Analytical debugging methods and problem analysis OC 4.5.2/RDC 4.5.3/TMS 4.6/AERS 4.6/Siebel Clinical environments

Sunil G. Singh

DBMS Consulting

8 October 2007

Administration & Configuration Management

Session 02

Acknowledgements

- Many thanks to the OCUG for this opportunity to present for the OCUG A&CM group.
- Many thanks to the OCUG Planning and Review Committee and OCUG A&CM Focus Group Chairs for their infinite patience in receiving and expeditious review of this presentation
- Many thanks to everyone who participated in the development of presentation.

Assumptions/Scope/Disclaimer

- Assumption: Audience has a basic understanding of the OLS 4.5.x architecture
- Scope: OC 4.5.2/RDC 4.5.3/TMS 4.6/AERS 4.6.x/Siebel 8.x.
- Disclaimer: These methods are for debugging production environment problems. They are not intended for bypassing security measures or regulatory policies, and nothing in this presentation should be construed as intended for such purposes.

Requirements for Debugging and Problem Analysis Within Production OLS environments.

- OLS production environments, especially those running RDC or with a global user base, have close to 24x7 usage and availability requirements.
- This type of environment increasingly presents problems and issues which must be debugged and analyzed in the production environment due to
    - critical time constraints
    - dependency on production infrastructure or components which are part of the issue
    - dependency on production data which does not exist elsewhere

Requirements for Debugging and Problem Analysis Within Production OLS environments. (2)

- While creating refreshed copies/clones of the production environment and reproducing a production issue is the best analytical method, it is not always practical because:
  - Production environments are more complex and sometimes can not be duplicated 100% in their entirety in a test environment (such as load balanced servers, public-facing network components, secure data)
  - Sufficient infrastructure (such as disk space and servers) may not exist to make copies of the production environment
  - Downtime may be required to create production copies which may not be available
  - System Administrator/DBA/Application Administrator resources and availability may not exist or may not be available in time

Requirements for Debugging and Problem Analysis Within Production OLS environments. (3)

- Executing many debugging techniques as documented can:
  - Cause short outages/downtimes which impact multiple users
  - Negatively impact performance for all users
  - Generate an excessive amount of debugging information/large logs, making it difficult to isolate a problem
- Having a way to debug a specific user's issue without effecting the production environment performance, causing downtime, or generating excessive debug files is a tremendous advantage in a production environment

Summary of Previously Discussed Debugging Methods

- Database Level RDBMS tracing for a running user’s session
  - Useful for tracing an already running session
- Static HTML file generation for individual testing of Forms-related configuration changes, not at the system-wide formsweb.cfg level
  - Useful for testing changes related to forms parameters or tracing forms which crash for a specific user
- Setting environment variables for specific sessions, not at the registry level
  - Useful for hiding forms such as OS Password, Job Scheduling and Reports Queue Monitoring

Setting User Logon Triggers

- Allows tracing or a session change to start at the time of the user logon
- Useful when
  - an entire job submitted by a user needs to be traced
  - the user's session fails during the login process
  - a session modifiable parameter needs to be changed/tested at the user level
    - Very useful in performance tuning, when combined with local schema objects

Setting User Logon Triggers: Method

- If the users themselves have CREATE TRIGGER privilege at the RDBMS level, a trigger can be compiled of the following form shown below.
- This example shows a tracing trigger than can be enabled for the specific user's session only.

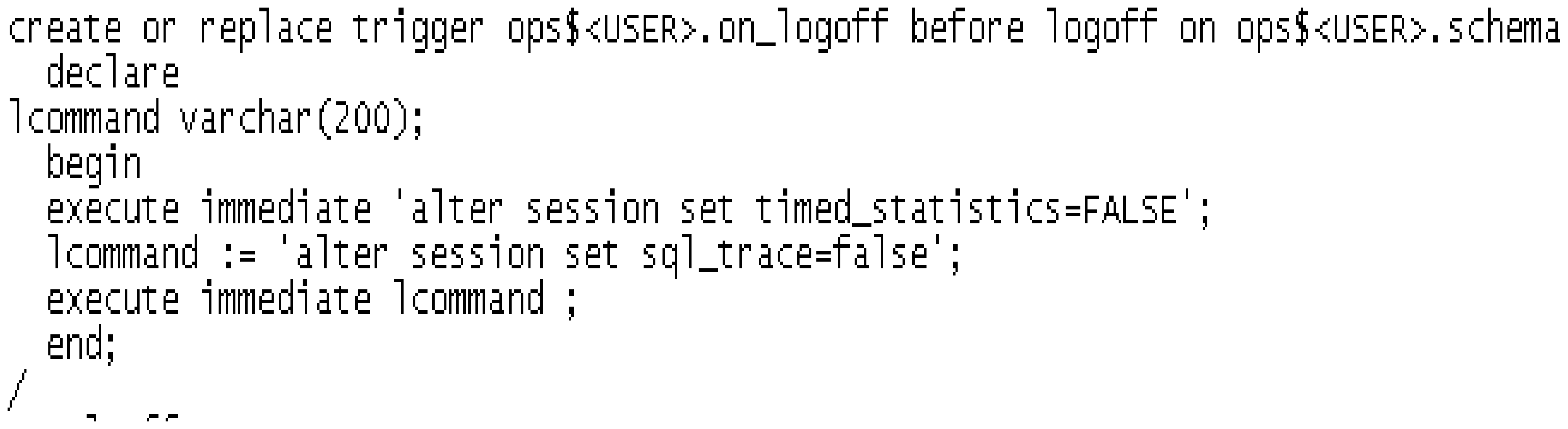
```
connect ops$<USER>
create or replace trigger ops$<USER>.on_logon after logon on ops$<USER>.schema
  declare
lcommand varchar(200);
  begin
  execute immediate 'alter session set max_dump_file_size=unlimited';
  execute immediate 'alter session set timed_statistics=TRUE';
  lcommand := 'alter session set events ''10046 trace name context forever, level 12''';
  execute immediate lcommand;
  end;
/
```

Setting User Logon Triggers: Method (2)

- The trigger can be enabled at any time
  - alter trigger ops$<user>.on_logon enable;
- This is especially useful for job tracing if a user must first interactively log on, and then submit a job. The trigger is enabled only AFTER the user initially logs on

Setting User Logon Triggers: Method (3)

- Be sure to disable or drop the trigger when tracing or testing is complete
  - alter trigger ops$<user>.on_logon disable;
  - drop trigger ops$<user>.on_logon;
- A Logoff trigger can also be defined to reverse any changes from the Logon Trigger:



