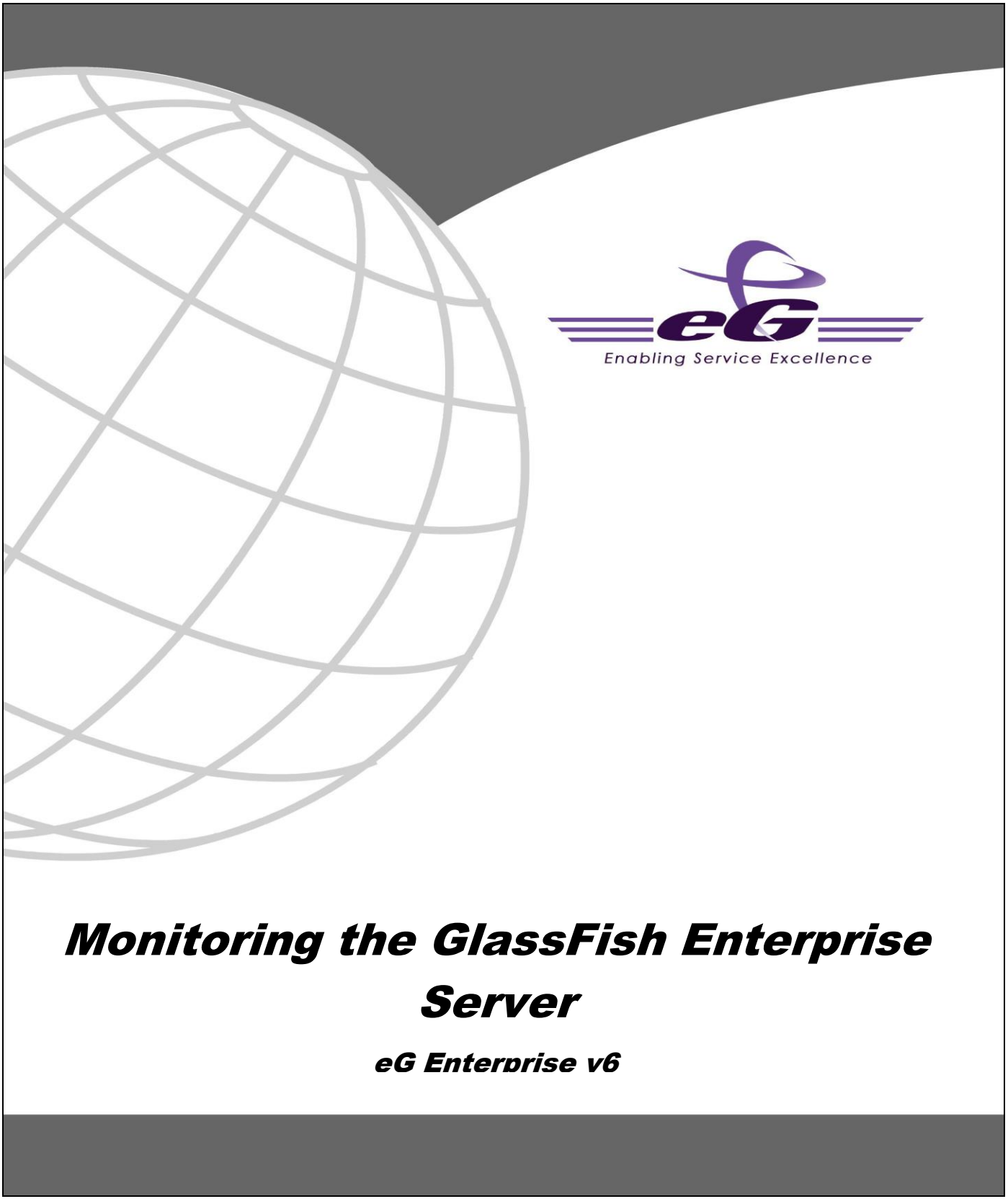

Monitoring the GlassFish Enterprise Server

eG Enterprise v6

Restricted Rights Legend

The information contained in this document is confidential and subject to change without notice. No part of this document may be reproduced or disclosed to others without the prior permission of eG Innovations, Inc. eG Innovations, Inc. makes no warranty of any kind with regard to the software and documentation, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

Trademarks

Microsoft Windows, Windows NT, Windows 2000, Windows 2003 and Windows 2008 are either registered trademarks or trademarks of Microsoft Corporation in United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Copyright

The copyright in this document belongs to eG Innovations, Inc. Complying with all applicable copyright laws is the responsibility of the user.

