Download	AvgDownloadTime	Advanced: Average time for downloading records during the monitored period
Total Time for Reconcile VisCheck	TotRecVisCheckTime	Advanced: Total time needed for reconcile VisCheck during the monitored period
Average Time for Recocile Vis-Check	AvgRecVisCheckTime	Advanced: Average time needed for reconcile vis-check during the monitored period
Total Number of Records Downloaded	TotRecDownload	Advanced: Total number of downloaded records routed during the last monitored period.

List of Communication Server Statistics

Table 68 describes the statistics specific to Communication Server. For background information on Siebel application statistics, see “About Siebel Application Statistics” on page 28.

Table 68. List of Communication Server Statistics

Statistic Name	Alias	Description
Events Processed	EventsProcessed	Total number of events processed
Events Processed Rate	EventsProcessedRate	Rate of Processing the events

List of Assignment Manager Statistics

Table 69 describes the statistics specific to Assignment Manager. For background information on Siebel application statistics, see "About Siebel Application Statistics" on page 28.

Table 69. List of Assignment Manager Statistics

Statistic Name	Alias	Description
Number of object rows assigned	Number of rows assigned	This statistic represents the cumulative number of records assigned by this component since the server was started.

List of Workflow Manager Statistics

Table 70 describes the statistics specific to Workflow Manager. For background information on Siebel application statistics, see "About Siebel Application Statistics" on page 28.

Table 70. List of Workflow Manager Statistics

Statistic Name	Alias	Description
Number Requests	NumRequests	Total Number of requests processed
Policy Violations	Violations	Total Number of policy violations

List of Siebel Server Infrastructure State Values

Table 71 describes the state values specific to the Siebel Server infrastructure. For background information on Siebel application state values, see "About Siebel Application State Values" on page 29.

Table 71. List of Siebel Server Infrastructure State Values

State Value Name	Alias	Level	Description
Number of notification messages processed	NumNotifyMsgsProcessed	Component	Number of notification messages processed
Number of notification messages received	NumNotifyMsg	Component	Number of notification messages received over the pipe
Number of successful notification handler invocations	NumSuccessHndlrNotifications	Component	Number of successful notification handler invocations

Table 71. List of Siebel Server Infrastructure State Values

State Value Name	Alias	Level	Description
Number of failed notification handler invocations	NumFailedHndlrNotifications	Component	Number of failed notification handler invocations
Component Disable Time	CompDisableTime	Component	Timestamp of when the component was disabled
Component Enable Time	CompEnableTime	Component	Timestamp of when the component was most recently enabled
Component Start Time	CompStartTime	Component	Timestamp of when the component was started
Component Status	CompStatus	Component	Current status of the server component
Component Stop Time	CompStopTime	Component	Timestamp of when the component was shutdown
Component Tasks	CompTasks	Component	Current running tasks for the server component
Task Idle	TaskIdle	Task	TRUE, if task is idle
Task Label	TaskLabel	Task	Identifying label for this task
Task Memory Used	TaskMemory	Task	Current amount of memory used by task
Task Pause Time	TaskPauseTime	Task	Timestamp of when the task was paused
Task Start Time	TaskStartTime	Task	Timestamp of when the task was started
Task Ping Time	TaskPingTime	Task	Timestamp of when the task was last known to be active
Task Resume Time	TaskResumeTime	Task	Timestamp of when the task was most recently resumed
Task Schedule Time	TaskSchedTime	Task	Timestamp of when the task was scheduled
Task Status	TaskStatus	Task	Current status of the task
Task Stop Time	TaskStopTime	Task	Timestamp of when the task was shutdown