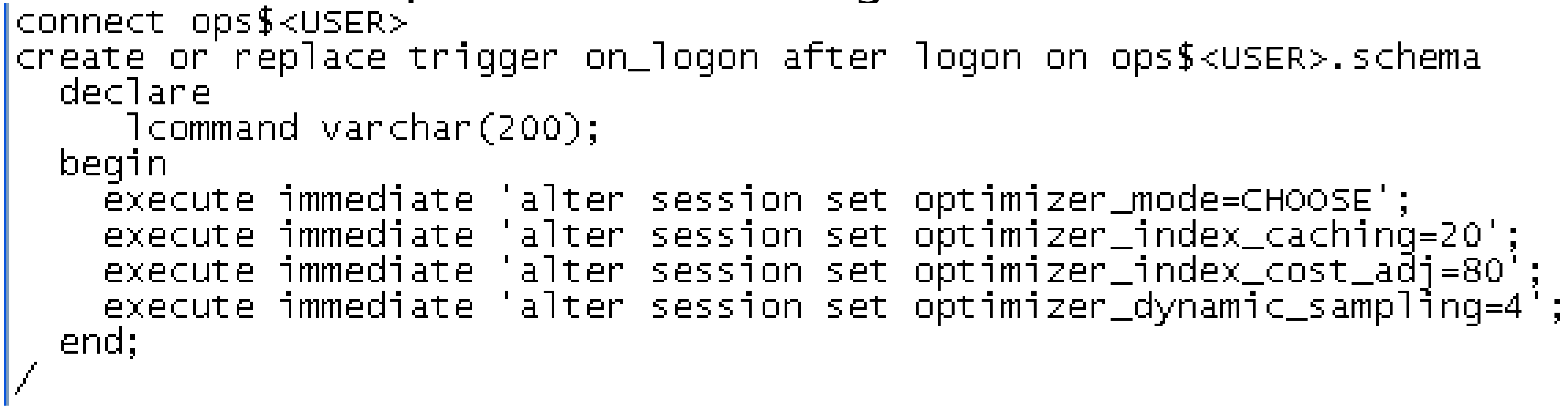
```
create or replace trigger ops$<USER>.on_logoff before logoff on ops$<USER>.schema
  declare
lcommand varchar(200);
  begin
  execute immediate 'alter session set timed_statistics=FALSE';
  lcommand := 'alter session set sql_trace=false';
  execute immediate lcommand ;
  end;
/
```

Setting User Logon Triggers: Isolating potential RDBMS changes

- Sometimes, this method is extremely useful to isolate potential RDBMS level changes to a specific user, so that they can be tested at the user level before impacting an entire production RDBMS.
- Changes to optimizer settings can be evaluated in this way, and optimizer bugs can be isolated if setting a User Logon trigger changes the behavior of a specific function

Setting User Logon Triggers: Isolating potential RDBMS changes

- In the case below, the Cost-Based Optimizer (CBO) settings for index caching and index costing are being tested at the user level before being changed at the RDBMS level
- If a specific script or database object, such as view, is being tested for performance, a local copy can be made in OPS$<USER>'s schema to evaluate these specific optimizer settings on that specific script or database object for isolated performance testing.



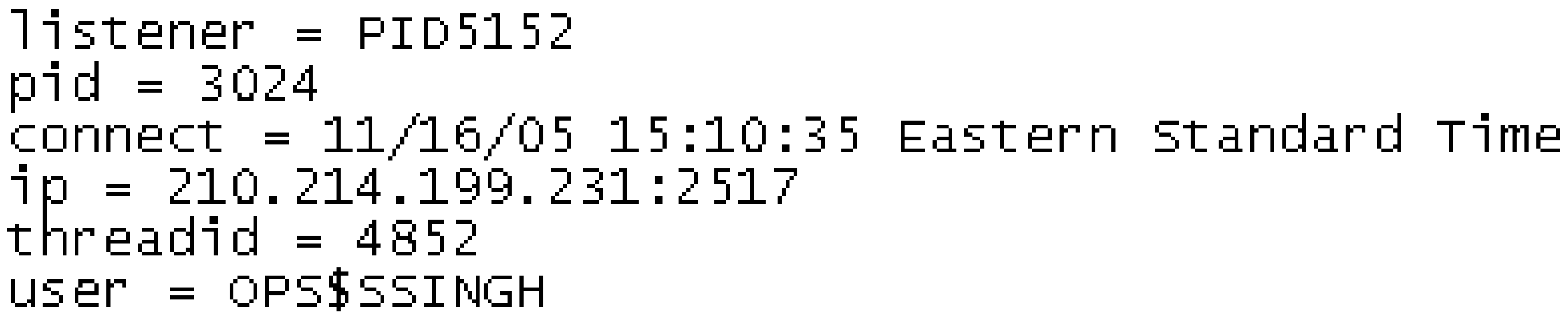
```
connect ops$<USER>
create or replace trigger on_logon after logon on ops$<USER>.schema
  declare
     lcommand varchar(200);
  begin
    execute immediate 'alter session set optimizer_mode=CHOOSE';
    execute immediate 'alter session set optimizer_index_caching=20';
    execute immediate 'alter session set optimizer_index_cost_adj=80';
    execute immediate 'alter session set optimizer_dynamic_sampling=4';
  end;
/
```

Cross Referencing Desktop Client session to Application Tier to RDBMS

- It is useful to determine which desktop user is connect to which Application Tier process
- Also, it is useful to relate this process back to the specific RDBMS process
- It is possible to map:
  - Application Server Process => Desktop IP address and username
  - RDBMS Process => Application Server Process
  - OS Process on the RDBMS Server => RDBMS Process
- This allows identification of "dead" or "orphaned" processes which can exhaust CPU, temporary space or server memory and provides an analytical method of killing these processes.

Cross Referencing Desktop Client session to Application Tier to RDBMS

- In Forms 6i (6.0.8.x) on Windows, each user connection creates at least one ifweb60.exe process
- At the time the ifweb60.exe is running normally, there is a .rti file created in the %ORACLE_806_HOME%\forms60\em directory, of the form em_<#####>.rti, where the <#####> is the Windows process ID of the ifweb60.exe process.
- The contents of this em_<####>.rti is shown below



```
listener = PID5152
pid = 3024
connect = 11/16/05 15:10:35 Eastern Standard Time
ip = 210.214.199.231:2517
threadid = 4852
user = OPS$SSINGH
```

Cross Referencing Desktop Client session to Application Tier to RDBMS

- From the em_<####>.rti file, the user ID and the IP address can be determined.
- When the session terminates normally, the em_<####>.rti file is removed.
- When the session times out or crashes, the em_<####>.rti remains and is not removed.
- Since Windows uses the same process ID over and over, these .rti files can be re-used and have multiple sections
- Also, in a Citrix environment, the .rti file will show the Citrix server IP address and not the desktop user's IP address.

