Monitoring Oracle Application Servers

eG Innovations Product Documentation

www.eginnovations.com

Total Performance Visibility

Table of Contents

Table of Figures

Chapter 1: Introduction

Oracle Application Server offers a comprehensive solution for developing, integrating, and deploying applications, portals, and Web services. Based on a powerful and scalable J2EE server, Oracle Application Server provides complete business integration and business intelligence suites, and best-of-breed portal software. Further, it is designed for grid computing and full lifecycle support for Service-Oriented Architecture (SOA).

In order to ensure that the quality of the business-critical services supported by the Oracle Application server does not suffer due to incapacities of the application server, it is essential that the server is monitored 24 x 7, and its performance fine-tuned to meet the demands of the service users.

eG Enterprise offers specialized monitoring models for Oracle Application Server 9i and 10G, etc. A plethora of metrics relating to the health of the application servers can be monitored in real-time and alerts can be generated based on user-defined thresholds or auto-computed baselines. These metrics enable administrators to quickly and accurately determine server availability and responsiveness, resource usage at the host-level and at the application server level, how well the application server processes requests, how quickly the server completes transactions, overall server security, etc.

This document engages you in an elaborate discussion on how eG Enterprise monitors each of the popular web application servers in the market.

Chapter 2: How to Monitor Oracle Application Servers Using eG Enterprise?

eG Enterprise is capable of monitoring the Oracle application servers in both an agent-based and agentless manners. An eG Agent reports JVM-related Metrics for the application servers.

The JVM layer of the Oracle Application Server monitoring model is mapped to tests that report critical statistics related to the O9i application server's JVM. These statistics typically reveal the following:

- The count of classes loaded/unloaded (Java Classes test)
- JVM thread usage (JVM Threads test)
- CPU and memory usage of the JVM (JVM Cpu Usage test and JVM Memory Usage test)
- The effectiveness of the JVM's garbage collection activity (JVM Garbage Collections test)
- The uptime of the JVM (JVM Uptime test)
- Whether JMX is currently enabled/disabled on the target WebLogic server (JMX Connection to JVM test)
- The count and status of file descriptors (JVM File Descriptors test)

These tests connect to the JRE used by the O9i application server to pull out the above-mentioned metrics.

2.1 Pre-requisites for Monitoring Oracle 9i Application Server

eG agent runs various tests to track the performance of the server. For these tests to work, the eG agent should be configured to connect to the JRE and collect the required metrics, using one of the following methodologies:

- JMX (Java Management Extensions)
- SNMP (Simple Network Management Protocol)

Since both JMX and SNMP support are available for JRE 1.5 and above only, these tests will work only if the

O9i application server being monitored uses JRE 1.5 and above.

To know how to enable JMX or SNMP support for the JRE, refer to the Monitoring Java Applications document.

Note:

- To effectively monitor an O9i Application server on Unix, ensure that the install user of the application server and that of the eG agent (which is monitoring the application server) are the same.
- Any component on an Oracle 9i Application server (be it OC4J or the Oracle HTTP server) can be monitored by eG only if the Oracle 9i Application Server Release 2.0 is installed, with dmctl and dmstool.

Chapter 3: Monitoring Oracle 9i Application Servers

eG Enterprise prescribes an exclusive *O9i Application Server* monitoring model (see Figure 3.1), which runs tests on all the key components of the Oracle 9iAS from its JVM, JDBC drivers, caches, to its web modules, web contexts, transactions, etc. By automatically analyzing the performance data extracted by these tests from the server, eG Enterprise proactively detects potential bottlenecks to performance, accurately pin-points the root-cause of these problems, and instantly alerts administrators to the problem source.

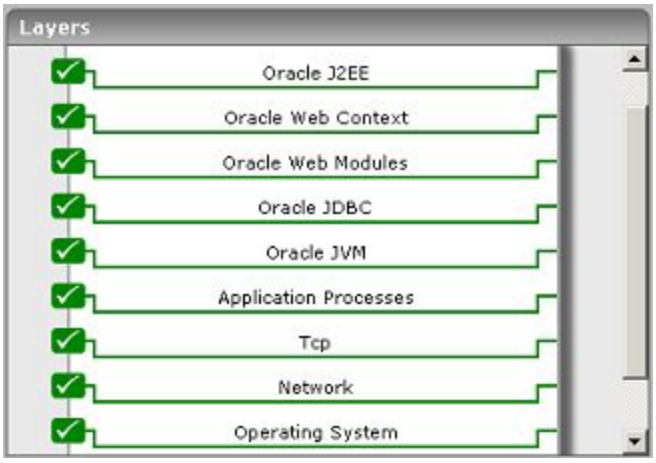

Figure 3.1: The layer model of the Oracle 9i AS

Each layer of the monitoring model depicted by Figure 3.1 monitors and reports metrics on every performance-influencing aspect of the Oracle 9i AS. Using these metrics, administrators can determine the following:

- Is the Oracle JVM available?
- Does the JVM have sufficient free memory?
- Is the workload on the JVM optimal?
- Is the server able to connect to the database quickly?
- Are too many JDBC connections open on the server?
- Does the connection cache have adequate free connections?
- Is the connection cache utilized effectively?
- Are transaction rollbacks kept at a minimum?
- Are the web modules processing requests quickly?

- Is any web module taking too much time to process requests?
- Does the server take too much time to service JSP and servlet requests?

The sections below discuss the metrics reported by the top 5 layers of Figure 3.1 only, as all other layers have been dealt with in the *Monitoring Unix and Windows Servers* document.

3.1 The Oracle JVM Layer

Use the test associated with this layer to track the workload on and the memory heap usage of the Oracle JVM.

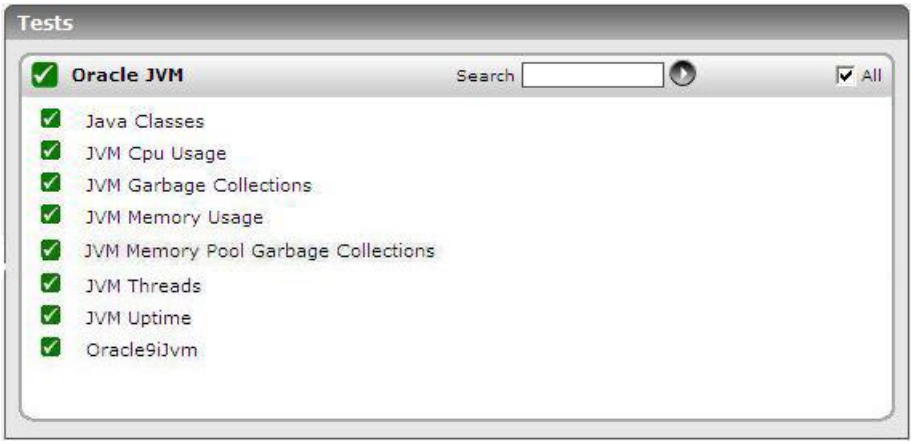

Figure 3.2: Tests associated with the Oracle JVM layer

3.1.1 Oracle 9i Jvm Test

The JVM test is used to analyze overall JVM performance for the processes of an Oracle 9i AS instance. The JVM metrics provide useful information about threads and heap memory allocation. You should check these values to make sure that JVM resources are utilized within expected ranges.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every Oracle 9i AS instance monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.

Parameter	Description
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http:///<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Active thread groups	Indicates the current number of active thread groups in the JVM.	Number	A high value for this measure is indicative of a high load on the JVM.
Active threads	Indicates the current number of active threads in the JVM.	Number	A high value for this measure is indicative of a high load on the JVM.
Free heap	Indicates the unused memory in the JVM.	MB	The free heap value governs the performance of the JVM. A low value could result in excessive garbage collection, slowing down responses/processing in the JVM.
Min free heap	Indicates the low water mark of memory in the JVM since it started.	MB	This metric indicates the time when the free heap reached its minimum value.
Heap usage	Indicates the used memory in the JVM currently.	MB	A very high value for this measure indicates heavy load on the JVM.
Max heap usage	The high water mark of heap used in the JVM since it was started.	MB	A very high value for this measure indicates heavy load on the JVM.
JVM availability	Indicates the availability	Percent	If this value is 100%, then it indicates

Measurement	Description	Measurement Unit	Interpretation
	of the Oracle 9i AS instance.		that the JVM is running. If it is 0, it indicates that the JVM is not available or is not running.

3.1.2 Java Transactions Test

When a user initiates a transaction to a Java-based web application, the transaction typically travels via many Java components before completing execution and sending out a response to the user.

The key Java components have been briefly described below:

- **Filter**: A filter is a program that runs on the server before the servlet or JSP page with which it is associated. All filters must implement *javax.servlet.Filter*. This interface comprises three methods: *init, doFilter, and destroy*.
- **Servlet**: A servlet acts as an intermediary between the client and the server. As servlet modules run on the server, they can receive and respond to requests made by the client. If a *servlet is designed to handle HTTP requests*, it is called an HTTP Servlet.
- **JSP**: Java Server Pages are an extension to the Java servlet technology. A JSP is translated into Java servlet before being run, and it processes HTTP requests and generates responses like any servlet. Translation occurs the first time the application is run.
- **Struts**: The Struts Framework is a standard for developing well-architected Web applications. Based on the Model-View-Controller (MVC) design paradigm, it distinctly separates all three levels (Model, View, and Control).

A delay experienced by any of the aforesaid Java components can adversely impact the total response time of the transaction, thereby scarring the user experience with the web application. In addition, delays in JDBC connectivity and slowdowns in SQL query executions (if the application interacts with a database), bottlenecks in delivery of mails via the Java Mail API (if used), and any slow method calls, can also cause insufferable damage to the 'user-perceived' health of a web application.

The challenge here for administrators is to not just isolate the slow transactions, but to also accurately identify where the transaction slowed down and why - is it owing to inefficent JSPs? poorly written servlets or struts? poor or the lack of any JDBC connectivity to the database? long running queries? inefficient API calls? or delays in accessing the POJO methods? The **eG JTM Monitor** provides administrators with answers to these questions!

With the help of the **Java Transactions** test, the **eG JTM Monitor** traces the route a configured web transaction takes, and captures live the total responsiveness of the transaction and the response time of each Java component it visits en route. This way, the solution proactively detects transaction slowdowns, and also precisely points you to the Java components causing it - is it the Filters? JSPs? Servlets? Struts? JDBC? SQL query? Java Mail API? or the POJO? In addition to revealing where (i.e., at which Java component) a transaction slowed down, the solution also provides the following intelligent insights, on demand, making root- cause identification and resolution easier:

a. A look at the methods that took too long to execute, thus leading you to those methods that may have contributed to the slowdown;

b. Single-click access to each invocation of a chosen method, which provides pointers to when and where a method spent longer than desired;

c. A quick glance at SQL queries and Java errors that may have impacted the responsiveness of the transaction;

Using these interesting pointers provided by the **eG JTM Monitor**, administrators can diagnose the root-cause of transaction slowdowns within minutes, rapidly plug the holes, and thus ensure that their critical web applications perform at peak capacity at all times!

Before attempting to monitor Java transactions using the **eG JTM Monitor** , the following configurations will have to be performed:

1. In the <EG_INSTALL_DIR>\lib directory (on Windows; on Unix, this will be /opt/egurkha/lib) of the eG agent, you will find the following files:

    a. eg_jtm.jar

    b. aspectjrt.jar

    c. aspectjweaver.jar

    d. jtmConn.props

    e. jtmLogging.props

    f. jtmOther.props

2. Login to the system hosting the Java application to be monitored.

3. If the eG agent will be 'remotely monitoring' the target Java application (i.e., if the Java application is to be monitored in an 'agentless manner'), then, copy all the files mentioned above from the

<EG_INSTALL_DIR>\lib directory (on Windows; on Unix, this will be /opt/egurkha/lib) of the eG agent to any location on the Java application host.

4. Then, proceed to edit the start-up script of the Java application being monitored, and append the following lines to it:

```
set JTM_HOME=<<PATH OF THE LOCAL FOLDER CONTAINING THE JAR FILES AND PROPERTY FILES
LISTED ABOVE>>"
-javaagent:%JTM_HOME%\aspectjweaver.jar"
"-DEG_JTM_HOME=%JTM_HOME%"
```

Note that the above lines will change based on the operating system and the web/web application server being monitored.

Then, add the **eg_jtm.jar**, **aspectjrt.jar**, and **aspectjweaver.jar** files to the **CLASSPATH** of the Java application being monitored.

Finally, save the file. Once this is done, then, the next time the Java application starts, the **eG JTM Monitor** scans the web requests to the application for configured URL patterns. When a match is found, the **eG JTM Monitor** collects the desired metrics and stores them in memory.

Then, every time the eG agent runs the **Java Transactions** test, the agent will poll the **eG JTM Monitor** (on the target application) for the required metrics, extract the same from the application's memory, and report them to the eG manager.

5. Next, edit the **jtmConn.props** file. You will find the following lines in the file:

```
#Contains the connection properties of eGurkha Java Transaction Monitor
JTM_Port=13631
Designated_Agent=
```

By default, the **JTM_Port** parameter is set to 13631. If the Java application being monitored listens on a different JTM port, then specify the same here. In this case, when managing a **Java Application** using the eG administrative interface, specify the **JTM_Port** that you set in the **jtmConn.props** file as the **Port** of the Java application.

Also, against the **Designated_Agent** parameter, specify the IP address of the eG agent which will poll the **eG JTM Monitor** for metrics. If no IP address is provided here, then the **eG JTM Monitor** will treat the host from which the very first 'measure request' comes in as the **Designated_Agent**.

Note:

In case a specific **Designated_Agent** is not provided, and the **eG JTM Monitor** treats the host from which the very first 'measure request' comes in as the **Designated_Agent**, then if such a **Designated_Agent** is stopped or uninstalled for any reason, the **eG JTM Monitor** will wait for a maximum of 10 measure periods for that 'deemed' **Designated_Agent** to request for metrics. If no requests come in for 10 consecutive measure periods, then the **eG JTM Monitor** will begin responding to 'measure requests' coming in from any other eG agent.

6. Finally, save the **jtmConn.props** file.

To know how to configure this test refer to *Monitoring Java Applications* document.

3.1.3 Java Classes Test

This test reports the number of classes loaded/unloaded from the memory.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the server being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.
Port	Refers to the port at which the specified host listens to.
Mode	This test can extract metrics from the Java application using either of the following mechanisms: • Using SNMP-based access to the Java runtime MIB statistics; • By contacting the Java runtime (JRE) of the application via JMX To configure the test to use SNMP, select the **SNMP** option. On the other hand, choose the **JMX** option to configure the test to use JMX instead. By default, the **JMX** option is chosen here.
JMX Remote Port	This parameter appears only if the Mode is set to **JMX**. Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_ Home>\jre\lib\management folder used by the target application (refer to the

Parameter	Description
	Monitoring Java Applications document for details).
User, Password, and Confirm Password	These parameters appear only if the Mode is set to **JMX**. If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds if the mode is **JMX**, and **10** seconds if the mode is **SNMP**.
SNMPPort	This parameter appears only if the Mode is set to **SNMP**. Here specify the port number through which the server exposes its SNMP MIB. Ensure that you specify the same port you configured in the *management.properties* file in the <JAVA_ HOME>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
SNMPVersion	This parameter appears only if the Mode is set to **SNMP**. By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the SNMPversion list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
SNMPCommunity	This parameter appears only if the Mode is set to **SNMP**. The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the SNMPVersion chosen is **v3**, then this parameter will not appear.
UserName	This parameter appears only when **v3** is selected as the SNMPVersion. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using

Parameter	Description
	the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against this parameter.
Context	This parameter appears only when v3 is selected as the SNMPVersion. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the SNMPEngineID value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a contextName). If the Username provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the UserName in the Context text box. By default, this parameter is set to *none*.
AuthPass	Specify the password that corresponds to the above-mentioned UserName. This parameter once again appears only if the SNMPversion selected is **v3**.
Confirm Password	Confirm the AuthPass by retyping it here.
AuthType	This parameter too appears only if **v3** is selected as the SNMPVersion. From the AuthType list box, choose the authentication algorithm using which SNMP v3 converts the specified username and password into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options: • **MD5** – Message Digest Algorithm • **SHA** – Secure Hash Algorithm
EncryptFlag	This flag appears only when **v3** is selected as the SNMPVersion. By default, the eG agent does not encrypt SNMP requests. Accordingly, the this flag is set to **No** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **Yes** option.
EncryptType	If this EncryptFlag is set to **Yes**, then you will have to mention the encryption type by selecting an option from the EncryptType list. SNMP v3 supports the following encryption types: • **DES** – Data Encryption Standard • **AES** – Advanced Encryption Standard
EncryptPassword	Specify the encryption password here.