Table 71. List of Siebel Server Infrastructure State Values

State Value Name	Alias	Level	Description
User Name	User	Task	Database user name for the task
Disk Full State	DiskFullState	Component	This state value updates when the disk full state is reached during logging.
SCB Batch Execution Time	SCBBatchTime	Component	The number of seconds to execute a batch of CDAction commands. **NOTE:** CDAction is a Resonate Central Dispatch command-line utility that provides node status.
SCB Deregistration time	SCBDeregTime	Component	Time of last deregistration
Max. Transfer Time	SCBMaxTransferTime	Task	Maximum time for transferring connection to component
Min. Transfer Time	SCBMinTransferTime	Task	Minimum time for transferring connection to component
Server Non-Essential Tasks	NonEssientialTasks	Server	Total Non-Essential running tasks for the server
Server Disable Time	ServerDisableTime	Server	Timestamp of when the Siebel Server was disabled
Server Enable Time	ServerEnableTime	Server	Timestamp of when the Siebel Server was most recently enabled
Server Start Time	ServerStartTime	Server	Timestamp of when the Siebel Server was started
Server Status	ServerStatus	Server	Current status of the Siebel Server
Server Stop Time	ServerStopTime	Server	Timestamp of when the Siebel Server was shutdown
Server Cipher Strength	SrvrCipherStrength	Server	Server Encryption key length in bits

Table 71. List of Siebel Server Infrastructure State Values

State Value Name	Alias	Level	Description
Server Tasks	SrvrTasks	Server	Total running tasks for the server
Communication Cipher Strength	ComCipherStrength	Component	Communication Encryption key length in bits

List of Siebel Application Object Manager State Values

Table 72 describes the state values specific to the Siebel Application Object Manager (AOM). For background information on Siebel application state values, see "About Siebel Application State Values" on page 29.

Table 72. List of Siebel Application Object Manager State Values

State Value Name	Alias	Level	Description
Maximum Reply Size	MaxRepSize	Component	Maximum reply message size
Maximum Request Size	MaxReqSize	Component	Maximum request message size
Maximum Response Time	MaxRespTime	Component	Maximum response time for any Object Manager operation
Applet Name	ObjMgrApplet	Task	Current Applet Name
Business Component	ObjMgrBusComp	Task	Current Business Component
Business Service	ObjMgrBusSvc	Task	Current Business Service
View Name	ObjMgrView	Task	Current View Name
Scripting State	ScriptingState	Task	Current VB/eScript Scripting State
Database Login Id	DbLogin	Task	Database Login ID for the current user

List of Siebel EAI State Values

Table 73 describes the state values specific to Siebel EAI at the task level. For background information on Siebel application state values, see "About Siebel Application State Values" on page 29.

Table 73. List of Siebel EAI State Values

State Value Name	Alias	Description
Number of IDOC messages failed to dispatch	NumIdocMsgsDispatchFail	Total number of IDOC messages failed to dispatch
Number of IDOC messages successfully dispatched	NumIdocMsgsDispatchSucc	Total number of IDOC messages successfully dispatched
Number of IDOC messages received	NumIdocMsgsReceived	Total number of IDOC messages received
Number of IDOC messages sent	NumIdocMsgsSent	Total number of IDOC messages sent
Number of IDOCs failed to dispatch	NumIdocsDispatchFail	Total number of IDOCs failed to dispatch
Number of IDOCs successfully dispatched	NumIdocsDispatchSucc	Total number of IDOCs successfully dispatched
Number of IDOCs ignored	NumIdocsIgnored	Total number of IDOCs ignored
Number of IDOCs read	NumIdocsRead	Total number of IDOCs read
Number of IDOCs received	NumIdocsReceived	Total number of IDOCs received
Number of IDOCs sent	NumIdocsSent	Total number of IDOCs sent

List of Siebel Remote State Values

Table 74 describes the state values specific to Siebel Remote at the task level. For background information on Siebel application state values, see "About Siebel Application State Values" on page 29.

Table 74. List of Siebel Remote State Values

State Value Name	Alias	Description
Current node	CurrNode	Current node being extracted
Current node start time	CurrNodeStart	Start time when current node is extracted
Max time	MaxTime	Maximum time consumed to extract a node (in seconds)
Min time	MinTime	Minimum time consumed to extract a node (in seconds)