Cross Referencing Desktop Client session to Application Tier to RDBMS

- The Windows process ID is the same as the first part of the process column in v$session in the RDBMS level.
- When querying v$session, be sure to include the username, process, program AND terminal column in environments where multiple Middle Tiers can connect to the same instance. This allows the correct identification of the Middle Tier and the correct ifweb60.exe process
- The second part of the process column is the Windows thread id. The thread ID can NOT be seen through the normal Task Manager process list and different utilties are required to see the thread ID.
- The OS-level process ID can be identified by joining PADDR in v$session to ADDR in v$process. The SPID is the process ID of the corresponding TNS listener process on the RDBMS Server if Dedicated Listener is being used.

Using URL additions for direct URL calls for Debugging Parameters

- Sometimes, an additional URL can be used instead of a static .html file to pass parameters to an application.
- This is especially true of RDC 4.5.2 and RDC 4.5.3, which do not use J-Initiator and therefore, do not respond to configurations or changes in formsweb.cfg
- Nearly all parameters in formsweb.cfg can be passed in the URL directly, but some applications do require the launch page to be used to work 100% correctly
    - RDC can not compute local PC timestamps if the RDC launch page is not used.

RDC 4.5.2 On-Site Individual URL Debugging

- Using variables in the calling URL, the following types of debug should be possible:
  - http://<Middle_Tier.Domain>/opardc/rdcLogin.do?event=doSetup&debug=<options>

  where <options> are:
  - all (all of the options below)
  - dcapi (Debug DCAPI, similar to OPA_DCAPI_PDF_DEBUG registry key = Y)
  - plugin (Debug RDC PDF Plugin, similar to setting the Debug in the Acrobat Reader => Oracle Clinical menu)
  - surround (Debug the Servlet, similar to setting the debug in the web.xml file)

RDC 4.5.3 Individual URL debugging

- The general URL is of the form:
  - http://<Middle_Tier.domain>:7778/olsa/oc/rdcLogin.do?event=doSetup&db=<global_tns_name>&debug=<options>

- This url also support additional debug parameters with &debug added to the end of the URL:
  - ALL (all of the options below, but ALL must be in Capital Letters)
  - dcapi (Debug DCAPI, similar to OPA_DCAPI_PDF_DEBUG registry key = Y)
  - plugin (Debug RDC PDF Plugin, similar to setting the Debug in the Acrobat Reader => Oracle Clinical menu)
  - surround (Debug the Servlet, similar to setting the debug in the web.xml file)

Additional Individual URL debugging

- Most OC/TMS/RDC URLs will support a &userid= parameter, where the userid is of the form <username>/<password>@instance.
- While this will allow automatic login to an application, its intended use is within a Single Sign-On environment, where the URL would be hidden from users but the login information would be constructed from an external source, such an AD or LDAP repository
- Most OC/TMS/RDC URLs will also support a &record= parameter, where record can be set to all or collect to create Forms tracing or measure statistics about the speed of the Form.

Decompiling .class files for Debugging Analysis ONLY

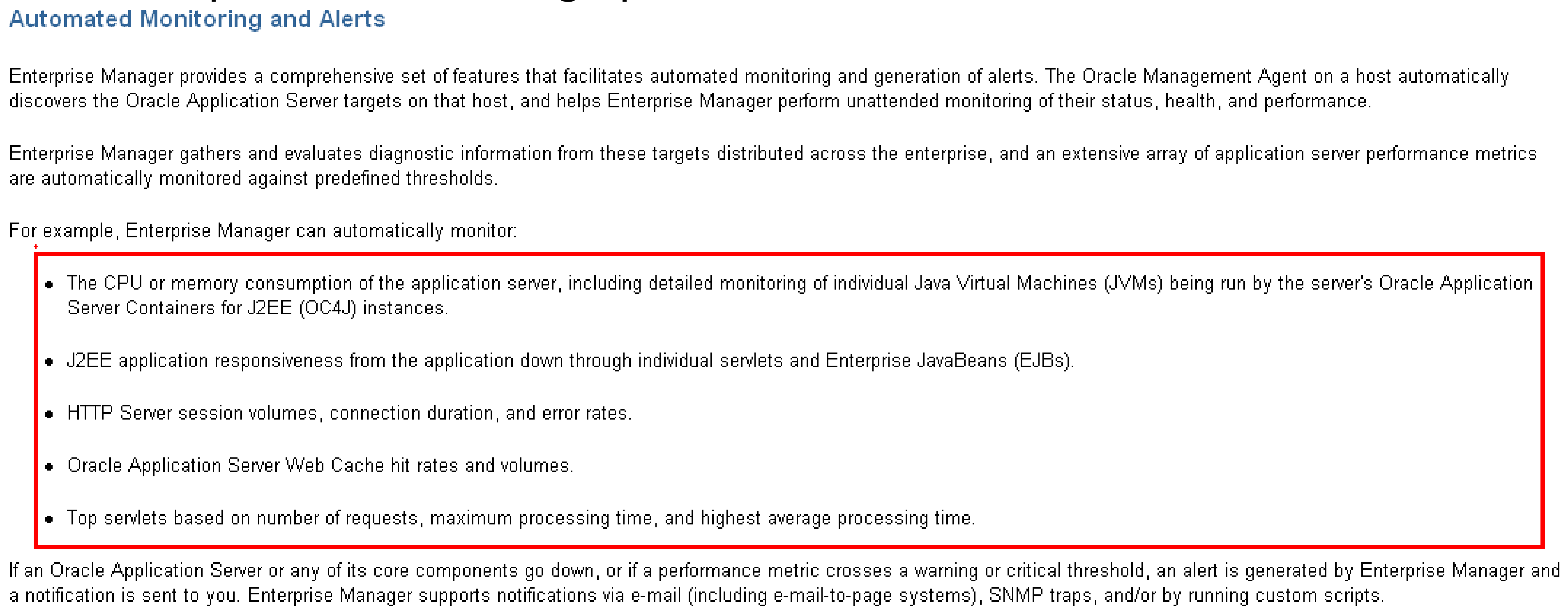
- The J2EE code which is currently available for RDC 4.5.2 and RDC 4.5.3 can be decompiled into source java code in most cases.
- Several freeware/GNU programs exist which provide the option of resolving .class files into source .java code.
- These should be used only for purposes of debugging servlet code in the case where specific problems are encountered which are clearly the result of the servlet behavior
  - Hard 2 hour disconnect timeout limit in RDC 4.5.0. This could only be determined by examining the source code of the specific class which caused this disconnection.