Parameter	Description
Confirm Password	Confirm the encryption password by retyping it here.
Data Over TCP	This parameter appears only if the Mode is set to **SNMP**. By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set this flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Classes loaded	Indicates the number of classes currently loaded into memory.	Number	Classes are fundamental to the design of Java programming language. Typically, Java applications install a variety of class loaders (that is, classes that implement java.lang.ClassLoader) to allow different portions of the container, and the applications running on the container, to have access to different repositories of available classes and resources. A consistent decrease in the number of classes loaded and unloaded could indicate a road-block in the loading/unloading of classes by the class loader. If left unchecked, critical resources/classes could be rendered inaccessible to the application, thereby severely affecting its performance.
Classes unloaded	Indicates the number of classes currently unloaded from memory.	Number	
Total classes loaded	Indicates the total number of classes loaded into memory since the JVM started, including those subsequently unloaded.	Number	

3.1.4 JVM Threads Test

This test reports the status of threads on the JVM, and also reveals resource-hungry threads, so that threads that are unnecessarily consuming CPU resources can be killed.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the server being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.
Port	Refers to the port at which the specified host listens to.
Mode	This test can extract metrics from the Java application using either of the following mechanisms: • Using SNMP-based access to the Java runtime MIB statistics; • By contacting the Java runtime (JRE) of the application via JMX To configure the test to use SNMP, select the **SNMP** option. On the other hand, choose the **JMX** option to configure the test to use JMX instead. By default, the **JMX** option is chosen here.
JMX Remote Port	This parameter appears only if the Mode is set to **JMX**. Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_Home>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
User, Password, and Confirm Password	These parameters appear only if the Mode is set to **JMX**. If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then

Parameter	Description
	you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds if the mode is **JMX**, and **10** seconds if the mode is **SNMP**.
SNMPPort	This parameter appears only if the Mode is set to **SNMP**. Here specify the port number through which the server exposes its SNMP MIB. Ensure that you specify the same port you configured in the *management.properties* file in the <JAVA_HOME>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
SNMPVersion	This parameter appears only if the Mode is set to **SNMP**. By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the SNMPversion list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
SNMPCommunity	This parameter appears only if the Mode is set to **SNMP**. The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the SNMPVersion chosen is **v3**, then this parameter will not appear.
UserName	This parameter appears only when **v3** is selected as the SNMPVersion. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against this parameter.
Context	This parameter appears only when v3 is selected as the SNMPVersion. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the SNMPEngineID value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a

Parameter	Description
	contextName). If the Username provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the UserName in the Context text box. By default, this parameter is set to *none*.
AuthPass	Specify the password that corresponds to the above-mentioned UserName. This parameter once again appears only if the SNMPversion selected is **v3**.
Confirm Password	Confirm the AuthPass by retyping it here.
AuthType	This parameter too appears only if **v3** is selected as the SNMPVersion. From the AuthType list box, choose the authentication algorithm using which SNMP v3 converts the specified username and password into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options: • **MD5** – Message Digest Algorithm • **SHA** – Secure Hash Algorithm
EncryptFlag	This flag appears only when **v3** is selected as the SNMPVersion. By default, the eG agent does not encrypt SNMP requests. Accordingly, the this flag is set to **No** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **Yes** option.
EncryptType	If this EncryptFlag is set to **Yes**, then you will have to mention the encryption type by selecting an option from the EncryptType list. SNMP v3 supports the following encryption types: • **DES** – Data Encryption Standard • **AES** – Advanced Encryption Standard
EncryptPassword	Specify the encryption password here.
Confirm Password	Confirm the encryption password by retyping it here.
Data Over TCP	This parameter appears only if the Mode is set to **SNMP**. By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set this flag to **Yes**. By default, this flag is set to **No**.
Pct Low CPU Util	This test reports the number of threads in the JVM that are consuming low CPU. This

Parameter	Description
Threads	thread count will include only those threads for which the CPU usage is equal to or lesser than the value specified in the Pct Low CPU Util Threads text box. The default value displayed here is 30.
Pct Medium CPU Util Threads	This test reports the number of threads in the JVM that are consuming CPU to a medium extent. This thread count will include only those threads for which the CPU usage is higher than the Pct Low CPU Util Threads configuration and is lower than or equal to the value specified in the Pct Medium CPU Util Threads text box. The default value displayed here is 50.
Pct High CPU Util Threads	This test reports the number of threads in the JVM that are consuming high CPU. This thread count will include only those threads for which the CPU usage is either greater than the Pct Medium CPU Util Threads configuration, or is equal to or greater than the value specified in the Pct High CPU Util Threads text box. The default value displayed here is 70.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against DD frequency.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Total threads	Indicates the total number	Number	

Measurement	Description	Measurement Unit	Interpretation
	of threads (including daemon and non-daemon threads).		
Runnable threads	Indicates the current number of threads in a runnable state.	Number	The detailed diagnosis of this measure, if enabled, provides the name of the threads, the CPU usage by the threads, the time for which the thread was in a blocked state, waiting state, etc.
Blocked threads	Indicates the number of threads that are currently in a blocked state.	Number	If a thread is trying to take a lock (to enter a synchronized block), but the lock is already held by another thread, then such a thread is called a blocked thread. The detailed diagnosis of this measure, if enabled, provides in-depth information related to the blocked threads.
Waiting threads	Indicates the number of threads that are currently in a waiting state.	Number	A thread is said to be in a Waiting state if the thread enters a synchronized block, tries to take a lock that is already held by another thread, and hence, waits till the other thread notifies that it has released the lock. Ideally, the value of this measure should be low. A very high value could be indicative of excessive waiting activity on the JVM. You can use the detailed diagnosis of this measure, if enabled, to figure out which threads are currently in the waiting state. While waiting, the Java application program does no productive work and its ability to complete the task-at-hand is degraded. A certain amount of waiting may be acceptable for Java

Measurement	Description	Measurement Unit	Interpretation
			application programs. However, when the amount of time spent waiting becomes excessive or if the number of times that waits occur exceeds a reasonable amount, the Java application program may not be programmed correctly to take advantage of the available resources. When this happens, the delay caused by the waiting Java application programs elongates the response time experienced by an end user. An enterprise may use Java application programs to perform various functions. Delays based on abnormal degradation consume employee time and may be costly to corporations.
Timed waiting threads	Indicates the number of threads in a TIMED_ WAITING state.	Number	When a thread is in the TIMED_ WAITING state, it implies that the thread is waiting for another thread to do something, but will give up after a specified time out period. To view the details of threads in the TIMED_WAITING state, use the detailed diagnosis of this measure, if enabled.
Low CPU threads	Indicates the number of threads that are currently consuming CPU lower than the value configured in the Pct Low CPU Util Threads text box.	Number	
Medium CPU threads	Indicates the number of threads that are currently consuming CPU that is higher than the value configured in the Pct Low CPU Util Threads text box	Number	

Measurement	Description	Measurement Unit	Interpretation
	and is lower than or equal to the value specified in the Pct Medium CPU Util Threads text box.		
High CPU threads	Indicates the number of threads that are currently consuming CPU that is either greater than the percentage configured in the Pct Medium CPU Util Threads or lesser than or equal to the value configured in the Pct High CPU Util Threads text box.	Number	Ideally, the value of this measure should be very low. A high value is indicative of a resource contention at the JVM. Under such circumstances, you might want to identify the resource-hungry threads and kill them, so that application performance is not hampered. To know which threads are consuming excessive CPU, use the detailed diagnosis of this measure.
Peak threads	Indicates the highest number of live threads since JVM started.	Number	
Started threads	Indicates the total number of threads started (including daemon, non-daemon, and terminated) since JVM started.	Number	
Daemon threads	Indicates the current number of live daemon threads.	Number	
Deadlock threads	Indicates the current number of deadlocked threads.	Number	Ideally, this value should be 0. A high value is a cause for concern, as it indicates that many threads are blocking one another causing the application performance to suffer. The detailed diagnosis of this measure, if enabled, lists the deadlocked threads and their resource usage.

Note:

If the Mode for the **JVM Threads** test is set to **SNMP**, then the detailed diagnosis of this test will not display the **Blocked Time and Waited Time** for the threads. To make sure that detailed diagnosis reports these details also, do the following:

- Login to the server host.
- Go to the <JAVA_HOME>\jre\lib\management folder used by the WebLogic server, and edit the *management.properties* file in that folder.
- Append the following line to the file:

```
com.sun.management.enableThreadContentionMonitoring
```

Note:

While viewing the measures reported by the **JVM Thread** test, you can also view the resource usage details and the **stack trace** information for all the threads, by clicking on the **STACK TRACE** link in the **Measurements** panel.

A **stack trace** (also called **stack backtrace** or **stack traceback**) is a report of the active stack frames instantiated by the execution of a program. It is commonly used to determine what threads are currently active in the JVM, and which threads are in each of the different states – i.e., alive, blocked, waiting, timed waiting, etc.

Typically, when the JVM begins exhibiting erratic resource usage patterns, it often takes administrators hours, even days to figure out what is causing this anomaly – could it be owing to one/more resource-intensive threads being executed by the WebLogic server? If so, what is causing the thread to erode resources? Is it an inefficient piece of code? In which case, which line of code could be the most likely cause for the spike in resource usage? To be able to answer these questions accurately, administrators need to know the complete list of threads that are executing on the JVM, view the **stack trace** of each thread, analyze each stack trace in a top-down manner, and trace where the problem originated.

3.1.5 JVM Cpu Usage Test

This test measures the CPU utilization of the JVM. If the JVM experiences abnormal CPU usage levels, you can use this test to instantly drill down to the classes and the methods within the classes that contributing to the resource contention at the JVM.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the server being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.
Port	Refers to the port at which the specified host listens to.
Mode	This test can extract metrics from the Java application using either of the following mechanisms: • Using SNMP-based access to the Java runtime MIB statistics; • By contacting the Java runtime (JRE) of the application via JMX To configure the test to use SNMP, select the **SNMP** option. On the other hand, choose the **JMX** option to configure the test to use JMX instead. By default, the **JMX** option is chosen here.
JMX Remote Port	This parameter appears only if the Mode is set to **JMX**. Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_ Home>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
User, Password, and Confirm Password	These parameters appear only if the Mode is set to **JMX**. If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds if the mode is **JMX**, and **10** seconds if the mode is **SNMP**.

Parameter	Description
SNMPPort	This parameter appears only if the Mode is set to **SNMP**. Here specify the port number through which the server exposes its SNMP MIB. Ensure that you specify the same port you configured in the *management.properties* file in the <JAVA_ HOME>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
SNMPVersion	This parameter appears only if the Mode is set to **SNMP**. By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the SNMPversion list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
SNMPCommunity	This parameter appears only if the Mode is set to **SNMP**. The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the SNMPVersion chosen is **v3**, then this parameter will not appear.
UserName	This parameter appears only when **v3** is selected as the SNMPVersion. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against this parameter.
Context	This parameter appears only when v3 is selected as the SNMPVersion. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the SNMPEngineID value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a contextName). If the Username provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the UserName in the Context text box. By default, this parameter is set to *none*.
AuthPass	Specify the password that corresponds to the above-mentioned UserName. This parameter once again appears only if the SNMPversion selected is **v3**.
Confirm Password	Confirm the AuthPass by retyping it here.
AuthType	This parameter too appears only if **v3** is selected as the SNMPVersion. From the AuthType list box, choose the authentication algorithm using which SNMP v3 converts

Parameter	Description
	the specified username and password into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options: • **MD5** – Message Digest Algorithm • **SHA** – Secure Hash Algorithm
EncryptFlag	This flag appears only when **v3** is selected as the SNMPVersion. By default, the eG agent does not encrypt SNMP requests. Accordingly, the this flag is set to **No** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **Yes** option.
EncryptType	If this EncryptFlag is set to **Yes**, then you will have to mention the encryption type by selecting an option from the EncryptType list. SNMP v3 supports the following encryption types: • **DES** – Data Encryption Standard • **AES** – Advanced Encryption Standard
EncryptPassword	Specify the encryption password here.
Confirm Password	Confirm the encryption password by retyping it here.
Data Over TCP	This parameter appears only if the Mode is set to **SNMP**. By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set this flag to **Yes**. By default, this flag is set to **No**.
Profiler Home	JIP (Java Interactive Profiler) is a high performance, low overhead profiler that is written entirely in Java and used extensively to gather performance data pertaining to a JVM. The eG agent comes bundled with JIP, and takes the help of JIP to provide detailed diagnosis information related to the CPU usage of the JVM. To make sure that this test contacts JIP for detailed resource usage metrics, you need to indicate the location of the JIP in the Profiler Home text box. Typically, the files related to this profiler are available in <EG_INSTALL_DIR>\lib\jip directory (in Windows; in Unix, this will be: /opt/egurkha/lib/jip). If only a single Java application on a host is being monitored, then the JIP folder can remain in the same location. The Profiler Home parameter should then be configured with /opt/egurkha/lib/jip or <EG_INSTALL_DIR>\lib\jip, as the case may be. However, if

Parameter	Description
	more than one Java application on a single host is to be monitored, then first ensure that the JIP folder is copied to two different locations on the same host. The Profiler Home parameter should in this case be configured with the location of the JIP folder that corresponds to the Java application for which this test is being currently configured. For instance, say japp1 and japp2 are 2 Java applications that are being managed by the eG Enterprise system. Assume that the JIP folder has been copied to the *C:\japp1 and D:\japp2* folders. Now, while configuring this test for the japp1 application, specify C:\japp\jip in Profiler Home. Similarly, when configuring this test for the japp2 application, enter D:\japp2\jip in Profiler Home.
Profiler	The JIP can be turned on or off while the JVM is still running. For the eG agent to be able to report detailed diagnosis of the CPU usage metric, the Profiler should be turned on. Accordingly, the Profiler flag is set to **On** by default. If you do not want detailed diagnosis, then you can set the Profiler flag to **Off**.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against DD frequency.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
CPU utilization of JVM	Indicates the percentage of CPU currently utilized	Percent	Ideally, this value should be low. An

Measurement	Description	Measurement Unit	Interpretation
	by the JVM.		unusually high value or a consistent increase in this value is indicative of abnormal CPU usage, and could warrant further investigation. In such a situation, you can use the detailed diagnosis of this measure, if enabled, to determine which methods invoked by which classes are causing the steady/sporadic spikes in CPU usage.

3.1.6 JVM Memory Usage Test

This test monitors every memory type on the JVM and reports how efficiently the JVM utilizes the memory resources of each type.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every memory type on the JVM being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.
Port	Refers to the port at which the specified host listens to.
Mode	This test can extract metrics from the Java application using either of the following mechanisms: • Using SNMP-based access to the Java runtime MIB statistics; • By contacting the Java runtime (JRE) of the application via JMX To configure the test to use SNMP, select the **SNMP** option. On the other hand, choose the **JMX** option to configure the test to use JMX instead. By default, the **JMX** option is chosen here.
JMX Remote Port	This parameter appears only if the Mode is set to **JMX**. Here, specify the port at which

Parameter	Description
	the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_ Home>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
User, Password, and Confirm Password	These parameters appear only if the Mode is set to **JMX**. If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds if the mode is **JMX**, and **10** seconds if the mode is **SNMP**.
SNMPPort	This parameter appears only if the Mode is set to **SNMP**. Here specify the port number through which the server exposes its SNMP MIB. Ensure that you specify the same port you configured in the *management.properties* file in the <JAVA_ HOME>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
SNMPVersion	This parameter appears only if the Mode is set to **SNMP**. By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the SNMPversion list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
SNMPCommunity	This parameter appears only if the Mode is set to **SNMP**. The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the SNMPVersion chosen is **v3**, then this parameter will not appear.
UserName	This parameter appears only when **v3** is selected as the SNMPVersion. SNMP version

Parameter	Description
	3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against this parameter.
Context	This parameter appears only when v3 is selected as the SNMPVersion. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the SNMPEngineID value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a contextName). If the Username provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the UserName in the Context text box. By default, this parameter is set to *none*.
AuthPass	Specify the password that corresponds to the above-mentioned UserName. This parameter once again appears only if the SNMPversion selected is **v3**.
Confirm Password	Confirm the AuthPass by retyping it here.
AuthType	This parameter too appears only if **v3** is selected as the SNMPVersion. From the AuthType list box, choose the authentication algorithm using which SNMP v3 converts the specified username and password into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options: • **MD5** – Message Digest Algorithm • **SHA** – Secure Hash Algorithm
EncryptFlag	This flag appears only when **v3** is selected as the SNMPVersion. By default, the eG agent does not encrypt SNMP requests. Accordingly, the this flag is set to **No** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **Yes** option.
EncryptType	If this EncryptFlag is set to **Yes**, then you will have to mention the encryption type by selecting an option from the EncryptType list. SNMP v3 supports the following encryption types: • **DES** – Data Encryption Standard

Parameter	Description
	• **AES** – Advanced Encryption Standard
EncryptPassword	Specify the encryption password here.
Confirm Password	Confirm the encryption password by retyping it here.
Data Over TCP	This parameter appears only if the Mode is set to **SNMP**. By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set this flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Initial memory	Indicates the amount of memory initially allocated at startup.	MB	The detailed diagnosis of this measure, when enabled, reveals the memory pools on the JVM, the memory type of every pool, and the memory manager managing the pool.
Used memory	Indicates the amount of memory currently used.	MB	It includes the memory occupied by all objects including both reachable and unreachable objects.
Available memory	Indicates the amount of memory guaranteed to be available for use by the JVM.	MB	The amount of *Available memory* may change over time. The Java virtual machine may release memory to the system and committed memory could be less than the amount of memory initially allocated at startup. Committed will always be greater than or equal to used memory.
Free memory	Indicates the amount of memory currently available for use by the JVM.	MB	This is the difference between *Available memory* and *Used memory*. Ideally, the value of this measure should be high.

Measurement	Description	Measurement Unit	Interpretation
Max free memory	Indicates the maximum amount of memory allocated for the JVM.	MB	
Used percentage	Indicates the percentage of used memory.	Percent	Ideally, the value of this measure should be low. A very high value of this measure could indicate excessive memory consumption by the JVM, which in turn, could warrant further investigation.