Table of Contents

Table of Figures

Chapter 1

Monitoring the Oracle GlassFish Enterprise Server

Oracle GlassFish Server is the world's first implementation of the Java Platform, Enterprise Edition (Java EE) 6 specification. Built using the GlassFish Server Open Source Edition, Oracle GlassFish Server delivers a flexible, lightweight and production-ready Java EE 6 platform.

Oracle GlassFish Server is based on the Java EE 6 Reference Implementation and is the first application server to support the full Java EE 6 platform and the new Java EE 6 Web Profile, which is designed specifically for Web applications. Oracle GlassFish Server delivers a new, efficient model for developing and deploying production-level applications. Developers can begin with just the Web Profile and grow to the full Java EE 6 platform when needed. Because Oracle GlassFish Server runs on the OSGi runtime, modular features can be added as necessary. Developers can also create hybrid applications - using OSGi services from Java EE Applications or using Java EE services from OSGi services. It also keeps the footprint as small as possible by loading only modules required to service deployed applications, improving startup time and reducing resource utilization.

Because it is swift and resource-efficient, the Oracle GlassFish server is the preferred middle-ware for mission-critical IT infrastructures where performance and prompt delivery of services to end-users is of utmost importance. In such environments, if the service level guarantees of the GlassFish server are not fulfilled, it is sure to result in dissatisfied users, severe penalties, and loss of revenue and reputation. You hence need to continuously monitor the performance of the GlassFish server, so that service level slippages are minimized and user satisfaction with the overlying business services is maximized.

eG Enterprise provides a specialized *GlassFish* monitoring model that monitors the health of the Oracle GlassFish server inside-out and promptly captures and reports abnormalities.

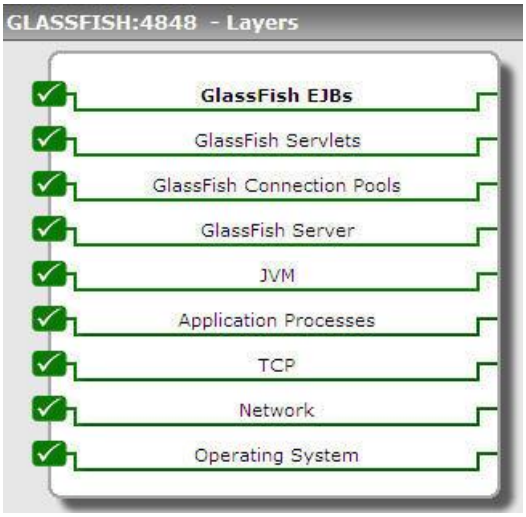

Figure 1: The layer model of the GlassFish server

Every layer of Figure 1 above is mapped to a variety of tests; each of these tests can be configured to use *agent-based* or *agentless* methodologies to report on the health of key components of the application server such as the servlets, EJBs, connection pools, caches, the JVM, and more!

To enable the eG agent to execute these tests, you need to make sure that the following pre-requisites are fulfilled:

a. The eG agent should be configured to use **JMX** to connect to the **JRE** of the GlassFish server and collect the metrics of interest. By default, **JMX** support is enabled for GlassFish. If, for some reason, **JMX** is disabled for the GlassFish server in your environment, then, follow the steps below to enable it:

   - In an editor, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below:

     ```
     <jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="false" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">
     ```

   - If **JMX** is disabled for the GlassFish server, then the **enabled** parameter in the code block above will be set to **false**. To enable JMX, set the **enabled** parameter to **true** as shown below:

     ```
     <jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">
     ```

   - Against the **port** parameter, configure the port at which JMX listens. The default JMX port is **8686**.
   - Finally, save the file.

b. Since **JMX** on GlassFish requires authentication by default, every test executed by the eG agent should be configured with the credentials of an *Administrator* to the GlassFish server. If you prefer not to expose the *administrator* credentials owing to security compulsions, then configure the test with the credentials of any valid user who has the right to use JMX. To know the name of such a user, do the following:

- In an editor, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below:

```
<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">
```

- The user name specified against the **auth-realm-name** parameter in the code block above can be configured as the **USER**, and the password of that user can be specified against **JMX PASSWORD**.

c. You should enable the **Monitoring** service of the GlassFish server. To achieve this, do the following:

- Connect to the GlassFish server using the URL: http://<GlassFishServer_IP>:<GlassFishServer_Port>/
- Login to the server as *administrator*.
- Figure 2 will then appear. Expand the **Configuration** node in the tree-structure in the left panel of Figure 2, and select the **Monitoring** sub-node within.