Using 10g Grid Control for Monitoring RDC 4.5.3 and Siebel Clinical 8.0

- Since RDC 4.5.3 runs solely on Oracle AS 10g R2 without additional Plug-Ins, it is a true J2EE application running with Oracle Containers for Java or OC4J
- Siebel Clinical can also Optionally deployed with the same technology stack for its application servers, although this is not required
- Oracle has introduced OEM 10g Grid Control with extensions to natively monitor and control both OC4J applications as well as Siebel RDBMS and application servers.
  - OEM requires that the Siebel Application Pack for OEM be installed on top of OEM 10g Grig Control to monitor Siebel
- Additionally, Siebel itself has some detailed logging configuration options available

Using 10g Grid Control for Monitoring RDC 4.5.3

- As shown in the next example, any J2EE application running in OC4J, such as RDC 4.5.3, can be monitored and controlled once a OEM 10g Grid Control agent is installed on the same Application Server running RDC 4.5.3.
- The Oracle Enterprise Manager Concepts Guide describes some possible Monitoring options and alerts show below



Automated Monitoring and Alerts

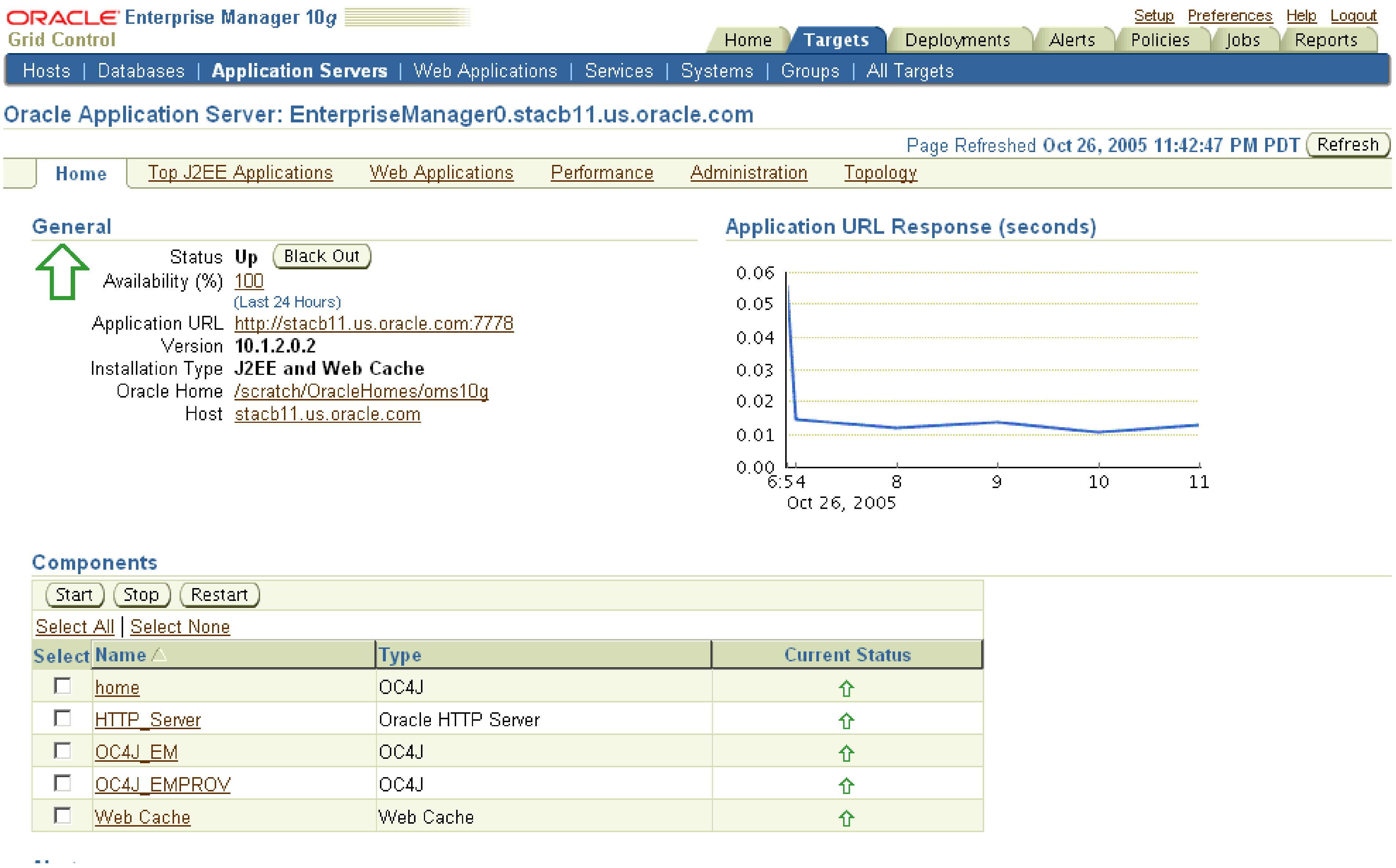
Enterprise Manager provides a comprehensive set of features that facilitates automated monitoring and generation of alerts. The Oracle Management Agent on a host automatically discovers the Oracle Application Server targets on that host, and helps Enterprise Manager perform unattended monitoring of their status, health, and performance.

Enterprise Manager gathers and evaluates diagnostic information from these targets distributed across the enterprise, and an extensive array of application server performance metrics are automatically monitored against predefined thresholds.

For example, Enterprise Manager can automatically monitor:

- The CPU or memory consumption of the application server, including detailed monitoring of individual Java Virtual Machines (JVMs) being run by the server's Oracle Application Server Containers for J2EE (OC4J) instances.
- J2EE application responsiveness from the application down through individual servlets and Enterprise JavaBeans (EJBs).
- HTTP Server session volumes, connection duration, and error rates.
- Oracle Application Server Web Cache hit rates and volumes.
- Top servlets based on number of requests, maximum processing time, and highest average processing time.