3.1.7 JVM Uptime Test

This test helps track whether a scheduled reboot of the JVM has occurred or not.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the server being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.
Port	Refers to the port at which the specified host listens to.
Mode	This test can extract metrics from the Java application using either of the following mechanisms: • Using SNMP-based access to the Java runtime MIB statistics; • By contacting the Java runtime (JRE) of the application via JMX To configure the test to use SNMP, select the **SNMP** option. On the other hand, choose the **JMX** option to configure the test to use JMX instead. By default, the **JMX** option is chosen here.
JMX Remote Port	This parameter appears only if the Mode is set to **JMX**. Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port

Parameter	Description
	that you configured in the *management.properties* file in the <JAVA_ Home>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
User, Password, and Confirm Password	These parameters appear only if the Mode is set to **JMX**. If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds if the mode is **JMX**, and **10** seconds if the mode is **SNMP**.
SNMPPort	This parameter appears only if the Mode is set to **SNMP**. Here specify the port number through which the server exposes its SNMP MIB. Ensure that you specify the same port you configured in the *management.properties* file in the <JAVA_ HOME>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
SNMPVersion	This parameter appears only if the Mode is set to **SNMP**. By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the SNMPversion list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
SNMPCommunity	This parameter appears only if the Mode is set to **SNMP**. The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the SNMPVersion chosen is **v3**, then this parameter will not appear.
UserName	This parameter appears only when **v3** is selected as the SNMPVersion. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2

Parameter	Description
	Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against this parameter.
Context	This parameter appears only when v3 is selected as the SNMPVersion. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the SNMPEngineID value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a contextName). If the Username provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the UserName in the Context text box. By default, this parameter is set to *none*.
AuthPass	Specify the password that corresponds to the above-mentioned UserName. This parameter once again appears only if the SNMPversion selected is **v3**.
Confirm Password	Confirm the AuthPass by retyping it here.
AuthType	This parameter too appears only if **v3** is selected as the SNMPVersion. From the AuthType list box, choose the authentication algorithm using which SNMP v3 converts the specified username and password into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options: • **MD5** – Message Digest Algorithm • **SHA** – Secure Hash Algorithm
EncryptFlag	This flag appears only when **v3** is selected as the SNMPVersion. By default, the eG agent does not encrypt SNMP requests. Accordingly, the this flag is set to **No** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **Yes** option.
EncryptType	If this EncryptFlag is set to **Yes**, then you will have to mention the encryption type by selecting an option from the EncryptType list. SNMP v3 supports the following encryption types: • **DES** – Data Encryption Standard • **AES** – Advanced Encryption Standard

Parameter	Description
EncryptPassword	Specify the encryption password here.
Confirm Password	Confirm the encryption password by retyping it here.
Data Over TCP	This parameter appears only if the Mode is set to **SNMP**. By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set this flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Has the JVM been restarted?	Indicates whether or not the JVM has restarted during the last measurement period.		If the value of this measure is *No*, it indicates that the JVM has not restarted. The value *Yes* on the other hand implies that the JVM has indeed restarted. The numeric values that correspond to the reboot states discussed above are listed in the table below: **State** / **Value**: Yes – 1; No – 0 **Note:** By default, this measure reports the value Yes or No to indicate whether a JVM has restarted. The graph of this measure however, represents the same using the numeric equivalents – 0 or 1.
Uptime during the last measure period	Indicates the time period that the JVM has been up since the last time this test	Secs	If the JVM has not been restarted during the last measurement period

Measurement	Description	Measurement Unit	Interpretation
	ran.		and the agent has been running continuously, this value will be equal to the measurement period. If the JVM was restarted during the last measurement period, this value will be less than the measurement period of the test. For example, if the measurement period is 300 secs, and if the JVM was restarted 120 secs back, this metric will report a value of 120 seconds. The accuracy of this metric is dependent on the measurement period – the smaller the measurement period, greater the accuracy.
Total uptime of the JVM	Indicates the total time that the JVM has been up since its last reboot.		This measure displays the number of years, months, days, hours, minutes and seconds since the last reboot. Administrators may wish to be alerted if a JVM has been running without a reboot for a very long period. Setting a threshold for this metric allows administrators to determine such conditions.

3.1.8 JVM Garbage Collections Test

Manual memory management is time consuming, and error prone. Most programs still contain leaks. This is all doubly true with programs using exception-handling and/or threads. Garbage collection (GC) is a part of a Java application's JVM that automatically determines what memory a program is no longer using, and recycles it for other use. It is also known as "automatic storage (or memory) reclamation". This test reports the performance statistics pertaining to the JVM's garbage collection.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every garbage collector configured on the server being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.
Port	Refers to the port at which the specified host listens to.
Mode	This test can extract metrics from the Java application using either of the following mechanisms: • Using SNMP-based access to the Java runtime MIB statistics; • By contacting the Java runtime (JRE) of the application via JMX To configure the test to use SNMP, select the **SNMP** option. On the other hand, choose the **JMX** option to configure the test to use JMX instead. By default, the **JMX** option is chosen here.
JMX Remote Port	This parameter appears only if the Mode is set to **JMX**. Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_ Home>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
User, Password, and Confirm Password	These parameters appear only if the Mode is set to **JMX**. If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds if the mode is **JMX**, and **10** seconds if the mode is **SNMP**.

Parameter	Description
SNMPPort	This parameter appears only if the Mode is set to **SNMP**. Here specify the port number through which the server exposes its SNMP MIB. Ensure that you specify the same port you configured in the *management.properties* file in the <JAVA_HOME>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
SNMPVersion	This parameter appears only if the Mode is set to **SNMP**. By default, the eG agent supports SNMP version 1. Accordingly, the default selection in the SNMPversion list is **v1**. However, if a different SNMP framework is in use in your environment, say SNMP **v2** or **v3**, then select the corresponding option from this list.
SNMPCommunity	This parameter appears only if the Mode is set to **SNMP**. The SNMP community name that the test uses to communicate with the firewall. This parameter is specific to SNMP **v1** and **v2** only. Therefore, if the SNMPVersion chosen is **v3**, then this parameter will not appear.
UserName	This parameter appears only when **v3** is selected as the SNMPVersion. SNMP version 3 (SNMPv3) is an extensible SNMP Framework which supplements the SNMPv2 Framework, by additionally supporting message security, access control, and remote SNMP configuration capabilities. To extract performance statistics from the MIB using the highly secure SNMP v3 protocol, the eG agent has to be configured with the required access privileges – in other words, the eG agent should connect to the MIB using the credentials of a user with access permissions to be MIB. Therefore, specify the name of such a user against this parameter.
Context	This parameter appears only when v3 is selected as the SNMPVersion. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the SNMPEngineID value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a contextName). If the Username provided is associated with a context name, then the eG agent will be able to poll the MIB and collect metrics only if it is configured with the context name as well. In such cases therefore, specify the context name of the UserName in the Context text box. By default, this parameter is set to *none*.
AuthPass	Specify the password that corresponds to the above-mentioned UserName. This parameter once again appears only if the SNMPversion selected is **v3**.
Confirm Password	Confirm the AuthPass by retyping it here.
AuthType	This parameter too appears only if **v3** is selected as the SNMPVersion. From the AuthType list box, choose the authentication algorithm using which SNMP v3 converts

Parameter	Description
	the specified username and password into a 32-bit format to ensure security of SNMP transactions. You can choose between the following options: • **MD5** – Message Digest Algorithm • **SHA** – Secure Hash Algorithm
EncryptFlag	This flag appears only when **v3** is selected as the SNMPVersion. By default, the eG agent does not encrypt SNMP requests. Accordingly, the this flag is set to **No** by default. To ensure that SNMP requests sent by the eG agent are encrypted, select the **Yes** option.
EncryptType	If this EncryptFlag is set to **Yes**, then you will have to mention the encryption type by selecting an option from the EncryptType list. SNMP v3 supports the following encryption types: • **DES** – Data Encryption Standard • **AES** – Advanced Encryption Standard
EncryptPassword	Specify the encryption password here.
Confirm Password	Confirm the encryption password by retyping it here.
Data Over TCP	This parameter appears only if the Mode is set to **SNMP**. By default, in an IT environment, all data transmission occurs over UDP. Some environments however, may be specifically configured to offload a fraction of the data traffic – for instance, certain types of data traffic or traffic pertaining to specific components – to other protocols like TCP, so as to prevent UDP overloads. In such environments, you can instruct the eG agent to conduct the SNMP data traffic related to the monitored target over TCP (and not UDP). For this, set this flag to **Yes**. By default, this flag is set to **No**.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
No of garbage collections started	Indicates the number of times garbage collection started to release dead objects form memory.	Number	
Time taken for garbage collection	Indicates the time taken to perform the current	Secs	Ideally, the value of both these measures should be low. This is

Measurement	Description	Measurement Unit	Interpretation
	garbage collection operation.		because, the garbage collection (GC) activity tends to suspend the operations of the application until such time that GC ends. Longer the GC time, longer it would take for the application to resume its functions. To minimize the impact of GC on application performance, it is best to ensure that GC activity does not take too long to complete.
Percent of time spent by JVM for garbage collection	Indicates the percentage of time spent by JVM in garbage collection.	Percent	

3.1.9 JVM Memory Pool Garbage Collections Test

While the **JVM Garbage Collections** test reports statistics indicating how well each collector on the JVM performs garbage collection, the measures reported by the **JVM Memory Pool Garbage Collections** test help assess the impact of the garbage collection activity on the availability and usage of memory in each memory pool of the JVM. Besides revealing the count of garbage collections per collector and the time taken by each collector to perform garbage collection on the individual memory pools, the test also compares the amount of memory used and available for use pre and post garbage collection in each of the memory pools. This way, the test enables administrators to guage the effectiveness of the garbage collection activity on the memory pools, and helps them accurately identify those memory pools where enough memory could not reclaimed or where the garbage collectors spent too much time.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for every *GarbageCollector:MemoryPool* pair on the JVM of the server being monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port number at which the specified host listens to.

Parameter	Description
Measure Mode	This test allows you the option to collect the desired metrics using one of the following methodologies: • By contacting the Java runtime (JRE) of the application via JMX • Using GC logs To use JMX for metrics collections, set the Measure Mode to **JMX**. On the other hand, if you intend to use the GC log files for collecting the required metrics, set the measure mode to **Log File**. In this case, you would be required to enable GC logging. The procedure for this has been detailed in Enabling GC Logging section of the Monitoring Java Applications document.
JMX Remote port	This parameter appears only if the Measure Mode is set to **JMX**. Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_HOME>\jre\lib\management folder used by the target application (Refer to *Monitoring Java Applications* document).
JNDIName	This parameter appears only if the Measure Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is **jmxrmi**. If you have registered the JMX connector in the RMI registry using a different lookup name, then you can change this default value to reflect the same.
User, Password, and Confirm password	These parameters appear only if the Measure Mode is set to **JMX**. If JMX requires authentication only (but no security), then ensure that the user and password parameters are configured with the credentials of a user with read-write access to JMX. To know how to create this user, refer to Configuring the eG Agent to Support JMX Authentication. Confirm the password by retyping it in the Confirm Password text box.
JREHome	This parameter will be available only if the Measure Mode is set to **Log File**. Specify the full path to the Java Runtime Environment (JRE) used by the target application.
LogFilename	This parameter will be available only if the Measure Mode is set to **Log File**. Specify the full path to the GC log file to be used for metrics collection.
Provider	This parameter appears only if the Measure Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*.

Measurements made by the test

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td>Has garbage collection happened</td><td>Indicates whether garbage collection occurred on this memory pool in the last measurement period.</td><td></td><td>This measure reports the value Yes if garbage collection took place or No if it did not take place on the memory pool.

The numeric values that correspond to the measure values of Yes and No are listed below:

<table><tr><th>State</th><th>Value</th></tr><tr><td>Yes</td><td>1</td></tr><tr><td>No</td><td>0</td></tr></table>
Note:

By default, this measure reports the value Yes or No to indicate whether a GC occurred on a memory pool or not. The graph of this measure however, represents the same using the numeric equivalents – 0 or 1.</td></tr>
<tr><td>Collection count</td><td>Indicates the number of time in the last measurement pool garbage collection was started on this memory pool.</td><td>Number</td><td></td></tr>
<tr><td>Initial memory before GC</td><td>Indicates the initial amount of memory (in MB) that this memory pool requests from the operating system for memory management during startup, before GC process.</td><td>MB</td><td>Comparing the value of these two measures for a memory pool will give you a fair idea of the effectiveness of the garbage collection activity.

If garbage collection reclaims a large amount of memory from the memory pool, then the Initial memory after GC will drop. On the other hand, if the garbage collector does not reclaim much memory from a memory pool, or if the Java application suddenly runs a memory-intensive process when GC</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
			is being performed, then the Initial memory after GC may be higher than the Initial memory before GC.
Initial memory after GC	Indicates the initial amount of memory (in MB) that this memory pool requests from the operating system for memory management during startup, after GC process	MB	
Max memory before GC	Indicates the maximum amount of memory that can be used for memory management by this memory pool, before GC process.	MB	Comparing the value of these two measures for a memory pool will provide you with insights into the effectiveness of the garbage collection activity. If garbage collection reclaims a large amount of memory from the memory pool, then the Max memory after GC will drop. On the other hand, if the garbage collector does not reclaim much memory from a memory pool, or if the Java application suddenly runs a memory-intensive process when GC is being performed, then the Max memory after GC value may exceed the Max memory before GC.
Max memory after GC	Indicates the maximum amount of memory (in MB) that can be used for memory management by this pool, after the GC process.	MB	
Committed memory before GC	Indicates the amount of memory that is guaranteed to be available for use by this memory pool, before the GC process.	MB	
Committed memory after GC	Indicates the amount of memory that is guaranteed to be available for use by this memory pool, after the GC process.	MB	
Used memory before GC	Indicates the amount of memory used by this memory pool before GC.	MB	Comparing the value of these two measures for a memory pool will provide you with insights into the effectiveness of the garbage collection

Measurement	Description	Measurement Unit	Interpretation
			activity. If garbage collection reclaims a large amount of memory from the memory pool, then the Used memory after GC may drop lower than the Used memory before GC. On the other hand, if the garbage collector does not reclaim much memory from a memory pool, or if the Java application suddenly runs a memory-intensive process when GC is being performed, then the Used memory after GC value may exceed the Used memory before GC.
Used memory after GC	Indicates the amount of memory used by this memory pool after GC.	MB	
Percentage memory collected	Indicates the percentage of memory collected from this pool by the GC activity.	Percent	A high value for this measure is indicative of a large amount of unused memory in the pool. A low value on the other hand indicates that the memory pool has been over-utilized. Compare the value of this measure across pools to identify the pools that have very little free memory. If too many pools appear to be running short of memory, it could indicate that the target application is consuming too much memory, which in the long run, can slow down the application significantly.
Collection duration	Indicates the time taken by this garbage collector for collecting unused memory from this pool.	Mins	Ideally, the value of this measure should be low. This is because, the garbage collection (GC) activity tends to suspend the operations of the application until such time that GC ends. Longer the GC time, longer it would take for the application to resume its functions. To minimize the impact of GC on application performance, it is best to ensure that GC activity does not take too long to complete.

3.1.10 JMX Connection to JVM

This test reports the availability of the target Java application, and also indicates whether JMX is enabled on the application or not. In addition, the test promptly alerts you to slowdowns experienced by the application, and also reveals whether the application was recently restarted or not.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the server being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.
Port	Refers to the port at which the specified host listens to.
JMX Remote Port	Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_Home>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
User, Password, and Confirm Password	If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
JMX availability	Indicates whether the target application is available or not and whether JMX is enabled or not on the application.	Percent	If the value of this measure is 100%, it indicates that the Java application is available with JMX enabled. The value 0 on the other hand, could indicate one/both the following: • The Java application is unavailable; • The Java application is available, but JMX is not enabled;
JMX response time	Indicates the time taken to connect to the JMX agent of the Java application.	Secs	A high value could indicate a connection bottleneck.
Has the PID changed?	Indicates whether/not the process ID that corresponds to the Java application has changed.		This measure will report the value *Yes* if the PID of the target application has changed; such a change is indicative of an application restart. If the application has not restarted - i.e., if the PID has not changed - then this measure will return the value *No*.

3.1.11 JVM File Descriptors Test

This test reports useful statistics pertaining to file descriptors.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the Java application being monitored.

Configurable parameters for the test

Parameter	Description
Test Period	How often should the test be executed.
Host	The IP address of the host for which this test is to be configured.

Parameter	Description
Port	Refers to the port at which the specified host listens to.
JMX Remote Port	Here, specify the port at which the **JMX** listens for requests from remote hosts. Ensure that you specify the same port that you configured in the *management.properties* file in the <JAVA_Home>\jre\lib\management folder used by the target application (refer to the *Monitoring Java Applications* document for details).
User, Password, and Confirm Password	If JMX requires **authentication only** (but no security), then ensure that the User and Password parameters are configured with the credentials of a user with *read-write* access to JMX. To know how to create this user, refer to the *Monitoring Java Applications* document. Confirm the password by retyping it in the Confirm Password text box.
JNDIName	This parameter appears only if the Mode is set to **JMX**. The JNDIName is a lookup name for connecting to the JMX connector. By default, this is *jmxrmi*. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
Provider	This parameter appears only if the Mode is set to **JMX**. This test uses a JMX Provider to access the MBean attributes of the target Java application and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to **com.sun.jmx.remote.protocol**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the target Java application. If there is no response from the target beyond the configured duration, the test will timeout. By default, this is set to **240** seconds.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Open file descriptors in JVM	Indicates the number of file descriptors currently open for the application.	Number	
Maximum file descriptors in JVM	Indicates the maximum number of file descriptors allowed for the application.	Number	
File descriptor usage by JVM	Indicates the file descriptor usage in percentage.	Percent	

3.2 The Oracle JDBC Layer

The tests associated with this layer (see Figure 3.3) enable administrators to:

- Measure the health of the JDBC connections in every instance of the server
- Monitor the usage of the connection cache
- Determine the number and type of transactions executing on the server

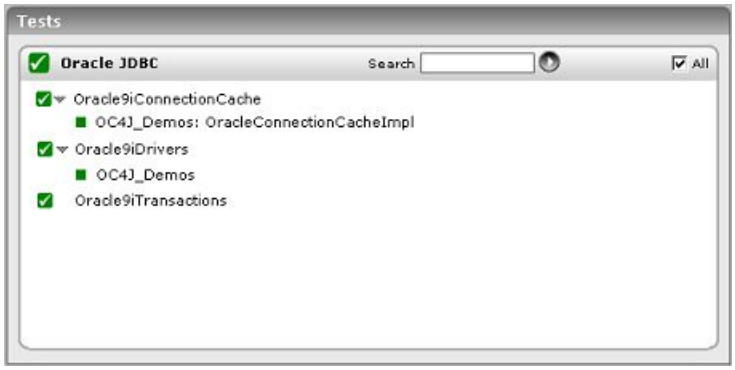

Figure 3.3: The tests associated with the Oracle JDBC layer

3.2.1 Oracle 9i Drivers Test

This test reports the performance metrics related to the JDBC Connection in an instance of the Oracle 9i application server.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every instance of the Oracle 9i AS monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the

Parameter	Description
	port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Threads creating connections	Indicates the current number of threads creating connections.	Number	
Avg connection create time	Indicates the average time spent creating connections during the last measurement period.	Secs	An increase in this metric could indicate a bottleneck in the database/database connection points.
Max connection create time	Indicates the high water mark indicating the maximum time spent creating connections since the server was started.	Secs	A high value for this measure is indicative of a bottleneck during database access.
Connections open	Indicates the number of connections that have been opened per second in the last measurement period.	Conns/Sec	A very high value for this measure indicates a heavy load on database access.
Connections closed	Indicates the number of connections that have been closed per second.	Conns/Sec	The value of this measure must be equal to the Connections open. If this value is consistently less than Connections open, then it indicates a potential problem - that connections are hanging on to the database server.