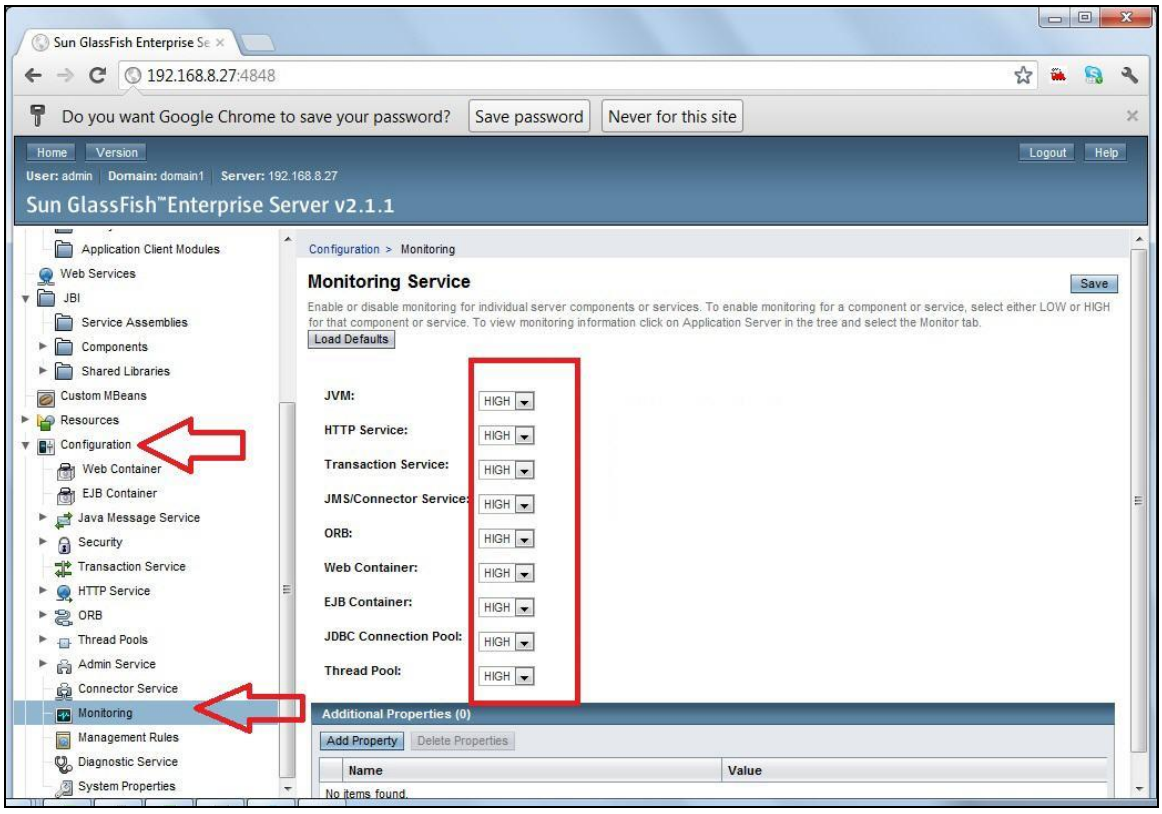

Figure 2: Enabling the Monitoring Service

- Set the monitoring-level for all the server components listed in the right panel to **HIGH**.
- Finally, click the **Save** button in the right-panel to save the changes.

Once the aforesaid pre-requisites are fulfilled, the eG agent will execute the tests and collect the desired performance metrics from the GlassFish server. Using the statistics reported by these tests, administrators can find quick and accurate answers for the following performance queries:

- Is any application deployed on the server experiencing a request overload? If so. which application is this?
- Is any application processing requests slowly? Which application is this and what is causing the processing delay - is it because of poorly responsive servlets? is it because of JSP errors? is it owing to EJB caches running out of beans or EJB-related errors? is it because of one/more inefficient EJB methods? Which servlet/JSP/EJB cache/EJB pool/EJB method is contributing to this slowdown?
- Did any application send out an error response recently?
- Which application is overloaded with sessions?
- Has any application rejected any sessions?
- Are all thread pools utilizing threads optimally? Are any thread pools guilty of ineffective thread usage?
- Do the thread pools have adequate threads to handle the request load?
- Are transaction rollbacks kept at a minimum?
- Are requests to any connection pool waiting a long time for connections? Are too many such requests in waiting? Is it because there are not enough connections in the pool to service the requests?
- Has a potential connection leak been detected in any pool?

The bottom four layers of Figure 1 have already been dealt with in the *Monitoring Unix and Windows Servers* document. The **JVM** layer has already been discussed elaborately in the *Monitoring Java Applications* document. The sections that follow will therefore focus on the top 4 layers of Figure 1 only.

1.1 The GlassFish Server Layer

Using the tests mapped to this layer, administrators can measure the session, request, and transaction load to the GlassFish server and determine how well the server handles the load.