If an Oracle Application Server or any of its core components go down, or if a performance metric crosses a warning or critical threshold, an alert is generated by Enterprise Manager and a notification is sent to you. Enterprise Manager supports notifications via e-mail (including e-mail-to-page systems), SNMP traps, and/or by running custom scripts.


ORACLE Enterprise Manager 10g
Grid Control
Setup Preferences Help Logout
Home Targets Deployments Alerts Policies Jobs Reports
Hosts | Databases | Application Servers | Web Applications | Services | Systems | Groups | All Targets

Oracle Application Server: EnterpriseManager0.stacb11.us.oracle.com

Page Refreshed Oct 26, 2005 11:42:47 PM PDT

Refresh
Home Top J2EE Applications Web Applications Performance Administration Topology

General

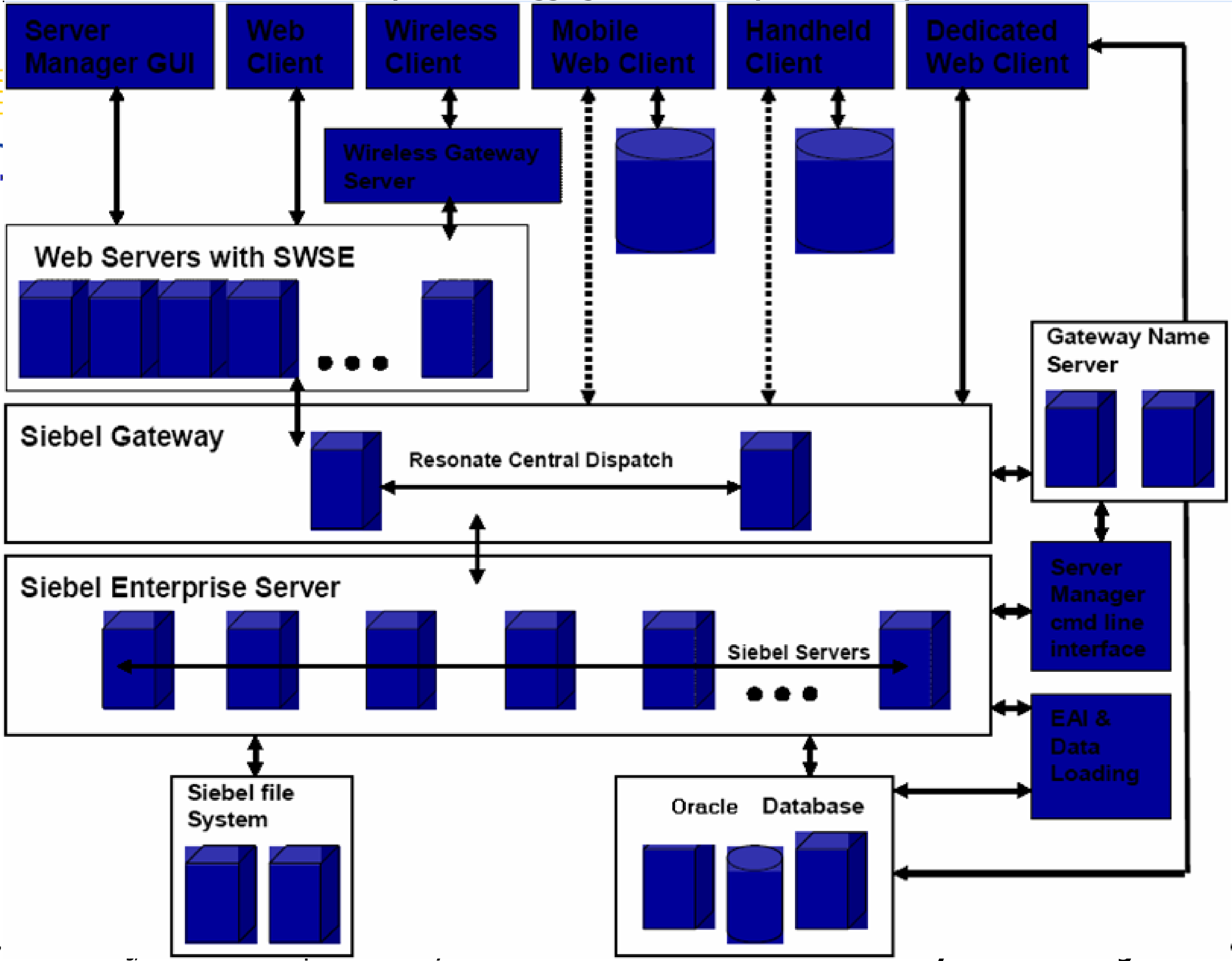
Status	Up
Availability (%)	100 (Last 24 Hours)
Application URL	http://stacb11.us.oracle.com:7778
Version	10.1.2.0.2
Installation Type	J2EE and Web Cache
Oracle Home	/scratch/OracleHomes/oms10g
Host	stacb11.us.oracle.com

Black Out
Application URL Response (seconds)
0.06
0.05
0.04
0.03
0.02
0.01
0.00
6:54
8
9
10
11
Oct 26, 2005