Table 74. List of Siebel Remote State Values

State Value Name	Alias	Description
Current file num	CurrFileNum	Current file number to be merged
Current node	CurrNode	Current node being merged
First file num	FirstFileNum	First file number to be merged
Last file num	LastFileNum	Last file number to be merged
Max time	Max Time	Maximum process time for a node (in milliseconds)
Min time	MinTime	Minimum process time for a node (in milliseconds)
Node iteration	NodeIter	The iteration number in which the current node is processed
Node start time	NodeStartime	Start time when current node is processed
Time for Txn to be Merged	TimeTxnMerge	Advanced: Elapsed time for a transaction to be merged in the last monitored period (in seconds)
Monitor Period (in Seconds)	MonitorPeriod	The period of time in which the statistic values are calculated
Low Scan Mark	LowScanMark	The lowest transaction ID to start to process
Time for Txn to be Processed	TimeTxnProcess	Advanced: Elapsed time for a transaction to be processed in the last monitored period (in seconds)
Current node	CurrNode	Advanced: Current node (mobile client or regional node) being routed
Current .dx read file	CurrRFile	Current .dx file being read
Current .dx write file	CurrWFile	Current .dx file being written
Current Transaction Id	CurrTxnId	Advanced: Current Transaction ID being routed
Current Node List	CurrNodeList	Advanced: Current list of nodes being routed
Last Update of Node List	LastUpdNodeList	Advanced: Timestamp of the last update of the node list being used
Time for Transaction to be Routed	TimeTxnRoute	Advanced: Elapsed time for a transaction to be routed in the last monitored period (in seconds)

List of Communications Server State Values

Table 75 describes the state values specific to Communications Server at the component level. For background information on Oracle's Siebel application state values, see "About Siebel Application State Values" on page 29.

Table 75. List of Communications Server State Values

State Value Name	Alias	Description
Feedback Counter	FeedbackCount	Number of feedback accumulated
Categorization Engine Initialized	Initialized	Include KB loaded
Last Update Time	LastUpdateTime	Last Time KB was updated
Number of Response Groups Loaded	NumResponseGroupsLoaded	Number of response groups currently loaded
Number of Comm Profiles Loaded	NumComm Profiles Loaded	Number of communication profiles currently loaded as part of the currently loaded response groups
Response Groups Loaded	ResponseGroupsLoaded	Response groups currently loaded
Number of busy work queue threads	NumBusyWorkerThreads	Number of busy work queue threads
Send Counter	SendCount	Number of messages sent

Related Topics

- About Siebel Application Statistics on page 28
- About Siebel Application State Values on page 29

B Sample Files for Monitoring Siebel Application Servers

Using the JMX API (Java Management Extensions application program interface), you can write Java code that performs application server monitoring operations across an enterprise.

This appendix contains the following sample JMX files for your use in monitoring Siebel application server operations at the enterprise, server, and component level:

Copy and modify these sample files for your specific monitoring needs. For more information about monitoring Siebel application servers, see "About Monitoring Application Server Operations Across an Enterprise" on page 43 and "Process of Monitoring Siebel Application Server Operations Across an Enterprise" on page 44.

key1.txt Sample JMX File

Use this file to create a unique key for generating an encrypted password. For example, you might use the following key:

sadmin1234567890

1srvr.xml Sample JMX File

Use this file to monitor a single application server across an enterprise. Edit the following code for your specific deployment:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Config>
   <KeyFileName>D:\src\key1.txt</KeyFileName>
- <Enterprises>
   - <Enterprise EnterpriseName="siebel" Username="SADMIN" Password="BNFJYDmXMiE=">
      - <Agents>
         <AgentSetting ServerName="sdchs21n625"
            URL="service:jmx:rmi://sdchs21n625/jndi/rmi://sdchs21n625:1599/jmx/siebel/agent" />
         </Agents>
      </Enterprise>
   </Enterprises>
</Config>
```

where:

- *EnterpriseName* is the name of the Siebel enterprise.

- *Username* is the user name for the management agent.
- *Password* is the encrypted password used by the management agent (see "To prepare for monitoring application server operations across an enterprise" on page 45).
- *ServerName* is the name of the Siebel Server.
- *URL* is the connect string (JMX Management Agent port numbers).

2srvr.xml Sample JMX File

Use this file to monitor two or more application servers across an enterprise. Edit the following code for your specific deployment:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Config>
   <KeyFileName>D:\src\key1.txt</KeyFileName>
- <Enterprises>
   - <Enterprise EnterpriseName="siebel" Username="SADMIN" Password="BNFJYDmXMiE=">
      - <Agents>
            <AgentSetting ServerName="sdchs21n019"
               URL="service:jmx:rmi://sdchs21n019/jndi/rmi://sdchs21n019:1499/jmx/siebel/agent" />
            <AgentSetting ServerName="sdchs21n078"
               URL="service:jmx:rmi://sdchs21n078/jndi/rmi://sdchs21n078:1599/jmx/siebel/agent" />
         </Agents>
      </Enterprise>
   </Enterprises>
</Config>
```

where:

- *EnterpriseName* is the name of the Siebel enterprise.
- *Username* is the user name for the management agent.
- *Password* is the encrypted password used by the management agent (see "To prepare for monitoring application server operations across an enterprise" on page 45).
- *ServerName* is the name of the Siebel Server.
- *URL* is the connect string (JMX Management Agent port numbers).