3.2.2 Oracle 9i Connection Cache Test

This test reports the performance metrics related to the connection cache of an instance of the Oracle9i application server.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every instance of the Oracle 9i AS monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Free cache size	Indicates the number of free slots in the connection cache.	Number	
Cache size	Indicates the total size of the connection cache.	Number	
Cache hits	Indicates the number of times a request for connection has been fulfilled by the cache.	Conns/Sec	Reading from the cache is less expensive than reading from disk. Therefore, the higher this value, the better.

Measurement	Description	Measurement Unit	Interpretation
Cache misses	Indicates the rate at which the cache failed to fulfill a request for connection.	Conns/Sec	Reading from the cache is less expensive than reading from disk. Therefore, the lower this value, the better.

3.2.3 Oracle 9i Transactions Test

This test reports the performance metrics related to the transactions occurring on the Oracle9i application server.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every instance of the Oracle 9i AS monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Active transactions	Indicates the current number of open	Number	A high value of this measure may indicate a large number of active

Measurement	Description	Measurement Unit	Interpretation
	transactions.		transactions. Alternatively, this may also indicate that due to some reasons the users are not able to complete the transactions.
Transactions commits	Indicates the rate at which the transactions were committed.	Trans/Sec	
Transaction rollbacks	Indicates the rate at which transactions wee rolled back during the last measurement period.	Trans/Sec	A high value here would mean that more number of transactions are being rolled back.

3.3 The Oracle Web Modules Layer

The test associated with this layer tracks the requests to every web module on each of the monitored Oracle 9i AS instances.

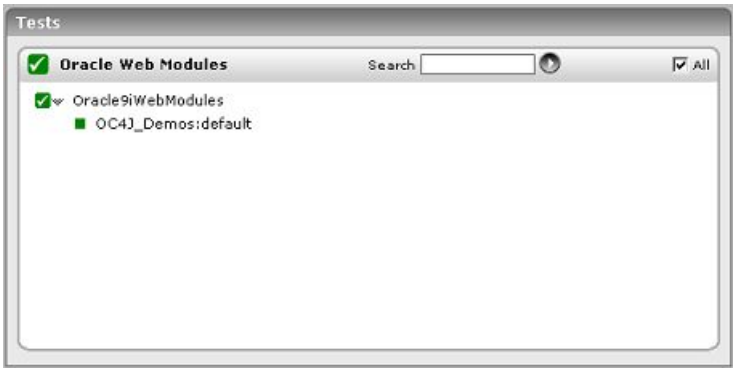

Figure 3.4: The tests associated with the Oracle Web Modules layer

3.3.1 Oracle 9i Web Modules Test

This test reports the performance metrics related to every web module in every instance of the Oracle9i application server.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every web module on every instance of the Oracle 9i AS monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Requests active	Indicates the current number of threads servicing web requests.	Number	A high value of this measure indicates a heavy load on the web module. An increase in requests running may indicate an increase in user workload. Alternatively, a slowdown of the application server may also cause the requests that are simultaneously executing to increase.
Requests completed	Indicates the average time spent servicing web requests during the last measurement period.	Secs	A very high value for this measure indicates that the web module is handling many requests.
Avg request process time	Indicates the average time spent servicing web requests during the last measurement period.	Trans/Sec	An increase in this value may be indicative of a problem in the application server. This increase could also be attributed to a problem in database tier provided the application

Measurement	Description	Measurement Unit	Interpretation
			is using the database for servicing the request.
Max request process time	Indicates the high water mark indicating the maximum time spend servicing a web request	Secs	
Avg context resolve time	Indicates the average time spent to create/find the servlet context during the last measurement period.	Secs	A very high value is an indication of heavy load on the web module.
Avg request parse time	Indicates the average time spent to read/parse requests during the last measurement.	Secs	Comparing the total request processing time with the context resolution time and request parsing time, can provide an indication of where a request is spending time in the O9i AS instance.

3.3.2 Web Service Test

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. A complete web service is, therefore, any service that:

- Is available over the Internet or private (intranet) networks
- Uses a standardized XML messaging system
- Is not tied to any one operating system or programming language
- Is self-describing via a common XML grammar
- Is discoverable via a simple find mechanism

The basic web services platform is XML + HTTP. All the standard web services work using the following components:

- SOAP (Simple Object Access Protocol)
- UDDI (Universal Description, Discovery and Integration)
- WSDL (Web Services Description Language)

A web service enables communication among various applications by using open standards such as HTML, XML, WSDL, and SOAP. A web service takes the help of the following:

- XML to tag the data
- SOAP to transfer a message
- WSDL to describe the availability of service.

The following are the major uses of the Web Services:

- **Reusable application-components**: Often applications need repeated access to application-components like currency conversion, weather reports, or even language translation. In such cases, the web services can be used to offer the application-components as services with ease.
- **Connect existing software**: Web services can help to solve the interoperability problem by giving different applications a way to link their data. With Web services you can exchange data between different applications and different platforms. Any application can have a Web Service component. Web Services can be created regardless of programming language.

In certain environments, administrators are required to keep an eye on the web services that offer repeated access to the application-components i.e., operations so that the work load on the users using those applicaiton components can be minimized. If for some reason the web service takes too long to respond or is unavailable to cater to the needs of the users, then the users will be deprived of access to the application-components involved in that particular web service. To avoid such inconvenience caused to the users, administrators are required to continuously monitor the web services. The **Web Service** test helps administrators to perform this task perfectly. By continuosly monitoring each operation i.e., application component of a web service that is offered, using the SOAP commands, this test helps administrators identify the availability, response time and response code of the web service and quickly figure out discrepancies if any web service is deemed unavailable. This way, the web services can be kept available round the clock thus helping the users perform their tasks without any difficulty.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for each WebService:Operation i.e., application-component performed on the target server that is being monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
WSDL URL	This test emulates a user accessing a specific web service(s) on the target server to determine the availability and responsiveness of the server. to enable this emulation, you need to configure the test with the URL of the web service that it should access. specify this URL against the WSDL URL parameter. if required, you can even configure multiple WSDL URLs - one each for every web service that the test should attempt to access. if each WSDL URL configured requires special permissions for logging in, then, you need to configure the test with separate credentials for logging into every WSDL URL. likewise, you need to provide instructions to the test on how to validate the content returned by every WSDL URL, and also set an encoding format for each WSDL URL. to enable administrators to easily configure the above per WSDL URL, eg enterprise provides a special interface. to access this interface, click on the encircled '+' button alongside the url text box in the test configuration page. alternatively, you can even click on the encircled '+' button adjacent to the WSDL URL parameter in the test configuration page. to know how to use this special interface, refer to Section **3.3.2.1**.
Operations	Once the WSDL URL(s) are specified, the operations that are offered by the web services and those that are to be monitored have to be configured. To select the required operations for monitoring, eG Enterprise provides a special interface. To access this interface, click on the encircled '+' button alongside the Operations text box in the test configuration page. Alternatively, you can even click on the encircled '+' button adjacent to the Operations parameter in the test configuration page. To know how to use this special interface, refer to Section **3.3.2**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the server. If there is no response from the server beyond the configured duration, the test

Parameter	Description
	will timeout. By default, this is set to 30 seconds
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
WSDL URL availability	Indicates whether the web service was able to respond successfully to the query made by the test.	Percent	Availability failures could be caused by several factors such as the web service process(es) being down, the web service being misconfigured, a network failure, etc. Temporary unavailability may also occur if the web service is overloaded. Availability is determined based on the response code returned by the service. A response code between 200 to 300 indicates that the service is available.
WSDL response time	Indicates the time taken by the eG agent to get the configured web service.	Secs	Response time being high denotes a problem. Poor response times may be due to the service being overloaded or misconfigured. If the URL accessed involves the generation of dynamic content by the service, backend problems (e.g., an overload at the application server or a database failure) can also result in an increase in response time.

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td>Port status</td><td>Indicates whether/not the port of the web server is reachable.</td><td></td><td>The values reported by this measure and the corresponding numeric equivalents are listed in the table below:
<table><tr><th>Measure Values</th><th>Numeric Values</th></tr><tr><td>Yes</td><td>1</td></tr><tr><td>No</td><td>0</td></tr></table>
Note:
By default, this measure reports the above-mentioned Measure Values to indicate whether the server has been rebooted or not. In the graph of this measure however, the Measure Values are represented using the numeric equivalents only.</td></tr>
<tr><td>TCP connection availability</td><td>Indicates whether the test managed to establish a TCP connection to the server.</td><td>Percent</td><td>Failure to establish a TCP connection may imply that either the web server process is not up, or that the process is not operating correctly. In some cases of extreme overload, the failure to establish a TCP connection may be a transient condition. As the load subsides, the server may start functioning properly again.</td></tr>
<tr><td>TCP connect time</td><td>This measure quantifies the time for establishing a TCP connection to the web server host.</td><td>Secs</td><td>Typically, the TCP connection establishment must be very small (of the order of a few milliseconds). Since TCP connection establishment is handled at the OS-level, rather than by the application, an increase in this value signifies a system-level bottleneck on the host that supports the web server.</td></tr>
<tr><td>Server response time</td><td>Indicates the time period</td><td>Secs</td><td>While the total response time may</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
	between when the connection was established and when the web server sent back a response header to the client.		depend on several factors, this measure is typically, a very good indicator of a server bottleneck (e.g., because all the available server threads or processes are in use).
Response code	The response code returned by the web server for the simulated request.	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (eg., page not found). A 5xx value indicates a server error.
Service availability	Indicates whether/not the web service is available.	Percent	A value of 100 indicates that the web service is available and a value of 0 indicates that the web service is not available.
Operation status	Indicates whether/not the configured operation is present in the web service.		This measure will not report metrics if the Operations parameter in the test configuration page is none in the test configuration page.
Operation Content length	Indicates the response code returned by the server for the simulated request.	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (e.g., page not found). A 5xx value indicates a server error. This measure will not report metrics if the OPERATION parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration page.
Operation Content validity	This measure validates whether the operation was successful in executing the request made to it.	Percent	A value of 100% indicates that the content returned by the test is valid. A value of 0% indicates that the content may not be valid. This capability for

Measurement	Description	Measurement Unit	Interpretation
			content validation is especially important for multi-tier web applications. For example, a user may not be able to login to the web site but the server may reply back with a valid HTML page where in the error message, say, “Invalid Login” is reported. In this case, the availability will be 100 % (since we got a valid HTML response). If the test is configured such that the content parameter should exclude the string “Invalid Login”, in the above scenario content validity would have a value 0. This measure will not report metrics if the Operations parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration page.
Operation execution time	Indicates the time taken to invoke the configured operation in the web service.	Secs	This measure will not report metrics if the OPERATION parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration page.

3.3.2.1 Configuring Multiple WSDL URLs for Monitoring

In order to enable the eG agent to connect to multiple WSDL URLs and pull out the required metrics from them, the eG administrative interface provides a special page using which different WSDL URLs and their corresponding operations that need to be monitored can be specified. To configure the WSDL URLs, do the following:

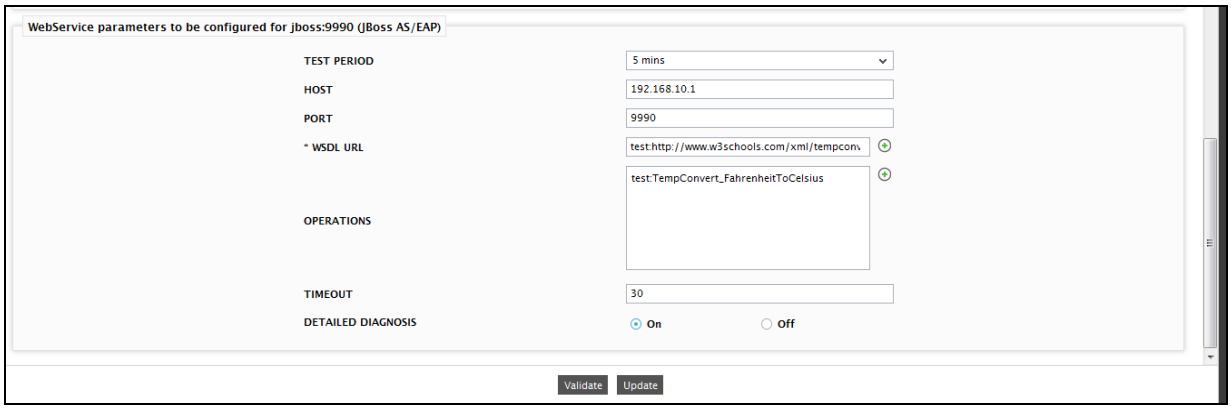

Figure 3.5: Configuring the WebService test

a. Click on the encircled '+' button alongside the WSDL URL text box. Figure 3.6 will then appear.

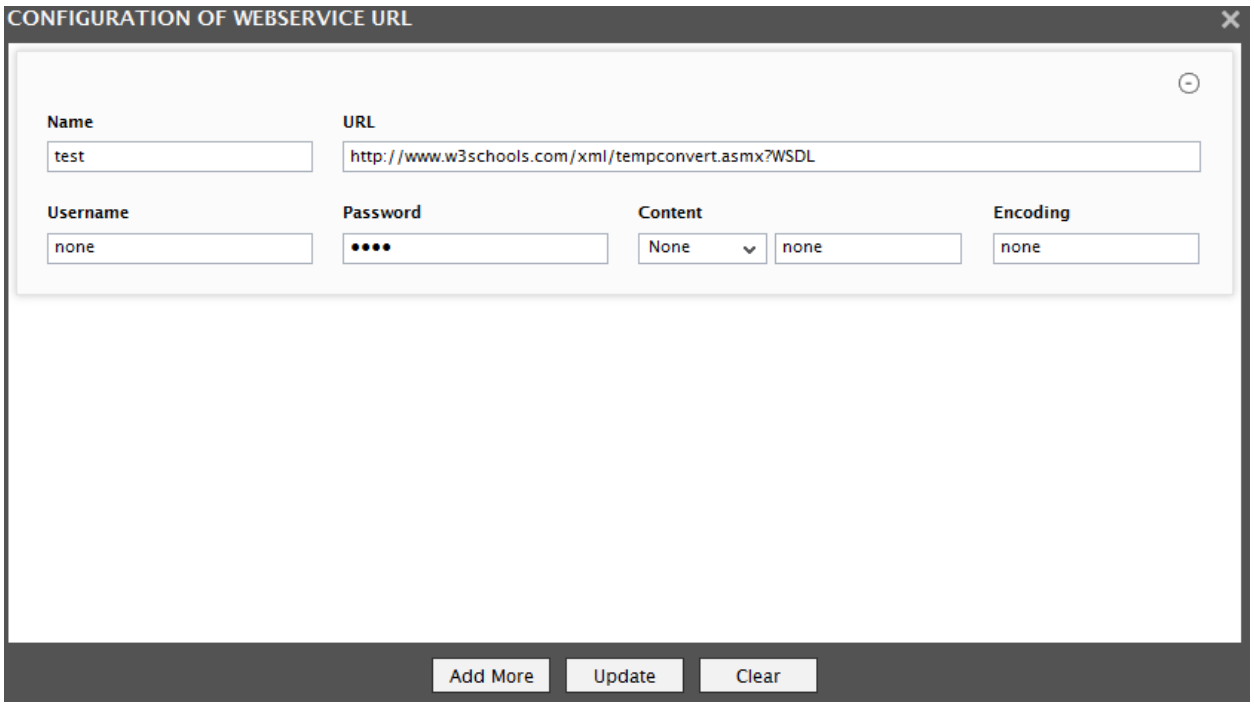

Figure 3.6: The WebService URL Configuration page

b. Specify the following Figure 3.6:

- **Name:** Specify a unique name by which the WSDL URL you will be specifiying shortly will be referred to across the eG user interface. This is the name that will appear as the descriptor of this test.
- **URL:** Enter the WSDL URL of the web service that this test should access.

- **Username** and **Password:** These parameters are to be set only if a specific user name / password has to be specified to login to the web service (i.e., WSDL URL) that you have configured for monitoring. In this case, provide valid login credentials using the **Username** and **Password** text boxes. If the server on which **WebService** test executes supports 'Anonymous user access', then these parameters will take either of the following values:
    - A valid **Username** and **Password** for the configured WSDL URL
    - *none* in both the **Username** and **Password** text boxes of the configured WSDL URL, if no user authorization is required
    - Some servers however, support NTLM (Integrated Windows) authentication, where valid login credentials are mandatory. In other words, a *none* specification will not be supported by such servers. Therefore, in this case, against each configured WSDL URL, you will have to provide a valid **Username** in the format: *domainname**username*, followed by a valid **Password**.
    - Please be sure to check if your web service requires HTTP authentication while configuring this parameter. HTTP authentication typically involves a separate pop-up window when you try to access the page. Many services use HTTP POST for obtaining the user name and password and validating the user login. In such cases, the username and password have to be provided as part of the POST information and NOT as part of the CREDENTIALS specification for the **Web Service** test.
- **Content**: The **Content** parameter has to be configured with an instruction:value pair that will be used to validate the content being returned by the test. If the **Content** value is None, no validation is performed. On the other hand, if you pick the Include option from the **Content** list, it indicates to the test that for the content returned by the web server to be valid, the content must include the specified value (a simple string search is done in this case). This value should be specified in the adjacent text box. Similarly, if the **Exclude** option is chosen from the **Content** drop-down, it indicates to the test that the server's output is valid if it does not contain the value specified in the adjacent text box. The **Include** or **Exclude** value you specify in the text box can include wildcard characters. For example, an Include instruction can be *Home page*.
- **Encoding**: Sometimes the eG agent has to parse the WSDL URL content with specific encoding other than the default (ISO-8859-1) encoding. In such a case, specify the type of encoding using which the eG agent can parse the WSDL URL content in the **Encoding** text box. By default, this value is *none*.

c. Similarly, you can add multiple URL specifications by clicking the **Add More** button. To remove a WSDL URL specification, click on the encircled '-' button corresponding to it. To clear all WSDL

URL specifications, click the **Clear** button. To update all the changes you made, click the **Update** button.

d. Once **Update** is clicked, you will return to the test configuration page as shown in Figure 3.5. The WSDL URL text box in the test configuration page will display just the **Name**s - i.e., the unique display names - that you may have configured for the multiple WSDL URLs, as a comma-separated list. To view the complete WSDL URL specification, click the encircled '+' button alongside the WSDL URL text box, once again.