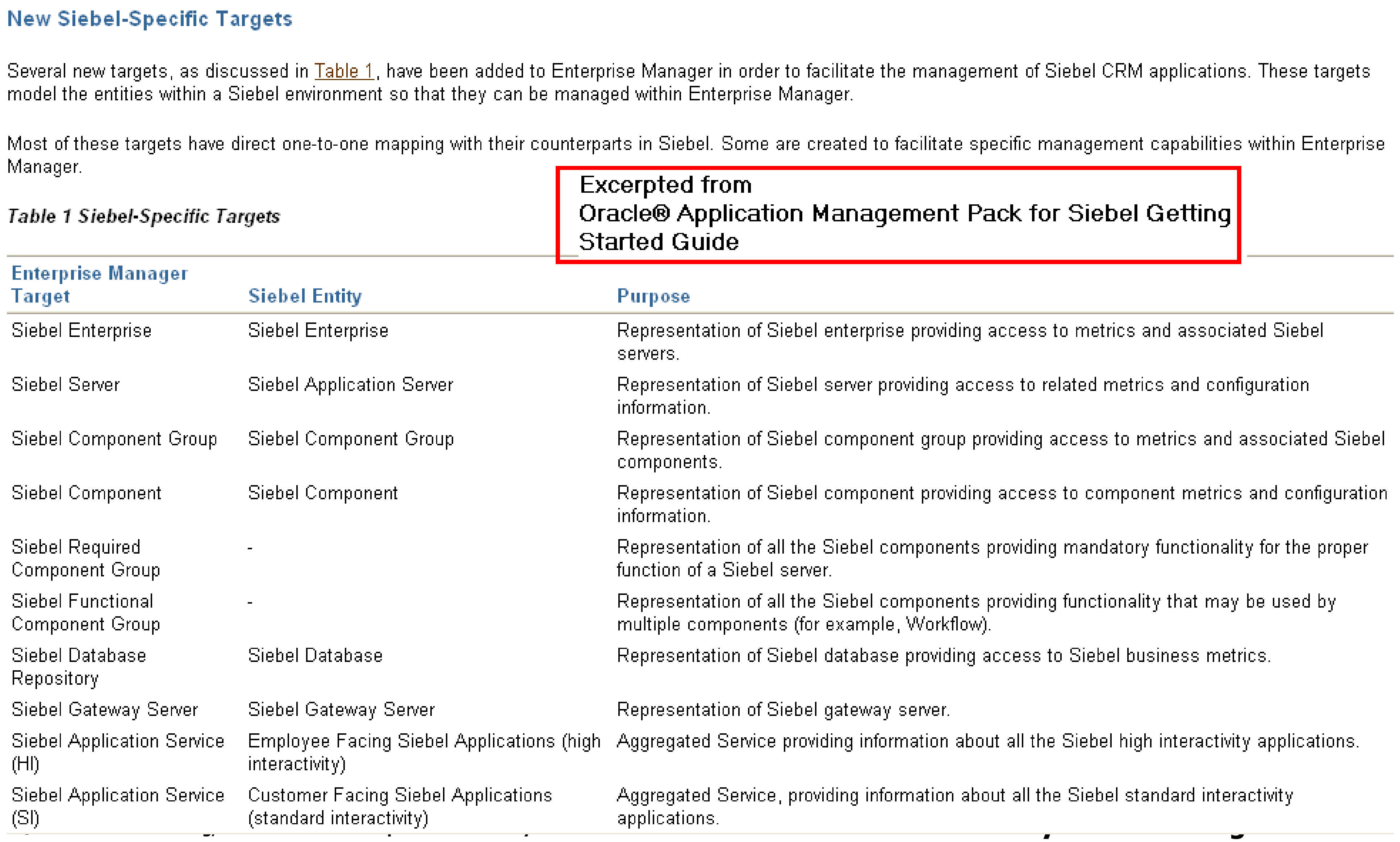
Components

Start Stop Restart
Select All | Select None

Select	Name △	Type	Current Status
☐	home	OC4J	⇧
☐	HTTP_Server	Oracle HTTP Server	⇧
☐	OC4J_EM	OC4J	⇧
☐	OC4J_EMPROV	OC4J	⇧
☐	Web Cache	Web Cache	⇧


Server Manager GUI
Web Client
Wireless Client
Mobile Web Client
Handheld Client
Dedicated Web Client
Wireless Gateway Server
Web Servers with SWSE
Gateway Name Server
Siebel Gateway
Resonate Central Dispatch
Siebel Enterprise Server
Siebel Servers
Server Manager cmd line interface
EAI & Data Loading
Siebel file System
Oracle Database

Using 10g Grid Control with Oracle Application Management Pack for Monitoring Siebel CRM



Excerpted from
Oracle® Application Management Pack for Siebel Getting Started Guide

New Siebel-Specific Targets

Several new targets, as discussed in Table 1, have been added to Enterprise Manager in order to facilitate the management of Siebel CRM applications. These targets model the entities within a Siebel environment so that they can be managed within Enterprise Manager.

Most of these targets have direct one-to-one mapping with their counterparts in Siebel. Some are created to facilitate specific management capabilities within Enterprise Manager.

Table 1 Siebel-Specific Targets

Enterprise Manager Target	Siebel Entity	Purpose
Siebel Enterprise	Siebel Enterprise	Representation of Siebel enterprise providing access to metrics and associated Siebel servers.
Siebel Server	Siebel Application Server	Representation of Siebel server providing access to related metrics and configuration information.
Siebel Component Group	Siebel Component Group	Representation of Siebel component group providing access to metrics and associated Siebel components.
Siebel Component	Siebel Component	Representation of Siebel component providing access to component metrics and configuration information.
Siebel Required Component Group	-	Representation of all the Siebel components providing mandatory functionality for the proper function of a Siebel server.
Siebel Functional Component Group	-	Representation of all the Siebel components providing functionality that may be used by multiple components (for example, Workflow).
Siebel Database Repository	Siebel Database	Representation of Siebel database providing access to Siebel business metrics.
Siebel Gateway Server	Siebel Gateway Server	Representation of Siebel gateway server.
Siebel Application Service (HI)	Employee Facing Siebel Applications (high interactivity)	Aggregated Service providing information about all the Siebel high interactivity applications.
Siebel Application Service (SI)	Customer Facing Siebel Applications (standard interactivity)	Aggregated Service, providing information about all the Siebel standard interactivity applications.