Enterprise.java Sample JMX File

Use this file to generate the Java code for monitoring *enterprise* operations across an enterprise, making sure the enterprise and component names are correct.

```
/**
*
*/
import com.siebel.management.jmxapi.*;
public class Enterprise
{
   /**
   * @param args
   */
   public static void main(String[] args)
   {
      JmxEnterpriseMBean emb = new JmxEnterprise();
      try
      {
         //+ The following 2 need to be changed by the person using this program
         String ent = "siebel";          // enterprise name
```

```
   //get servers//
   System.out.println("Servers:");
   String[] Servers = emb.getServers (ent);
   for (int i = 0; i < Servers.length; i++)
   {
      System.out.println("   " + Servers[i]);
   }

   //getConnectedServers//
   System.out.println("ConnectedServers:");
   String[] ConnectedServers = emb.getConnectedServers (ent);
   for (int i = 0; i < ConnectedServers.length; i++)
   {
      System.out.println("   " + ConnectedServers[i]);
   }

   //getDisconnectedServers//
   System.out.println("DisconnectedServers:");
   String[] DisconnectedServers = emb.getDisconnectedServers (ent);
   for (int i = 0; i < DisconnectedServers.length; i++)
   {
      System.out.println("   " + DisconnectedServers[i]);
   }

   //get comp availability//
   String arg = "ServerMgr";
   Float compState = emb.getComponentAvailability (ent, arg);
   System.out.println("getComponentAvailability('" + arg + "'):" + compState);

   //shutdownComponent//
   String arg1 = "Dbxtract";
   Boolean shutdownComp = emb.shutdownComponent(ent, arg1, false);
   System.out.println("shutdownComponent('" + arg1 + "'):" + shutdownComp);

   //A sleep time of 2min before staring the component//
   System.out.println("SleepTime:2 min");
   try {
   Thread.sleep( 120000 ); }
   catch ( InterruptedException e ) { System.out.println( "awakened prematurely" );}

   //startComponent//
   //String arg1 = "Dbxtract";
   Boolean startComp = emb.startComponent(ent, arg1);
   System.out.println("startComponent('" + arg1 + "'):" + startComp);

   //A sleep time of 2min before staring the component//
   System.out.println("SleepTime:2 min");
   try {
   Thread.sleep( 120000 ); }
   catch ( InterruptedException e ) { System.out.println( "awakened prematurely" );}

   //getparam//
   String arg2 = "Connect";
   String Param = emb.getParam (ent, arg2);
   System.out.println("getParam('" + arg2 + "'):" + Param);

   //shutdownEnterprise//
   Boolean shutdownEnt = emb.shutdownEnterprise (ent);
   System.out.println("shutdownEnterprise('" + ent + "'):" + shutdownEnt);

   //A sleeptime of 5 min before starting the enterprise//
   System.out.println("SleepTime:5 min");
   try {
   Thread.sleep( 300000 ); }
   catch ( InterruptedException e ) { System.out.println( "awakened prematurely" ); }

   //startEnterprise//
   Boolean startEnt = emb.startEnterprise (ent);
   System.out.println("startEnterprise('" + ent + "'):" + startEnt);
}
catch (Exception e)
```

```
      {
         e.printStackTrace();
      }
   }
}
```