3.3.2.2 Configuring Multiple Operations for Monitoring - WebServiceTest

By default, the **WebServiceTest** test will be configured with the WSDL URLs that offer the web services that are to be monitored. To configure the operations that are offered by the WSDL URLs, do the following:

a. Click on the encircled '+' button alongside the Operations text box as shown in Figure 3.5. Figure 3.7 will then appear.

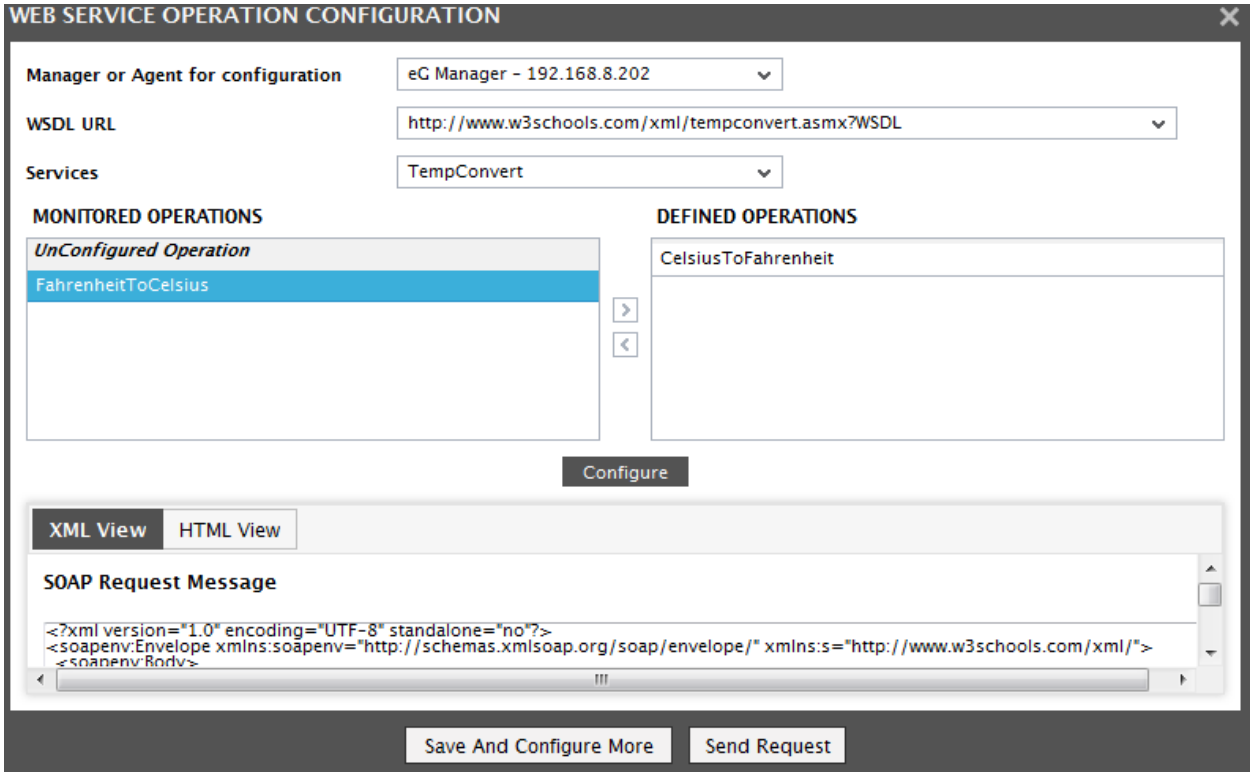

Figure 3.7: Configuring the Web Service Operation

b. Specify the following in Figure 3.7:

- **Manager/Agent for accessing WSDL URL**: Select the eG agent or the eG Manager that is authorized to access the configured WSDL URL from this list.

- **WSDL URL**: Once the eG agent/eG Manager is chosen from the Manager/Agent for accessing WSDL URL list, this list will be populated automatically with all the WSDL URLs specified in the **WSDL URL** text box (See Figure 3.5). Select the **WSDL URL** of your choice from this list.
- **Services**: The web services offered by the chosen WSDL URL will then be populated in this list. Select a service of your choice from this list.
    - The operations that are offered by the chosen service will then be populated in the **DEFINED OPERATIONS** list. To monitor a chosen operation, select the operation and click the < button. This will move the chosen operation to the **MONITORED OPERATIONS** list.
    - Click the **Configure** button to save the changes.
    - The eG agent uses SOAP requests to obtain the necessary metrics from the web service. Once the operation is configured, the XML View of the SOAP Request corresponding to the chosen operation will be generated and listed in the **XML View** tab. Likewise, the **HTML View** tab lists the **SOAP Parameter** that is passed to collect the required metrics for the chosen operation.
    - To obtain operation-level statistics, it is important to specify a valid value in the VALUE text box of the HTML View tab as shown in Figure 3.7. Each time the test is executed, this value will be provided as an input to the chosen operation.

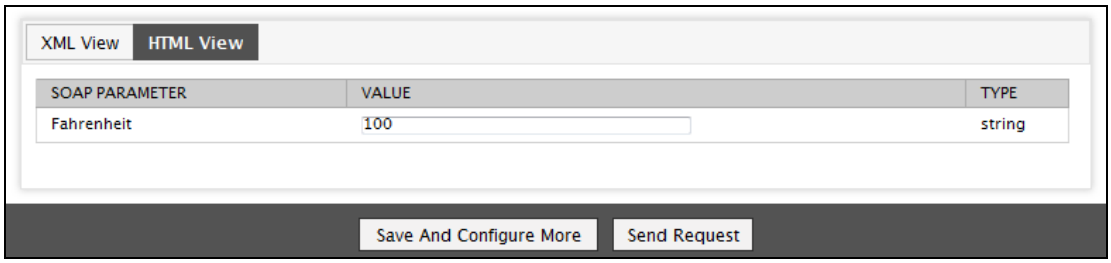

Figure 3.8: Specifying the value for the chosen operation

- Click the **Save and Configure More** button to save the changes made.
- If you wish to verify if the VALUE specified in the **HTML View** tab is valid, then you can do so by clicking the **Send Request** button. Figure 3.8 will then appear. If the value specified in the **VALUE** text box is indeed valid, then the operation will be performed on the value and the result will be specified. For example, if your chosen operation is FahrenheittoCelsius, the SOAP Parameter is Farenheit and the value that you wish to convert is 100, the result will be specified in the WEB SERVICE RESPONSE pop up window as below: <FahrenheitToCelsiusResult>37.7777777777778</FahrenheitToCelsiusResult>

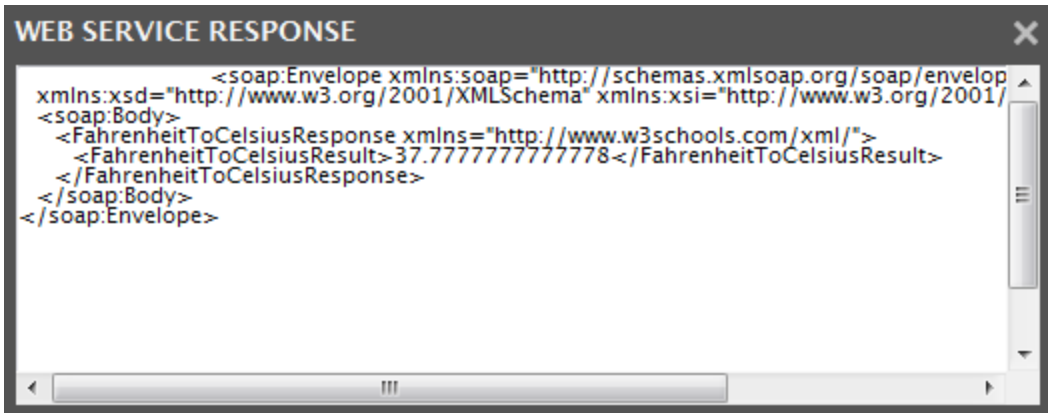

Figure 3.9: The value that appears when the operation is performed successfully

- If you have specified an invalid value, then a message as follows will be displayed in the pop up window: *<FahrenheitToCelsiusResult>Error</FahrenheitToCelsiusResult>*

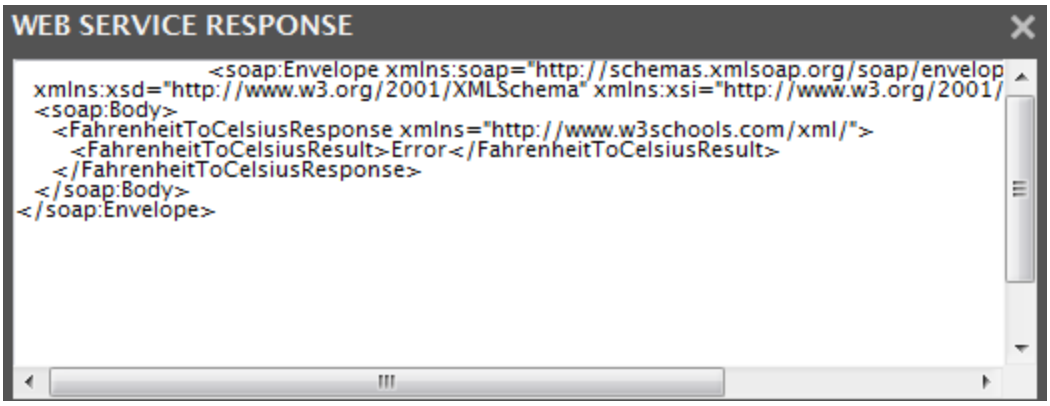

Figure 3.10: An Error appearing during value conversion

- If you do not specify a VALUE or specify an invalid value, operation-level statistics will not be collected by the eG agent and such metrics will not be available in the eG monitoring interface.

c. Similarly, you can configure multiple Operations by clicking the **Configure** button in Figure 3.7. To remove an operation, select the operation from the **MONITORED OPERATION** list and click the **>** button.

d. Once **Save and Configure More** button is clicked, you will return to the test configuration page (see Figure 3.5). The Operations text box in the test configuration page will display just the operations, as a comma-separated list. To view the complete operation specification, click the encircled '+' button alongside the Operations text box, once again.

3.3.3 Web Service Test

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. A complete web service is, therefore, any service that:

- Is available over the Internet or private (intranet) networks
- Uses a standardized XML messaging system
- Is not tied to any one operating system or programming language
- Is self-describing via a common XML grammar
- Is discoverable via a simple find mechanism

The basic web services platform is XML + HTTP. All the standard web services work using the following components:

- SOAP (Simple Object Access Protocol)
- UDDI (Universal Description, Discovery and Integration)
- WSDL (Web Services Description Language)

A web service enables communication among various applications by using open standards such as HTML, XML, WSDL, and SOAP. A web service takes the help of the following:

- XML to tag the data
- SOAP to transfer a message
- WSDL to describe the availability of service.

The following are the major uses of the Web Services:

- **Reusable application-components**: Often applications need repeated access to application-components like currency conversion, weather reports, or even language translation. In such cases, the web services can be used to offer the application-components as services with ease.
- **Connect existing software**: Web services can help to solve the interoperability problem by giving different applications a way to link their data. With Web services you can exchange data between different applications and different platforms. Any application can have a Web Service component. Web Services can be created regardless of programming language.

In certain environments, administrators are required to keep an eye on the web services that offer repeated access to the application-components i.e., operations so that the work load on the users using those applicaiton components can be minimized. If for some reason the web service takes too long to respond or is unavailable to cater to the needs of the users, then the users will be deprived of access to the application-components involved in that particular web service. To avoid such inconvenience caused to the users, administrators are required to continuously monitor the web services. The **Web Service** test helps administrators to perform this task perfectly. By continuosly monitoring each operation i.e., application component of a web service that is offered, using the SOAP commands, this test helps administrators identify the availability, response time and response code of the web service and quickly figure out discrepancies if any web service is deemed unavailable. This way, the web services can be kept available round the clock thus helping the users perform their tasks without any difficulty.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for each WebService:Operation i.e., application-component performed on the target server that is being monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
WSDL URL	This test emulates a user accessing a specific web service(s) on the target server to determine the availability and responsiveness of the server. to enable this emulation, you need to configure the test with the URL of the web service that it should access. specify this URL against the WSDL URL parameter. if required, you can even configure multiple WSDL URLs - one each for every web service that the test should attempt to access. if each WSDL URL configured requires special permissions for logging in, then, you need to configure the test with separate credentials for logging into

Parameter	Description
	every WSDL URL. likewise, you need to provide instructions to the test on how to validate the content returned by every WSDL URL, and also set an encoding format for each WSDL URL. to enable administrators to easily configure the above per WSDL URL, eg enterprise provides a special interface. to access this interface, click on the encircled '+' button alongside the url text box in the test configuration page. alternatively, you can even click on the encircled '+' button adjacent to the WSDL URL parameter in the test configuration page. to know how to use this special interface, refer to Section **3.3.3.1**.
Operations	Once the WSDL URL(s) are specified, the operations that are offered by the web services and those that are to be monitored have to be configured. To select the required operations for monitoring, eG Enterprise provides a special interface. To access this interface, click on the encircled '+' button alongside the Operations text box in the test configuration page. Alternatively, you can even click on the encircled '+' button adjacent to the Operations parameter in the test configuration page. To know how to use this special interface, refer to Section **3.3.3**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to 30 seconds
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
WSDL URL availability	Indicates whether the web service was able to respond successfully to	Percent	Availability failures could be caused by several factors such as the web service process(es) being down, the

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td>the query made by the test.</td><td></td><td>web service being misconfigured, a network failure, etc. Temporary unavailability may also occur if the web service is overloaded. Availability is determined based on the response code returned by the service. A response code between 200 to 300 indicates that the service is available.</td></tr>
<tr><td>WSDL response time</td><td>Indicates the time taken by the eG agent to get the configured web service.</td><td>Secs</td><td>Response time being high denotes a problem. Poor response times may be due to the service being overloaded or misconfigured. If the URL accessed involves the generation of dynamic content by the service, backend problems (e.g., an overload at the application server or a database failure) can also result in an increase in response time.</td></tr>
<tr><td>Port status</td><td>Indicates whether/not the port of the web server is reachable.</td><td></td><td>The values reported by this measure and the corresponding numeric equivalents are listed in the table below:
<table><tr><th>Measure Values</th><th>Numeric Values</th></tr><tr><td>Yes</td><td>1</td></tr><tr><td>No</td><td>0</td></tr></table>
Note:
By default, this measure reports the above-mentioned Measure Values to indicate whether the server has been rebooted or not. In the graph of this measure however, the Measure Values are represented using the numeric equivalents only.</td></tr>
<tr><td>TCP connection availability</td><td>Indicates whether the test</td><td>Percent</td><td>Failure to establish a TCP connection</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
	managed to establish a TCP connection to the server.		may imply that either the web server process is not up, or that the process is not operating correctly. In some cases of extreme overload, the failure to establish a TCP connection may be a transient condition. As the load subsides, the server may start functioning properly again.
TCP connect time	This measure quantifies the time for establishing a TCP connection to the web server host.	Secs	Typically, the TCP connection establishment must be very small (of the order of a few milliseconds). Since TCP connection establishment is handled at the OS-level, rather than by the application, an increase in this value signifies a system-level bottleneck on the host that supports the web server.
Server response time	Indicates the time period between when the connection was established and when the web server sent back a response header to the client.	Secs	While the total response time may depend on several factors, this measure is typically, a very good indicator of a server bottleneck (e.g., because all the available server threads or processes are in use).
Response code	The response code returned by the web server for the simulated request.	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (eg., page not found). A 5xx value indicates a server error.
Service availability	Indicates whether/not the web service is available.	Percent	A value of 100 indicates that the web service is available and a value of 0 indicates that the web service is not available.
Operation status	Indicates whether/not the configured operation is present in the web service.		This measure will not report metrics if the Operations parameter in the test configuration page is none in the test configuration page.

Measurement	Description	Measurement Unit	Interpretation
Operation Content length	Indicates the response code returned by the server for the simulated request.	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (e.g., page not found). A 5xx value indicates a server error. This measure will not report metrics if the OPERATION parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration page.
Operation Content validity	This measure validates whether the operation was successful in executing the request made to it.	Percent	A value of 100% indicates that the content returned by the test is valid. A value of 0% indicates that the content may not be valid. This capability for content validation is especially important for multi-tier web applications. For example, a user may not be able to login to the web site but the server may reply back with a valid HTML page where in the error message, say, “Invalid Login” is reported. In this case, the availability will be 100 % (since we got a valid HTML response). If the test is configured such that the content parameter should exclude the string “Invalid Login”, in the above scenario content validity would have a value 0. This measure will not report metrics if the Operations parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration

Measurement	Description	Measurement Unit	Interpretation
			page.
Operation execution time	Indicates the time taken to invoke the configured operation in the web service.	Secs	This measure will not report metrics if the OPERATION parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration page.

3.3.3.1 Configuring Multiple WSDL URLs for Monitoring

In order to enable the eG agent to connect to multiple WSDL URLs and pull out the required metrics from them, the eG administrative interface provides a special page using which different WSDL URLs and their corresponding operations that need to be monitored can be specified. To configure the WSDL URLs, do the following:

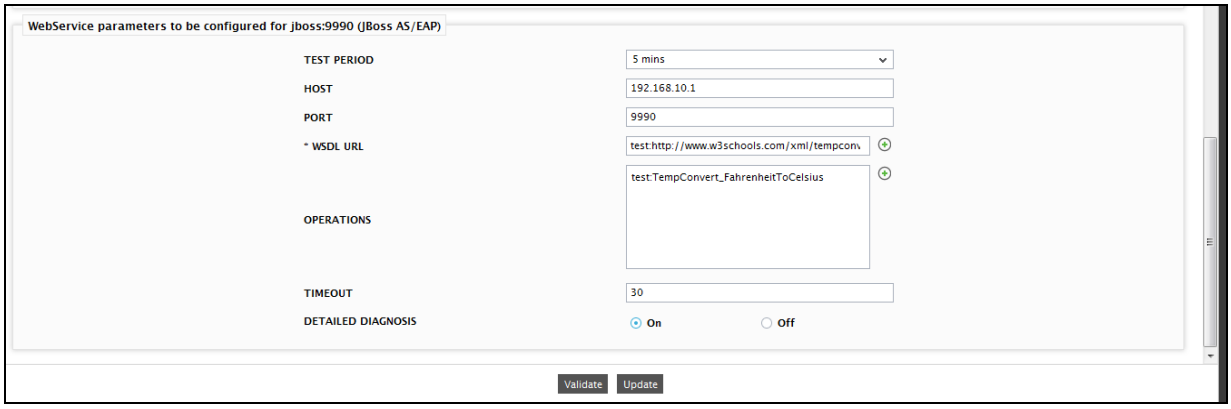

Figure 3.11: Configuring the WebService test

a. Click on the encircled '+' button alongside the WSDL URL text box. Figure 3.12 will then appear.