Types of Logging in available in Siebel Clinical: Application Object Manager

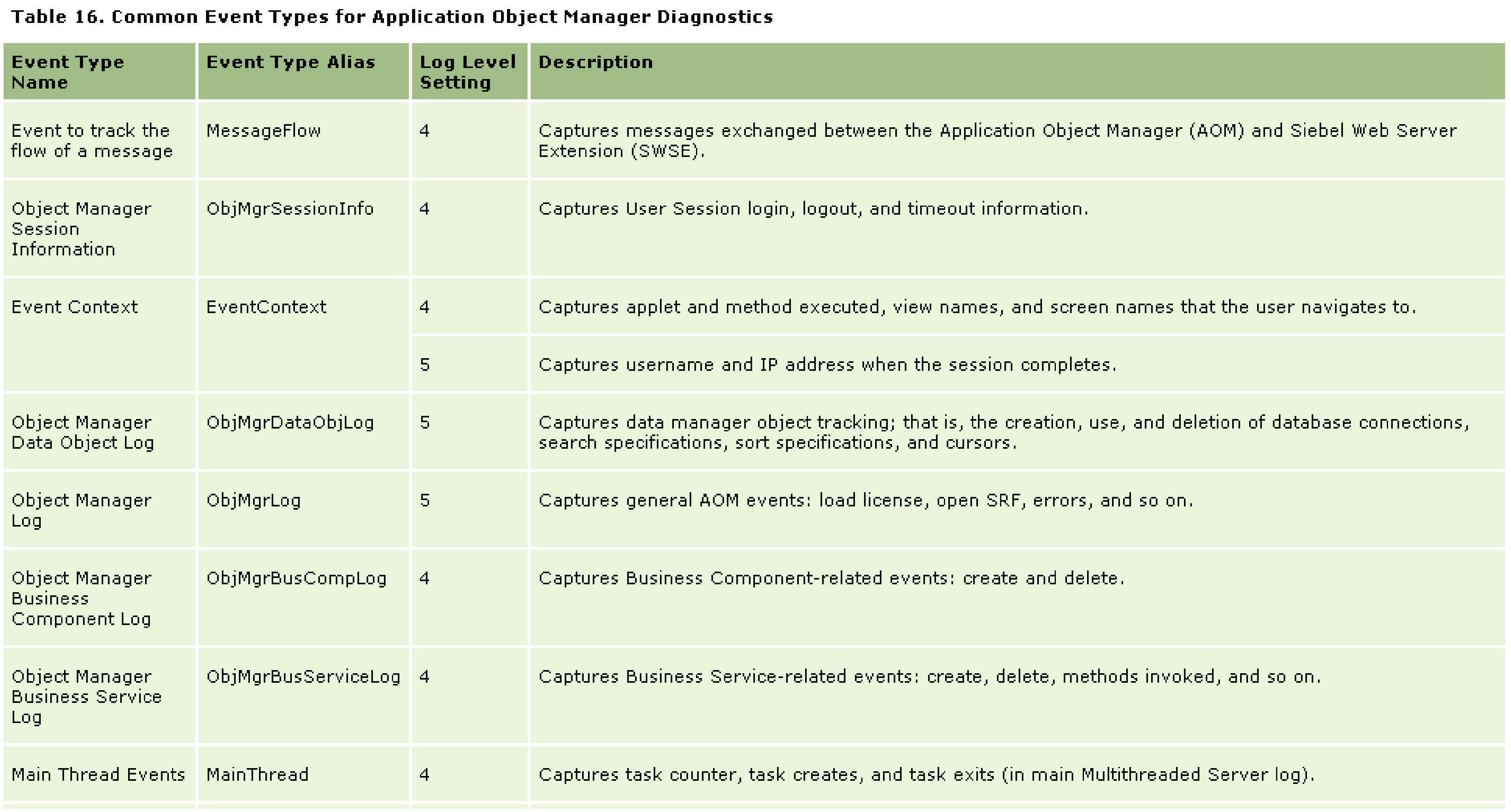
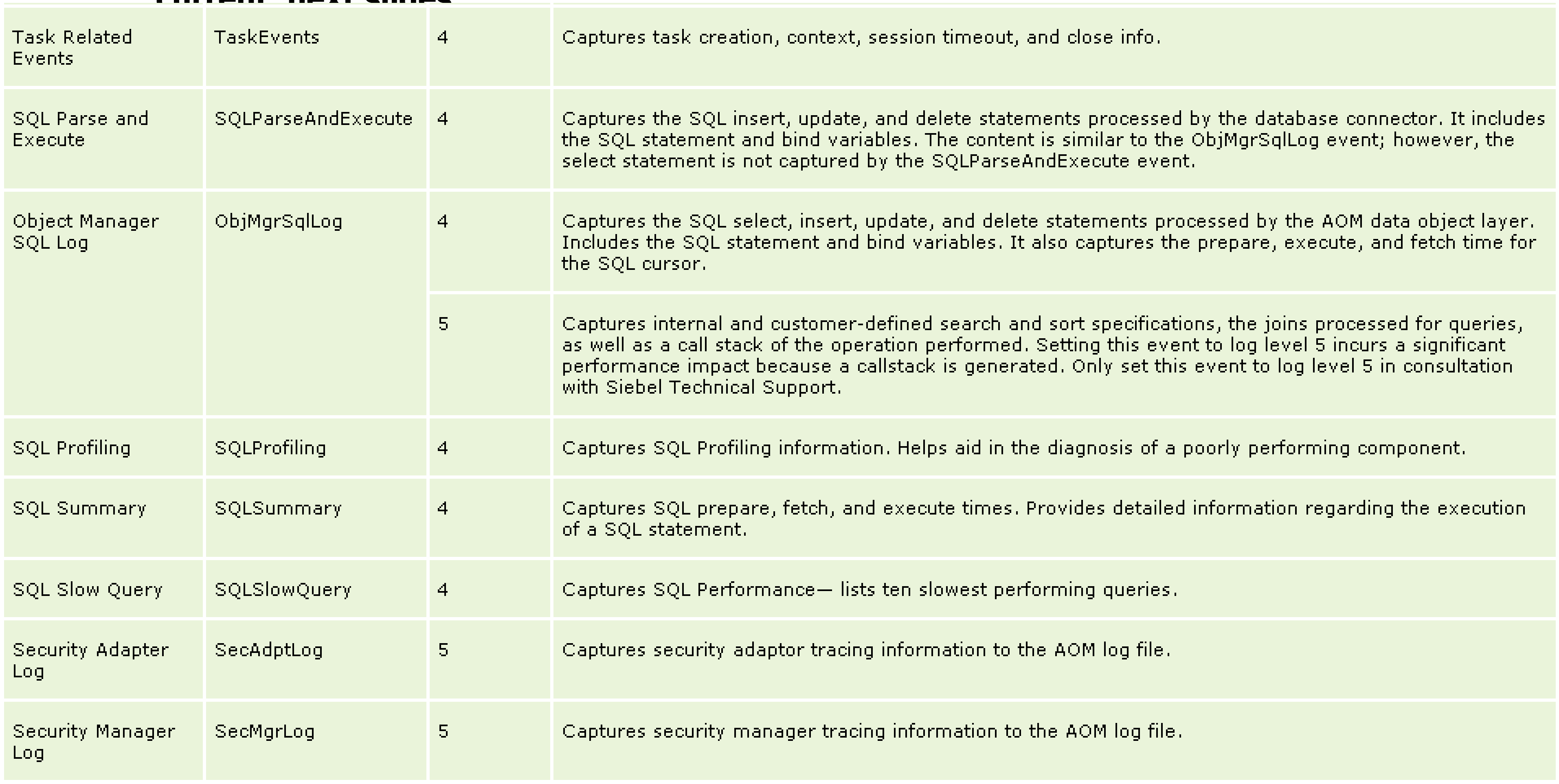


Table 16. Common Event Types for Application Object Manager Diagnostics

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 Application Object Manager (AOM) and Siebel Web Server Extension (SWSE).
Object Manager Session Information	ObjMgrSessionInfo	4	Captures User Session login, logout, and timeout information.
Event Context	EventContext	4	Captures applet and method executed, view names, and screen names that the user navigates to.
		5	Captures username and IP address when the session completes.
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.
Object Manager Log	ObjMgrLog	5	Captures general AOM events: load license, open SRF, errors, and so on.
Object Manager Business Component 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).