Server.java Sample JMX File

Use this file to generate the Java code for monitoring *server* operations across an enterprise, making sure the enterprise and component names are correct.

```
/**
*
*/
import com.siebel.management.jmxapi.*;
public class Server
{
   /**
   * @param args
   */
   public static void main(String[] args)
   {
      JmxServerMBean smb = new JmxServer();
      try
      {
         //+ The following 2 need to be changed by the person using this program
         String ent = "siebel";                // enterprise name
         String srv = "sdchs21n016";          // server name

         //getstate//
         String srvState = smb.getState (ent, srv);
         System.out.println("getState('" + srv + "'):" + srvState);

         //shutdownserver//
         Boolean shutdownSrv = smb.shutdownServer (ent, srv);
         System.out.println("shutdownServer('" + srv + "'):" + shutdownSrv);

         //A sleep time of 5 min before starting the siebel server//
         System.out.println("SleepTime:5 min");
         try {
         Thread.sleep( 300000 ); }
         catch ( InterruptedException e ) { System.out.println( "awakened prematurely" );}

         //startserver//
         Boolean startSrv = smb.startServer (ent, srv);
         System.out.println("startServer('" + srv + "'):" + startSrv);

         //A sleep time of 5 min before starting the siebel server//
         System.out.println("SleepTime:5 min");
         try {
         Thread.sleep( 300000 ); }
         catch ( InterruptedException e ) { System.out.println( "awakened prematurely" );}

         //getparam//
         String param = "Connect";
         String paramval = smb.getParam (ent, srv, param);
         System.out.println("getParam ('" + param + "'):" + paramval);

         //getstat//
         String stat = "NumErrors";
         String statval = smb.getStat (ent, srv, stat);
         System.out.println("getStat ('" + stat + "'):" + statval);

         //getsval//
         String sval = "SrvrTasks";
         String svalval = smb.getSval (ent, srv, sval);
         System.out.println("getSval ('" + sval + "'):" + svalval);
```

```
            //getComps//
            String[] Comps = smb.getComps (ent, srv);
            System.out.println("Components:");
            for (int i = 0; i < Comps.length; i++)
            {
               System.out.println("   " + Comps[i]);
            }

            //getCompstate//
            String arg = "ServerMgr";
            String Compval = smb.getCompState (ent, srv, arg);
            System.out.println("getCompState ('" + arg + "'):" + Compval);

            //shutdowncomp//
            String Comp = "Dbxtract";
            Boolean compstop = smb.shutdownComp (ent, srv, Comp, false);
            System.out.println("shutdownComp ('" + Comp + "'):" + compstop);

            //A Sleep time of 2min before starting the component//
            System.out.println("Sleeptime: 2 min");
            try {
            Thread.sleep( 120000 ); }
            catch ( InterruptedException e ) { System.out.println( "awakened prematurely" ); }

            //startComp//
            Boolean compstart = smb.startComp (ent, srv, Comp);
            System.out.println("startComp ('" + Comp + "'):" + compstart);
         }
         catch (Exception e)
         {
            e.printStackTrace();
         }
      }
}
```

Component.java Sample JMX File

Use this file to generate the Java code for monitoring *component* operations across an enterprise, making sure the enterprise and component names are correct.

```
/**
*
*/
import com.siebel.management.jmxapi.*;
public class Component
{
   /**
   * @param args
   */
   public static void main(String[] args)
   {
      JmxComponentMBean cmb = new JmxComponent();
      try
      {
         //+ The following 3 need to be changed by the person using this program
         String ent = "siebel";          // enterprise name
         String srv = "sdchs21n625";     // server name
         String Comp = "SRBroker";       //component name

         //getState//
         String State = cmb.getState (ent, srv, Comp);
         System.out.println("getState('" + Comp + "'):" + State);

         //getAvailability//
         Float Availability = cmb.getAvailability (ent, srv, Comp);
         System.out.println("getAvailability ('" + Comp + "'):" + Availability);
```

```
            //getNumRunningTasks//
            Float RunningTasks = cmb.getNumRunningTasks (ent, srv, Comp);
            System.out.println("getNumRunningTasks ('" + Comp + "'):" + RunningTasks);

            //getParam//
            String param = "Connect";
            String paramval = cmb.getParam (ent, srv, Comp, param);
            System.out.println("getParam('" + param + "','" + Comp + "'):" + paramval);

            //getStat//
            String stat = "SleepTime";
            String statval = cmb.getStat (ent, srv, Comp, stat);
            System.out.println("getStat('" + stat + "','" + Comp + "'):" + statval);

            //getSval//
            String sval = "CompTasks";
            String svalval = cmb.getSval (ent, srv, Comp, sval);
            System.out.println("getSval('" + sval + "','" + Comp + "'):" + svalval);
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

Index

M

O

P

Q

R

S

T

U

W