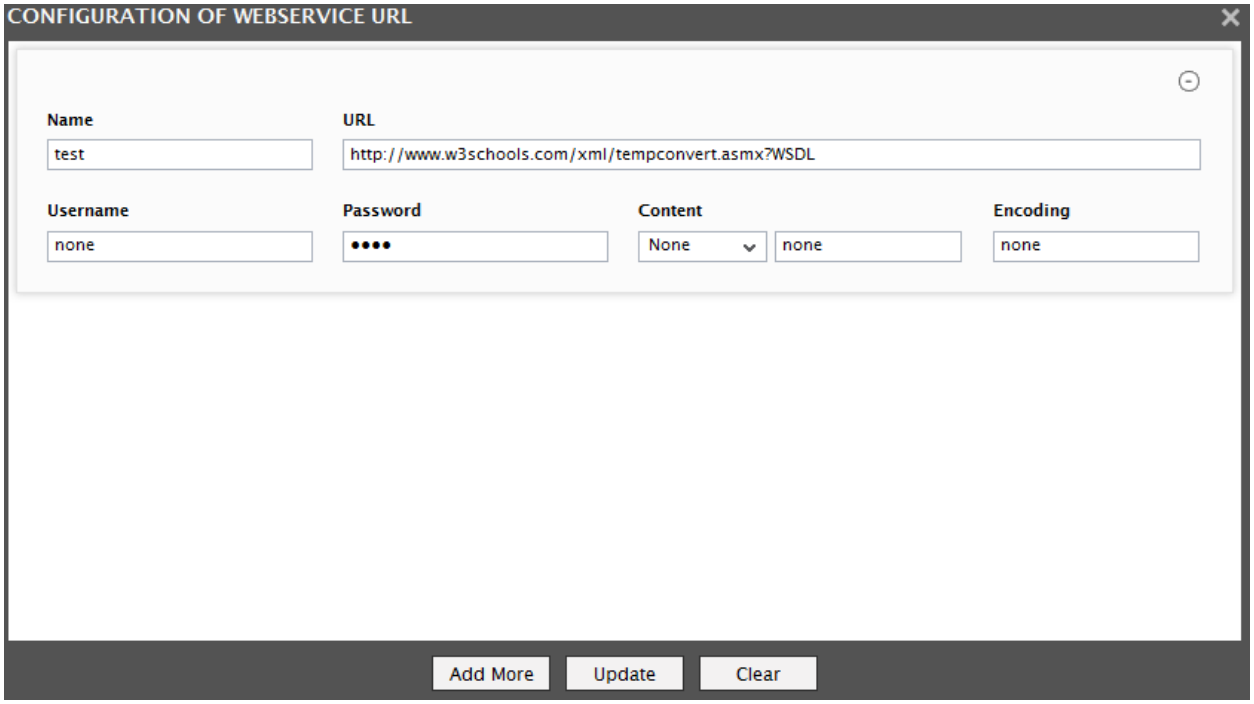

Figure 3.12: The WebService URL Configuration page

b. Specify the following Figure 3.12:

- **Name:** Specify a unique name by which the WSDL URL you will be specifiying shortly will be referred to across the eG user interface. This is the name that will appear as the descriptor of this test.
- **URL:** Enter the WSDL URL of the web service that this test should access.
- **Username** and **Password:** These parameters are to be set only if a specific user name / password has to be specified to login to the web service (i.e., WSDL URL) that you have configured for monitoring. In this case, provide valid login credentials using the **Username** and **Password** text boxes. If the server on which **WebService** test executes supports 'Anonymous user access', then these parameters will take either of the following values:
    - A valid **Username** and **Password** for the configured WSDL URL
    - *none* in both the **Username** and **Password** text boxes of the configured WSDL URL, if no user authorization is required
    - Some servers however, support NTLM (Integrated Windows) authentication, where valid login credentials are mandatory. In other words, a *none* specification

will not be supported by such servers. Therefore, in this case, against each configured WSDL URL, you will have to provide a valid **Username** in the format: *domainname**username*, followed by a valid **Password**.

- Please be sure to check if your web service requires HTTP authentication while configuring this parameter. HTTP authentication typically involves a separate pop-up window when you try to access the page. Many services use HTTP POST for obtaining the user name and password and validating the user login. In such cases, the username and password have to be provided as part of the POST information and NOT as part of the CREDENTIALS specification for the **Web Service** test.

- **Content**: The **Content** parameter has to be configured with an instruction:value pair that will be used to validate the content being returned by the test. If the **Content** value is None, no validation is performed. On the other hand, if you pick the Include option from the **Content** list, it indicates to the test that for the content returned by the web server to be valid, the content must include the specified value (a simple string search is done in this case). This value should be specified in the adjacent text box. Similarly, if the **Exclude** option is chosen from the **Content** drop-down, it indicates to the test that the server's output is valid if it does not contain the value specified in the adjacent text box. The **Include** or **Exclude** value you specify in the text box can include wildcard characters. For example, an Include instruction can be *Home page*.

- **Encoding**: Sometimes the eG agent has to parse the WSDL URL content with specific encoding other than the default (ISO-8859-1) encoding. In such a case, specify the type of encoding using which the eG agent can parse the WSDL URL content in the **Encoding** text box. By default, this value is *none*.

c. Similarly, you can add multiple URL specifications by clicking the **Add More** button. To remove a WSDL URL specification, click on the encircled '-' button corresponding to it. To clear all WSDL URL specifications, click the **Clear** button. To update all the changes you made, click the **Update** button.

d. Once **Update** is clicked, you will return to the test configuration page as shown in Figure 3.11. The WSDL URL text box in the test configuration page will display just the **Name**s - i.e., the unique display names - that you may have configured for the multiple WSDL URLs, as a comma-separated list. To view the complete WSDL URL specification, click the encircled '+' button alongside the WSDL URL text box, once again.

3.3.3.2 Configuring Multiple Operations for Monitoring - WebServiceTest

By default, the **WebServiceTest** test will be configured with the WSDL URLs that offer the web services that are to be monitored. To configure the operations that are offered by the WSDL URLs, do the following:

a. Click on the encircled '+' button alongside the Operations text box as shown in Figure 3.11. Figure 3.13 will then appear.

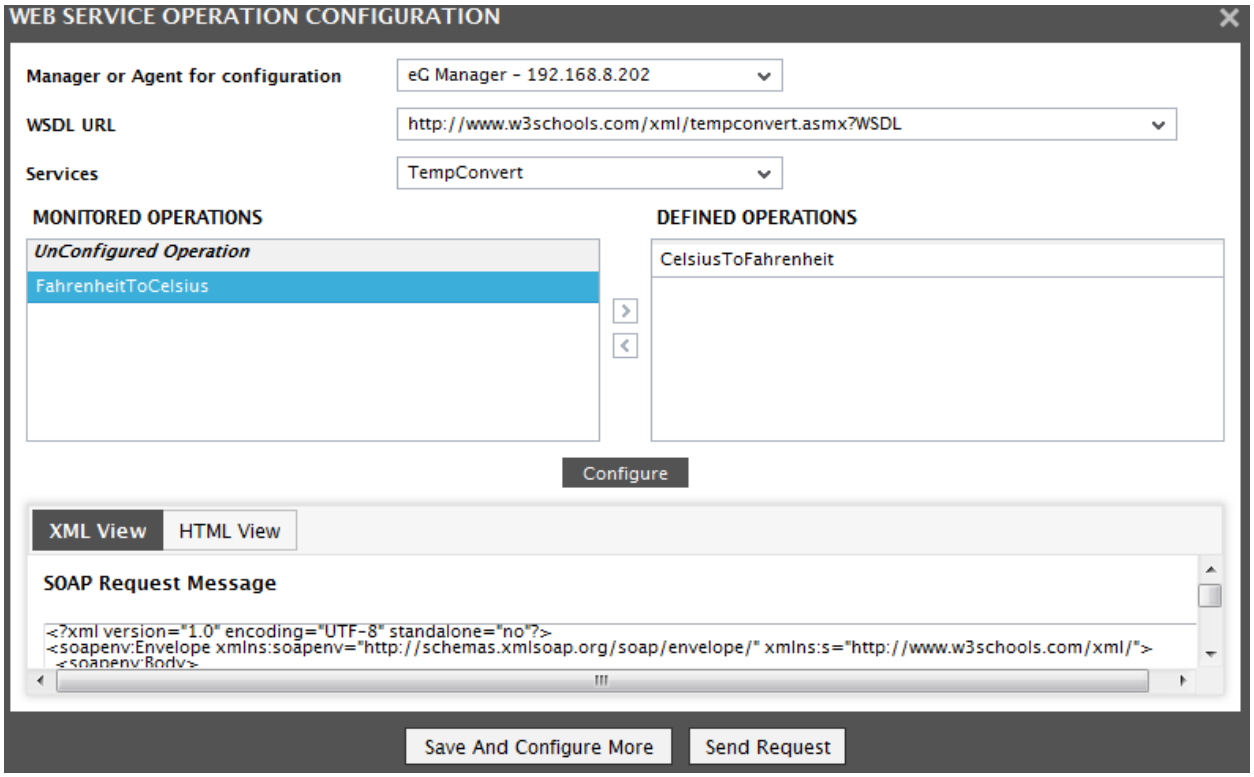

Figure 3.13: Configuring the Web Service Operation

b. Specify the following in Figure 3.13:

- **Manager/Agent for accessing WSDL URL**: Select the eG agent or the eG Manager that is authorized to access the configured WSDL URL from this list.
- **WSDL URL**: Once the eG agent/eG Manager is chosen from the Manager/Agent for accessing WSDL URL list, this list will be populated automatically with all the WSDL URLs specified in the **WSDL URL** text box (See Figure 3.11). Select the **WSDL URL** of your choice from this list.
- **Services**: The web services offered by the chosen WSDL URL will then be populated in this list. Select a service of your choice from this list.

- The operations that are offered by the chosen service will then be populated in the **DEFINED OPERATIONS** list. To monitor a chosen operation, select the operation and click the < button. This will move the chosen operation to the **MONITORED OPERATIONS** list.
- Click the **Configure** button to save the changes.
- The eG agent uses SOAP requests to obtain the necessary metrics from the web service. Once the operation is configured, the XML View of the SOAP Request corresponding to the chosen operation will be generated and listed in the **XML View** tab. Likewise, the **HTML View** tab lists the **SOAP Parameter** that is passed to collect the required metrics for the chosen operation.
- To obtain operation-level statistics, it is important to specify a valid value in the VALUE text box of the HTML View tab as shown in Figure 3.13. Each time the test is executed, this value will be provided as an input to the chosen operation.

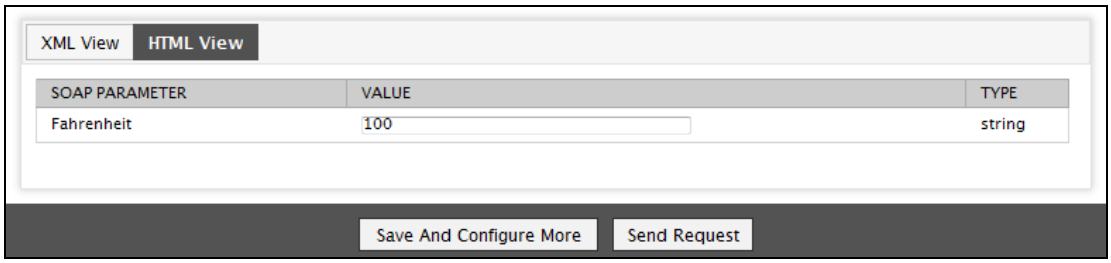

Figure 3.14: Specifying the value for the chosen operation

- Click the **Save and Configure More** button to save the changes made.
- If you wish to verify if the VALUE specified in the **HTML View** tab is valid, then you can do so by clicking the **Send Request** button. Figure 3.14 will then appear. If the value specified in the **VALUE** text box is indeed valid, then the operation will be performed on the value and the result will be specified. For example, if your chosen operation is FahrenheittoCelsius, the SOAP Parameter is Farenheit and the value that you wish to convert is 100, the result will be specified in the WEB SERVICE RESPONSE pop up window as below: <FahrenheitToCelsiusResult>37.7777777777778</FahrenheitToCelsiusResult>

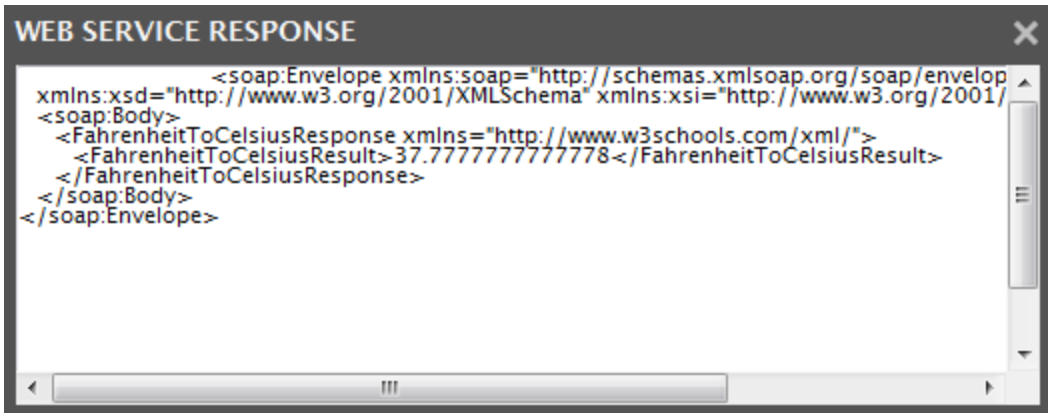

Figure 3.15: The value that appears when the operation is performed successfully

- If you have specified an invalid value, then a message as follows will be displayed in the pop up window: *<FahrenheitToCelsiusResult>Error</FahrenheitToCelsiusResult>*

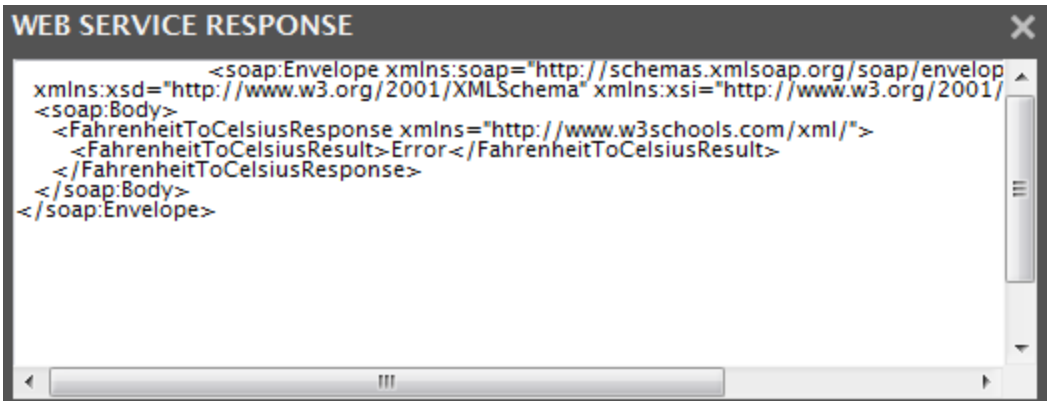

Figure 3.16: An Error appearing during value conversion

- If you do not specify a VALUE or specify an invalid value, operation-level statistics will not be collected by the eG agent and such metrics will not be available in the eG monitoring interface.

c. Similarly, you can configure multiple Operations by clicking the **Configure** button in Figure 3.13. To remove an operation, select the operation from the **MONITORED OPERATION** list and click the **>** button.

d. Once **Save and Configure More** button is clicked, you will return to the test configuration page (see Figure 3.11). The Operations text box in the test configuration page will display just the operations, as a comma-separated list. To view the complete operation specification, click the encircled '+' button alongside the Operations text box, once again.

3.3.4 Web Service Test

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. A complete web service is, therefore, any service that:

- Is available over the Internet or private (intranet) networks
- Uses a standardized XML messaging system
- Is not tied to any one operating system or programming language
- Is self-describing via a common XML grammar
- Is discoverable via a simple find mechanism

The basic web services platform is XML + HTTP. All the standard web services work using the following components:

- SOAP (Simple Object Access Protocol)
- UDDI (Universal Description, Discovery and Integration)
- WSDL (Web Services Description Language)

A web service enables communication among various applications by using open standards such as HTML, XML, WSDL, and SOAP. A web service takes the help of the following:

- XML to tag the data
- SOAP to transfer a message
- WSDL to describe the availability of service.

The following are the major uses of the Web Services:

- **Reusable application-components**: Often applications need repeated access to application-components like currency conversion, weather reports, or even language translation. In such cases, the web services can be used to offer the application-components as services with ease.
- **Connect existing software**: Web services can help to solve the interoperability problem by giving different applications a way to link their data. With Web services you can exchange data between different applications and different platforms. Any application can have a Web Service component. Web Services can be created regardless of programming language.