Types of Logging in available in Siebel Clinical: Application Object Manager (2)

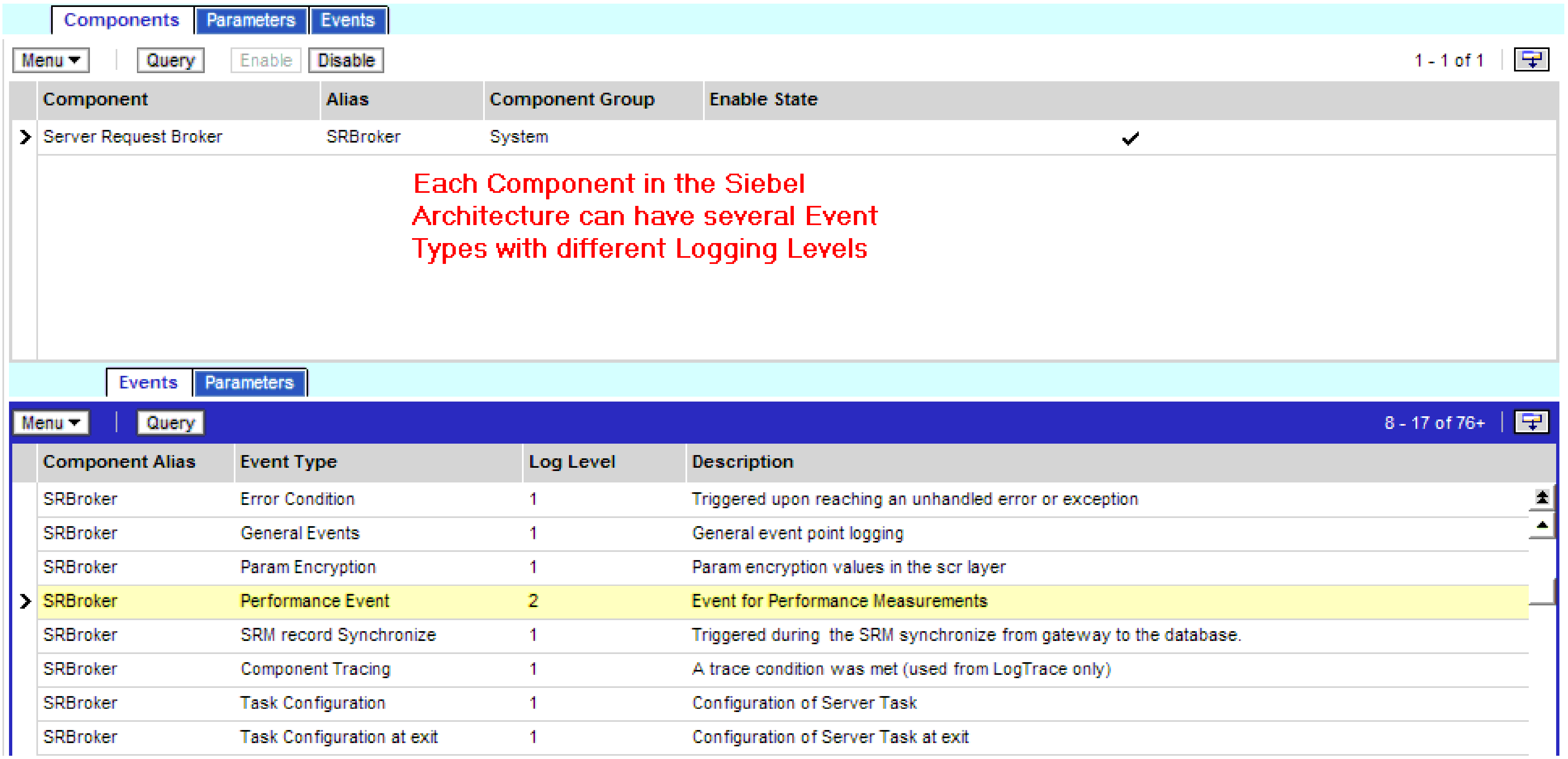
- **Excerpted from Siebel Systems Monitoring and Diagnostics Guide, previous, current, next slides**



Task Related Events	TaskEvents	4	Captures task creation, context, session timeout, and close info.
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 prepare, 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 Siebel Technical 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 a SQL statement.
SQL Slow Query	SQLSlowQuery	4	Captures SQL Performance— lists ten slowest performing queries.
Security Adapter Log	SecAdptLog	5	Captures security adaptor tracing information to the AOM log file.
Security Manager Log	SecMgrLog	5	Captures security manager tracing information to the AOM log file.

Types of Logging in available in Siebel Clinical

- Log Levels are 1=Most Severe to 6=Informational messages



	Component	Alias	Component Group	Enable State
>	Server Request Broker	SRBroker	System	✓

Each Component in the Siebel Architecture can have several Event Types with different Logging Levels

	Component Alias	Event Type	Log Level	Description
	SRBroker	Error Condition	1	Triggered upon reaching an unhandled error or exception
	SRBroker	General Events	1	General event point logging
	SRBroker	Param Encryption	1	Param encryption values in the scr layer
>	SRBroker	Performance Event	2	Event for Performance Measurements
	SRBroker	SRM record Synchronize	1	Triggered during the SRM synchronize from gateway to the database.
	SRBroker	Component Tracing	1	A trace condition was met (used from LogTrace only)
	SRBroker	Task Configuration	1	Configuration of Server Task
	SRBroker	Task Configuration at exit	1	Configuration of Server Task at exit

Conclusions

- The OLS Application Suite has become more complex but many components are using more current technologies. As a result, there are more options available to analyze and debug production level problems and issues
- It is still possible to utilize the core RDBMS-level and Forms level methods shown to isolate and identify issues in many cases, even when complex architecture or technology stacks are present

Question and Answers

All follow-up questions, please contact:

Sunil G. Singh
singh@clinicalserver.com
+1-860-983-5848
+1-888-463-4751
+91-98-181-34-017

Jose Garcia
jgarcia@clinicalserver.com
+1-347-452-9501

Dr. Letian Liu
lliu@clinicalserver.com
+86-134-0212-4879