In certain environments, administrators are required to keep an eye on the web services that offer repeated access to the application-components i.e., operations so that the work load on the users using those applicaiton components can be minimized. If for some reason the web service takes too long to respond or is unavailable to cater to the needs of the users, then the users will be deprived of access to the application-components involved in that particular web service. To avoid such inconvenience caused to the users, administrators are required to continuously monitor the web services. The **Web Service** test helps administrators to perform this task perfectly. By continuosly monitoring each operation i.e., application component of a web service that is offered, using the SOAP commands, this test helps administrators identify the availability, response time and response code of the web service and quickly figure out discrepancies if any web service is deemed unavailable. This way, the web services can be kept available round the clock thus helping the users perform their tasks without any difficulty.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for each WebService:Operation i.e., application-component performed on the target server that is being monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
WSDL URL	This test emulates a user accessing a specific web service(s) on the target server to determine the availability and responsiveness of the server. to enable this emulation, you need to configure the test with the URL of the web service that it should access. specify this URL against the WSDL URL parameter. if required, you can even configure multiple WSDL URLs - one each for every web service that the test should attempt to access. if each WSDL URL configured requires special permissions for logging in, then, you need to configure the test with separate credentials for logging into

Parameter	Description
	every WSDL URL. likewise, you need to provide instructions to the test on how to validate the content returned by every WSDL URL, and also set an encoding format for each WSDL URL. to enable administrators to easily configure the above per WSDL URL, eg enterprise provides a special interface. to access this interface, click on the encircled '+' button alongside the url text box in the test configuration page. alternatively, you can even click on the encircled '+' button adjacent to the WSDL URL parameter in the test configuration page. to know how to use this special interface, refer to Section **3.3.4.1**.
Operations	Once the WSDL URL(s) are specified, the operations that are offered by the web services and those that are to be monitored have to be configured. To select the required operations for monitoring, eG Enterprise provides a special interface. To access this interface, click on the encircled '+' button alongside the Operations text box in the test configuration page. Alternatively, you can even click on the encircled '+' button adjacent to the Operations parameter in the test configuration page. To know how to use this special interface, refer to Section **3.3.4**.
Timeout	Specify the duration (in seconds) for which this test should wait for a response from the server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to 30 seconds
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
WSDL URL availability	Indicates whether the web service was able to respond successfully to	Percent	Availability failures could be caused by several factors such as the web service process(es) being down, the

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td>the query made by the test.</td><td></td><td>web service being misconfigured, a network failure, etc. Temporary unavailability may also occur if the web service is overloaded. Availability is determined based on the response code returned by the service. A response code between 200 to 300 indicates that the service is available.</td></tr>
<tr><td>WSDL response time</td><td>Indicates the time taken by the eG agent to get the configured web service.</td><td>Secs</td><td>Response time being high denotes a problem. Poor response times may be due to the service being overloaded or misconfigured. If the URL accessed involves the generation of dynamic content by the service, backend problems (e.g., an overload at the application server or a database failure) can also result in an increase in response time.</td></tr>
<tr><td>Port status</td><td>Indicates whether/not the port of the web server is reachable.</td><td></td><td>The values reported by this measure and the corresponding numeric equivalents are listed in the table below:
<table><tr><th>Measure Values</th><th>Numeric Values</th></tr><tr><td>Yes</td><td>1</td></tr><tr><td>No</td><td>0</td></tr></table>
Note:
By default, this measure reports the above-mentioned Measure Values to indicate whether the server has been rebooted or not. In the graph of this measure however, the Measure Values are represented using the numeric equivalents only.</td></tr>
<tr><td>TCP connection availability</td><td>Indicates whether the test</td><td>Percent</td><td>Failure to establish a TCP connection</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
	managed to establish a TCP connection to the server.		may imply that either the web server process is not up, or that the process is not operating correctly. In some cases of extreme overload, the failure to establish a TCP connection may be a transient condition. As the load subsides, the server may start functioning properly again.
TCP connect time	This measure quantifies the time for establishing a TCP connection to the web server host.	Secs	Typically, the TCP connection establishment must be very small (of the order of a few milliseconds). Since TCP connection establishment is handled at the OS-level, rather than by the application, an increase in this value signifies a system-level bottleneck on the host that supports the web server.
Server response time	Indicates the time period between when the connection was established and when the web server sent back a response header to the client.	Secs	While the total response time may depend on several factors, this measure is typically, a very good indicator of a server bottleneck (e.g., because all the available server threads or processes are in use).
Response code	The response code returned by the web server for the simulated request.	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (eg., page not found). A 5xx value indicates a server error.
Service availability	Indicates whether/not the web service is available.	Percent	A value of 100 indicates that the web service is available and a value of 0 indicates that the web service is not available.
Operation status	Indicates whether/not the configured operation is present in the web service.		This measure will not report metrics if the Operations parameter in the test configuration page is none in the test configuration page.

Measurement	Description	Measurement Unit	Interpretation
Operation Content length	Indicates the response code returned by the server for the simulated request.	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (e.g., page not found). A 5xx value indicates a server error. This measure will not report metrics if the OPERATION parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration page.
Operation Content validity	This measure validates whether the operation was successful in executing the request made to it.	Percent	A value of 100% indicates that the content returned by the test is valid. A value of 0% indicates that the content may not be valid. This capability for content validation is especially important for multi-tier web applications. For example, a user may not be able to login to the web site but the server may reply back with a valid HTML page where in the error message, say, “Invalid Login” is reported. In this case, the availability will be 100 % (since we got a valid HTML response). If the test is configured such that the content parameter should exclude the string “Invalid Login”, in the above scenario content validity would have a value 0. This measure will not report metrics if the Operations parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration

Measurement	Description	Measurement Unit	Interpretation
			page.
Operation execution time	Indicates the time taken to invoke the configured operation in the web service.	Secs	This measure will not report metrics if the OPERATION parameter in the test configuration page is none or if an invalid Value is specified or if the Value is not specified in the HTML View tab while configuring the operation for monitoring in the test configuration page.

3.3.4.1 Configuring Multiple WSDL URLs for Monitoring

In order to enable the eG agent to connect to multiple WSDL URLs and pull out the required metrics from them, the eG administrative interface provides a special page using which different WSDL URLs and their corresponding operations that need to be monitored can be specified. To configure the WSDL URLs, do the following:

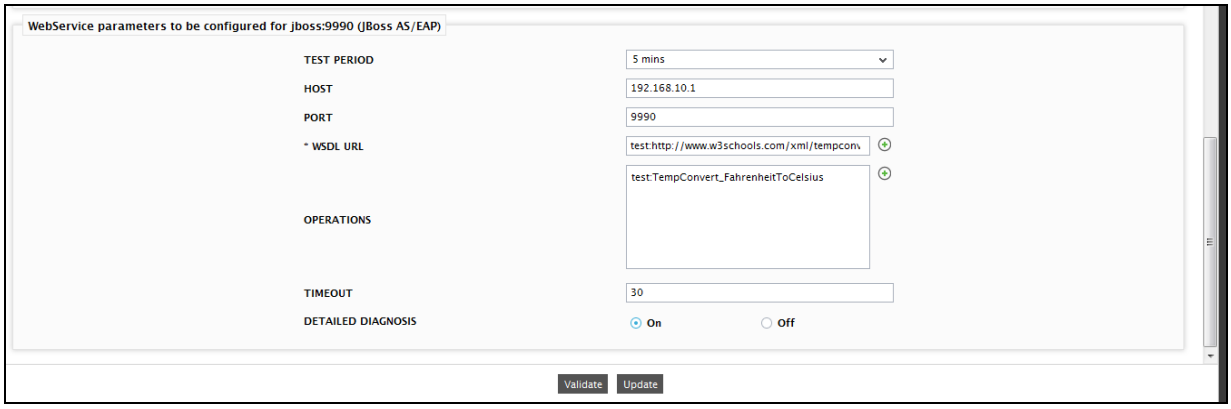

Figure 3.17: Configuring the WebService test

a. Click on the encircled '+' button alongside the WSDL URL text box. Figure 3.18 will then appear.

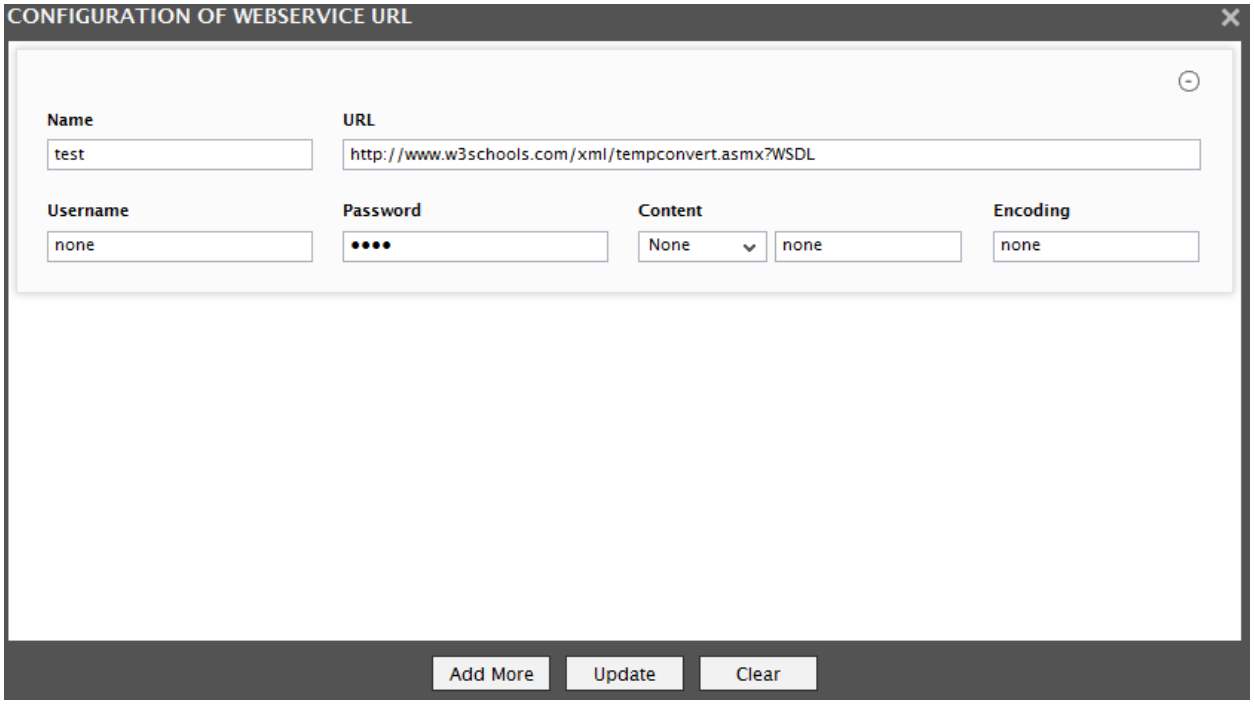

Figure 3.18: The WebService URL Configuration page

b. Specify the following Figure 3.18:

- **Name:** Specify a unique name by which the WSDL URL you will be specifiying shortly will be referred to across the eG user interface. This is the name that will appear as the descriptor of this test.
- **URL:** Enter the WSDL URL of the web service that this test should access.
- **Username** and **Password:** These parameters are to be set only if a specific user name / password has to be specified to login to the web service (i.e., WSDL URL) that you have configured for monitoring. In this case, provide valid login credentials using the **Username** and **Password** text boxes. If the server on which **WebService** test executes supports 'Anonymous user access', then these parameters will take either of the following values:
    - A valid **Username** and **Password** for the configured WSDL URL
    - *none* in both the **Username** and **Password** text boxes of the configured WSDL URL, if no user authorization is required
    - Some servers however, support NTLM (Integrated Windows) authentication, where valid login credentials are mandatory. In other words, a *none* specification

will not be supported by such servers. Therefore, in this case, against each configured WSDL URL, you will have to provide a valid **Username** in the format: *domainname**username*, followed by a valid **Password**.

- Please be sure to check if your web service requires HTTP authentication while configuring this parameter. HTTP authentication typically involves a separate pop-up window when you try to access the page. Many services use HTTP POST for obtaining the user name and password and validating the user login. In such cases, the username and password have to be provided as part of the POST information and NOT as part of the CREDENTIALS specification for the **Web Service** test.

- **Content**: The **Content** parameter has to be configured with an instruction:value pair that will be used to validate the content being returned by the test. If the **Content** value is None, no validation is performed. On the other hand, if you pick the Include option from the **Content** list, it indicates to the test that for the content returned by the web server to be valid, the content must include the specified value (a simple string search is done in this case). This value should be specified in the adjacent text box. Similarly, if the **Exclude** option is chosen from the **Content** drop-down, it indicates to the test that the server's output is valid if it does not contain the value specified in the adjacent text box. The **Include** or **Exclude** value you specify in the text box can include wildcard characters. For example, an Include instruction can be *Home page*.

- **Encoding**: Sometimes the eG agent has to parse the WSDL URL content with specific encoding other than the default (ISO-8859-1) encoding. In such a case, specify the type of encoding using which the eG agent can parse the WSDL URL content in the **Encoding** text box. By default, this value is *none*.

c. Similarly, you can add multiple URL specifications by clicking the **Add More** button. To remove a WSDL URL specification, click on the encircled '-' button corresponding to it. To clear all WSDL URL specifications, click the **Clear** button. To update all the changes you made, click the **Update** button.

d. Once **Update** is clicked, you will return to the test configuration page as shown in Figure 3.17. The WSDL URL text box in the test configuration page will display just the **Name**s - i.e., the unique display names - that you may have configured for the multiple WSDL URLs, as a comma-separated list. To view the complete WSDL URL specification, click the encircled '+' button alongside the WSDL URL text box, once again.

3.3.4.2 Configuring Multiple Operations for Monitoring - WebServiceTest

By default, the **WebServiceTest** test will be configured with the WSDL URLs that offer the web services that are to be monitored. To configure the operations that are offered by the WSDL URLs, do the following:

a. Click on the encircled '+' button alongside the Operations text box as shown in Figure 3.17. Figure 3.19 will then appear.

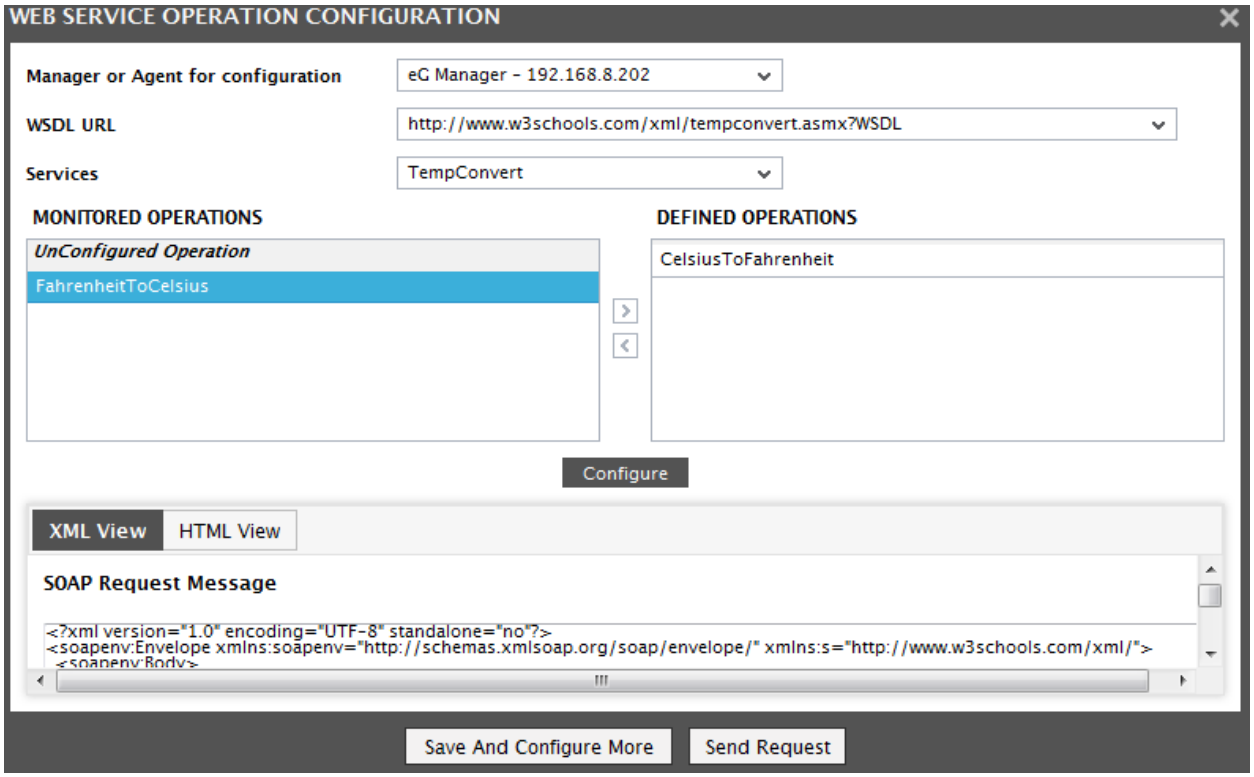

Figure 3.19: Configuring the Web Service Operation

b. Specify the following in Figure 3.19:

- **Manager/Agent for accessing WSDL URL**: Select the eG agent or the eG Manager that is authorized to access the configured WSDL URL from this list.
- **WSDL URL**: Once the eG agent/eG Manager is chosen from the Manager/Agent for accessing WSDL URL list, this list will be populated automatically with all the WSDL URLs specified in the **WSDL URL** text box (See Figure 3.17). Select the **WSDL URL** of your choice from this list.
- **Services**: The web services offered by the chosen WSDL URL will then be populated in this list. Select a service of your choice from this list.

- The operations that are offered by the chosen service will then be populated in the **DEFINED OPERATIONS** list. To monitor a chosen operation, select the operation and click the < button. This will move the chosen operation to the **MONITORED OPERATIONS** list.
- Click the **Configure** button to save the changes.
- The eG agent uses SOAP requests to obtain the necessary metrics from the web service. Once the operation is configured, the XML View of the SOAP Request corresponding to the chosen operation will be generated and listed in the **XML View** tab. Likewise, the **HTML View** tab lists the **SOAP Parameter** that is passed to collect the required metrics for the chosen operation.
- To obtain operation-level statistics, it is important to specify a valid value in the VALUE text box of the HTML View tab as shown in Figure 3.19. Each time the test is executed, this value will be provided as an input to the chosen operation.

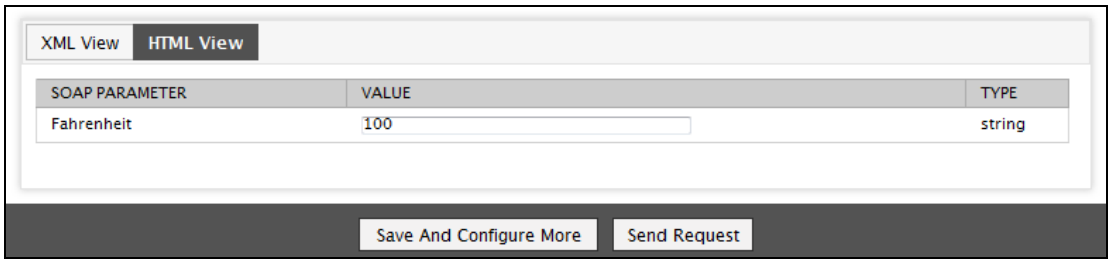

Figure 3.20: Specifying the value for the chosen operation

- Click the **Save and Configure More** button to save the changes made.
- If you wish to verify if the VALUE specified in the **HTML View** tab is valid, then you can do so by clicking the **Send Request** button. Figure 3.20 will then appear. If the value specified in the **VALUE** text box is indeed valid, then the operation will be performed on the value and the result will be specified. For example, if your chosen operation is FahrenheittoCelsius, the SOAP Parameter is Farenheit and the value that you wish to convert is 100, the result will be specified in the WEB SERVICE RESPONSE pop up window as below: <FahrenheitToCelsiusResult>37.7777777777778</FahrenheitToCelsiusResult>

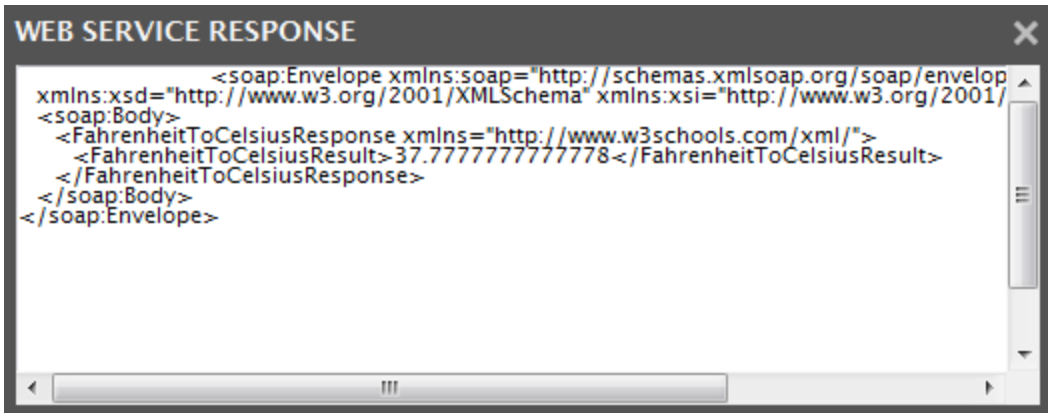

Figure 3.21: The value that appears when the operation is performed successfully

- If you have specified an invalid value, then a message as follows will be displayed in the pop up window: *<FahrenheitToCelsiusResult>Error</FahrenheitToCelsiusResult>*

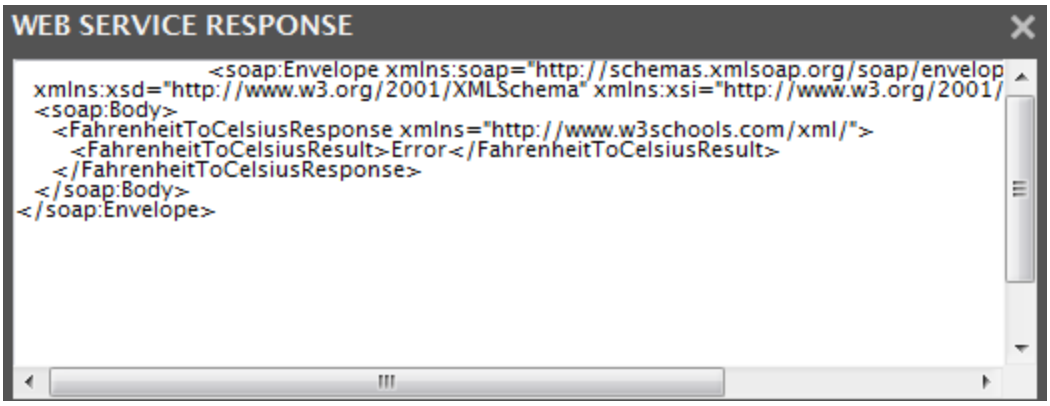

Figure 3.22: An Error appearing during value conversion

- If you do not specify a VALUE or specify an invalid value, operation-level statistics will not be collected by the eG agent and such metrics will not be available in the eG monitoring interface.

c. Similarly, you can configure multiple Operations by clicking the **Configure** button in Figure 3.19. To remove an operation, select the operation from the **MONITORED OPERATION** list and click the **>** button.

d. Once **Save and Configure More** button is clicked, you will return to the test configuration page (see Figure 3.17). The Operations text box in the test configuration page will display just the operations, as a comma-separated list. To view the complete operation specification, click the encircled '+' button alongside the Operations text box, once again.

3.4 The Oracle Web Context Layer

Using the test associated with this layer, administrators can monitor the session load on every web context on an Oracle 9i AS instance, and can determine whether any web context is taking too much time to create/locate a servlet instance.

Figure 3.23: The tests associated with the Oracle Web Context layer

3.4.1 Oracle 9i Web Contexts Test

This test reports the performance metrics related to every web context on each instance of the Oracle 9i AS.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every web context on each instance of the Oracle 9i AS monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port

Parameter	Description
	entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Active sessions	Indicates the current number of active sessions.	Number	A high value may indicate that there is a high load on the web context.
Avg session lifetime	Indicates the average session life time during the last measurement period.	Secs	A high value may indicate that a session that was opened on this web context may not be closed properly.
Avg servlet resolve time	Indicates the maximum time spent to create/locate the servlet instance (within the servlet context).	Secs	
Avg servlet resolve time	Indicates the average time spent to create/locate the servlet instance (within the servlet context) during the last measurement period.	Secs	

3.5 The Oracle J2EE Layer

To monitor the health of the JSPs and servlets on an Oracle 9i AS instance, use the tests associated with this layer (see Figure 3.24).

Figure 3.24: The tests associated with the Oracle J2EE layer

3.5.1 Oracle 9i Jsps Test

This test reports the performance metrics related to the JSPs deployed in an Oracle9i application server instance. In order to enable users to easily manage and monitor the JSPs deployed on an Oracle 9i application server, the eG Enterprise suite provides the Click Here hyperlink, which when clicked allows users to add, modify, or delete groups of JSPs. **Note that by default eG Enterprise monitors only those JSPs that are part of a group**.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every JSP group monitored or for every JSP auto-discovered (as the case may be).

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.
AutoDiscovery	By default, eG Enterprise allows administrators to configure JSP groups using the eG administrative interface, and reports metrics pertaining to every group so created. Accordingly, by default, AutoDiscovery is set to **No**. If you want JSPs to be discovered and monitored automatically, then select the **Yes** option against AutoDiscovery. When this is done, the eG agent automatically discovers all the JSPs on the server, and reports one set of measures for every JSP so discovered.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Threads serving JSPs	Indicates the current number of active requests for the JSP.	Number	If a majority of the threads/processes are in use simultaneously to serve requests for a specific JSP, this may be indicative of a problem with the design of this page. Further investigation is needed to determine the cause of the bottleneck - whether it is the processing of the JSP, whether it is a bottleneck in the database tier, whether the database query (ies) are non-optimal etc. Alternatively, a slowdown of the application server may also cause the requests that are simultaneously executing to increase.
Requests completed	Indicates the rate at which requests were processed by this JSP.	Reqs/Sec	Comparing this metric across JSPs can indicate which pages are most accessed. Optimizing the most heavily accessed pages can result in a significant improvement in the user-perceived performance of that web application.
Avg request process time	Indicates the average time spent in servicing the JSP during the last measurement period.	Secs	An increase in this value may be indicative of a problem in the application server/JSP logic.
Max request process time	Indicates the maximum time spent servicing a JSP since the server started.	Secs	

3.5.2 Oracle 9i Servlets Test

This test reports the performance metrics pertaining to the servlets deployed in an Oracle 9i AS instance. In order to enable users to easily manage and monitor servlets, the eG Enterprise suite provides the Click Here hyperlink, which when clicked allows users to add, modify, or delete servlet groups. **Note that by default eG Enterprise monitors only those servlets that are part of a group**.

Target of the test : An Oracle 9i Application Server

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every servlet group configured or for every discovered servlet (as the case may be) .

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.
AutoDiscovery	By default, eG Enterprise allows administrators to configure JSP groups using the eG administrative interface, and reports metrics pertaining to every group so created. Accordingly, by default, AutoDiscovery is set to **No**. If you want JSPs to be discovered and monitored automatically, then select the **Yes** option against AutoDiscovery. When this is done, the eG agent automatically discovers all the JSPs on the server, and reports one set of measures for every JSP so discovered.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Threads for servlet	Indicates the current number of threads servicing this servlet.	Number	If a majority of the threads/processes are in use simultaneously to serve requests for a specific servlet, this may be indicative of a problem with a specific servlet or one of the components used by it. Alternatively, a slowdown of the application server

Measurement	Description	Measurement Unit	Interpretation
			may also cause the requests that are simultaneously executing to increase for all servlets.
Requests completed	Indicates the rate of calls to the service() method.	Reqs/Sec	A very high value for this measure indicates that the servlet has been accessed many times.
Avg request process time	Indicates the average time spent in servicing the servlet during the last measurement period.	Secs	Comparing the request processing time across servlets, an administrator can determine which servlet(s) can be a performance bottleneck. An increase in response time of a servlet with load, could indicate a design problem with the servlet - eg., use of a non-optimal database query, database connection pool shortage, contention for shared resources, etc.
Max request process time	Indicates the maximum time spent on a servlet's service() call.	Secs	

Chapter 4: Monitoring Oracle 10g Application Servers

eG Enterprise provides out-of-the-box an Oracle Application monitoring model, that caters to the specific monitoring needs of the Oracle 10g application servers (see Figure 4.1).

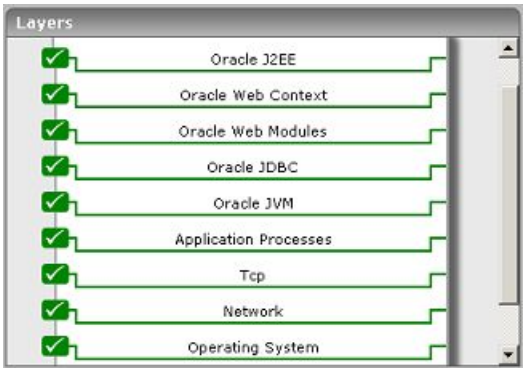

Figure 4.1: The layer model of the Oracle 10G application server

As you can see, the monitoring model depicted by Figure 4.1 is the same as that which eG Enterprise offers for the Oracle 9i Application server. The tests associated with every layer of Figure 4.1, and the metrics that each test reports, have therefore been discussed already in **Monitoring Oracle 9i Application Servers** chapter.

The difference however lies in the **Oracle J2EE** layer , which supports two additional tests for the *Oracle Application* model – the **Oracle EJBs** test and the **Oracle JMS Store** test.

Note:

To effectively monitor an Oracle application server on Unix, ensure that the install user of the application server and that of the eG agent (which is monitoring the application server) are the same.

Hence, the sections to come will discuss only the **Oracle J2EE** layer and the two tests associated with it.

4.1 The Oracle J2EE Layer

The tests associated with this layer monitor the health of the following:

- JSPs
- Servlets

- The EJB container
- Java Message Service (JMS)

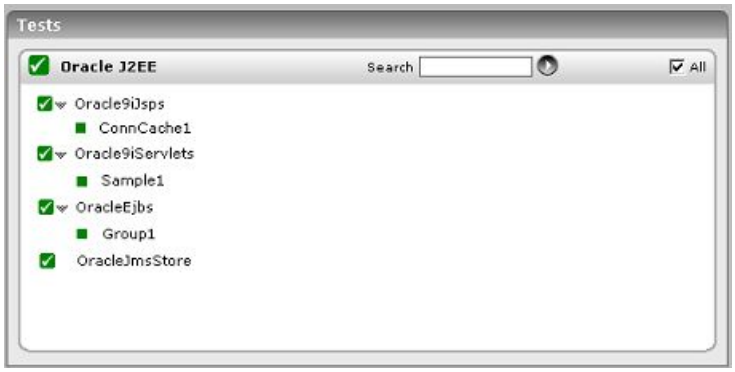

Figure 4.2: The tests associated with the Oracle J2EE layer

4.1.1 Oracle EJBs Test

This test monitors the state of EJB component groups hosted on an Oracle application server (10g or higher). Use the **Click here** hyperlink in the test configuration page to configure the EJB groups that need to be monitored by the eG Enterprise suite. **By default, the eG Enterprise system will monitor only those EJBs that are part of a group**.

Target of the test : An Oracle Application Server 10g or higher

Agent deploying the test : An internal agent

Outputs of the test : One set of results for every EJB group configured or every EJB auto-discovered.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port

Parameter	Description
	entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.
AutoDiscovery	By default, the eG suite allows administrators to configure EJB groups using the eG administrative interface, and reports metrics pertaining to every group so created. Accordingly, by default, AutoDiscovery is set to **No**. If you want EJBs to be discovered and monitored automatically, then select the **Yes** option against AutoDiscovery. When this is done, the eG agent automatically discovers all the EJBs on the application server, and reports one set of measures for every EJB on the server.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Avg activation time	Indicates the average time taken for activating this bean/ beans in this EJB group.	Secs	
Avg creation time	Indicates the average time taken for creating this bean/ beans in this EJB group.	Secs	
Avg passivation time	Indicates the average time taken for passivating this bean/ beans in this EJB	Secs	

Measurement	Description	Measurement Unit	Interpretation
	group.		
Avg post create time	Indicates the average time taken for a post create call on this bean/ beans in this EJB group.	Secs	
Avg removal time	Indicates the average time taken for removing this bean/ beans in this EJB group.	Secs	
Bean activations	Indicates the number of times this bean/ beans in this EJB group were activated.	Number	
Bean creations	Indicates the number of times this bean/beans in this EJB group were created.	Number	
Bean passivations	Indicates the number of times this bean/ beans in this EJB group were passivated.	Number	
Bean postcreations	Indicates the number of times this bean/ beans in this EJB group were post created.	Number	
Bean removals	Indicates the number of times this bean/ beans in this EJB group were removed.	Number	
Current threads activate	Indicates the number of threads that are currently used for activating this bean/ beans in this EJB group.	Number	
Current threads creation	Indicates the number of threads that are currently	Number	

Measurement	Description	Measurement Unit	Interpretation
	used for invoking a create call on this bean/ beans in this EJB group.		
Current threads passivate	Indicates the number of threads that are currently used for passivating this bean/ beans in this EJB group.	Number	
Current threads postcreate	Indicates the number of threads that are currently used for placing a postcreate call on this bean/ beans in this EJB group.	Number	
Current threads remove	Indicates the number of threads that are currently used for removing this bean/ beans in this EJB group.	Number	

4.1.2 Oracle JMS Store Test

This test monitors the health of the Java Message Service (JMS) store that the Oracle Enterprise Messaging Service (OEMS) supports. OEMS is a standards-based (JMS, JCA) messaging platform offering a comprehensive set of messaging and integration features for developing and deploying distributed applications in a service-oriented architecture (SOA) environment. OEMS also forms the underlying messaging infrastructure for Oracle's Enterprise Service Bus (ESB) and the BPEL Process Manager components of Oracle Fusion Middleware.

Target of the test : An Oracle Application Server 10g or higher

Agent deploying the test : An internal agent

Outputs of the test : One set of results for the Oracle Application server 10g that is monitored.

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	In the Port text box, it is recommended that you provide the port at which the OPMN (Oracle Process Manager and Notification) process of the Oracle application server instance listens. To know at which port OPMN listens, click on the Ports tab in the following URL: *http://<oraHttpServerIP>:<OraHttpServerport>*. This tab lists the port numbers that were assigned to the services executing on the Oracle application server. The port number displayed against the Oracle notification server request port entry is the OPMN port, and the same should be specified in the Port text box.
HomeDir	The absolute path of the directory in which the Oracle 9i application server has been installed.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Messages count	Indicates the total number of messages in the store - both committed and rolledback.	Number	
Messages dequeued	Indicates the number of dequeued messages.	Number	
Messages discarded	Indicates the number of messages discarded from the store.	Number	
Messages enqueued	Indicates the number of enqueued messages.	Number	
Messages expired	Indicates the number of expired messages.	Number	
Messages paged in	Indicates the number of paged in messages.	Number	
Messages paged out	Indicates the number of paged out messages.	Number	
Messages pending	Indicates the number of	Number	

Measurement	Description	Measurement Unit	Interpretation
	messages pending completion.		
Messages recovered	Indicates the number of recovered messages.	Number	
Messages rolledback	Indicates the number of messages that were rolled back.	Number	
Store size	Indicates the size of the JMS store.	MB	

4.2 Troubleshooting the Oracle Application server

If the Oracle Application server that is being monitored is not reporting measures, then check the following:

- Check if the **dmstool** is available in opt/app/Oracle/product/10.1.3/OracleAS_1/bin/dmstool; if not, then the Oracle application server-related tests will not run.
- Next, check if the eG user has permission to run the **dmstool** utility. If not, you will either need to install the eG agent as the Oracle application server user, or you will need to ensure that the eG user is in the same group as the Oracle application server user.

About eG Innovations

eG Innovations provides intelligent performance management solutions that automate and dramatically accelerate the discovery, diagnosis, and resolution of IT performance issues in on-premises, cloud and hybrid environments. Where traditional monitoring tools often fail to provide insight into the performance drivers of business services and user experience, eG Innovations provides total performance visibility across every layer and every tier of the IT infrastructure that supports the business service chain. From desktops to applications, from servers to network and storage, from virtualization to cloud, eG Innovations helps companies proactively discover, instantly diagnose, and rapidly resolve even the most challenging performance and user experience issues.

eG Innovations is dedicated to helping businesses across the globe transform IT service delivery into a competitive advantage and a center for productivity, growth and profit. Many of the world's largest businesses use eG Enterprise to enhance IT service performance, increase operational efficiency, ensure IT effectiveness and deliver on the ROI promise of transformational IT investments across physical, virtual and cloud environments.

To learn more visit www.eginnovations.com.

Contact Us

For support queries, email support@eginnovations.com.

To contact eG Innovations sales team, email sales@eginnovations.com.