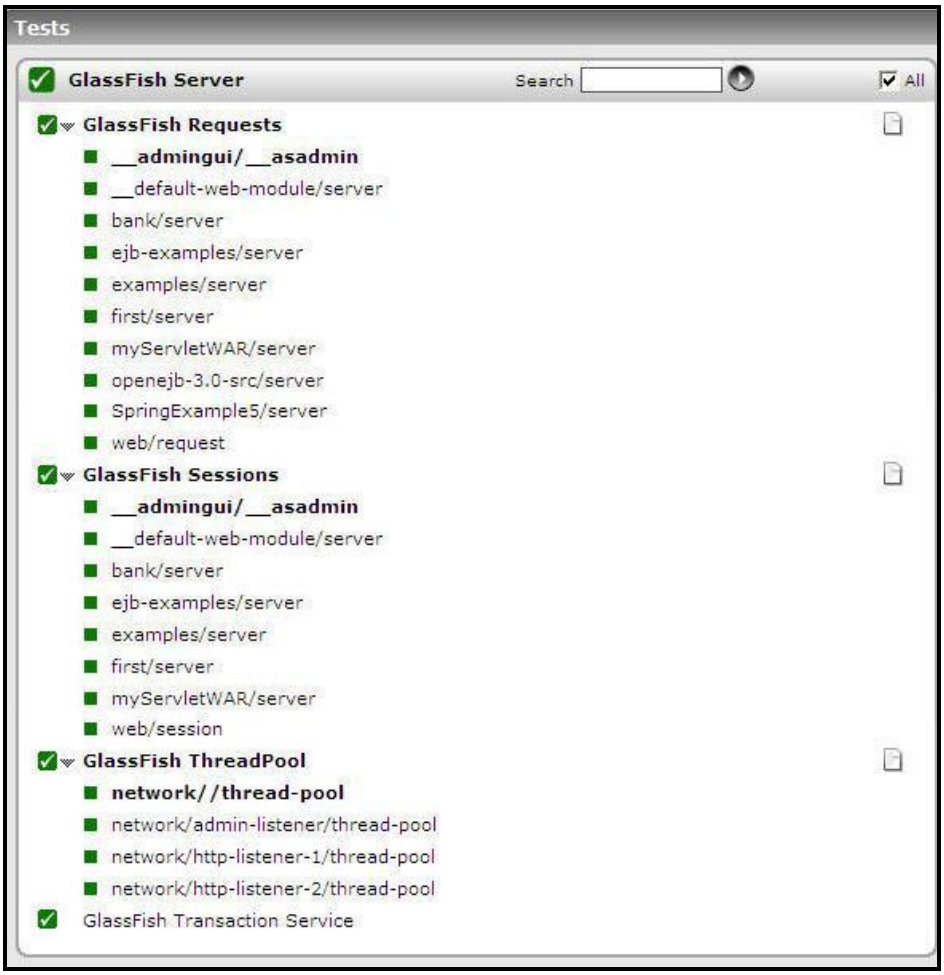

Figure 3: The tests mapped to the GlassFish Server layer

1.1.1 GlassFish Requests Test

The true test for the efficiency of a web application server, such as the GlassFish Enterprise server, is the speed with which it processes requests to its web applications. The **GlassFish Requests** test enables administrators to judge the efficiency of GlassFish by monitoring the requests to each web application deployed on it, and proactively alerting them to current or potential bottlenecks in request processing. This way, overloaded applications and the least responsive applications can be isolated. In the process, the test also promptly captures errors (if any) that the server may have experienced during request processing.

Purpose	Monitors the requests to the GlassFish Enterprise server and proactively alerts them to current or potential bottlenecks in request processing. This way, overloaded applications and the least responsive applications can be isolated. In the process, the test also promptly captures errors (if any) that the server may have experienced during request processing
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.

Outputs of the test	One set of results for each web application deployed on the GlassFish Enterprise server being monitored		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Requests processed: Indicates the number of requests to this application that were processed during the last measurement period.	Number	This measure is a good indicator of the load on each application. Compare the value of this measure across applications to identify which application is experiencing very high load.
	Average processing time: Indicates the time taken by this application to process the requests to it.	Secs	A low value is desired for this measure. A consistent rise in the value of this measure could indicate a processing bottleneck, which in turn may affect application performance. Compare the value of this measure across applications to know which application is the least responsive to user requests.
	Maximum time: Indicates the maximum time taken by this application to process requests.	Secs	
	Errors: Indicates the number of error responses that were sent by this application during the last measurement period.	Number	This count includes the following: d. The number of requests for which responses with response code that is greater than or equal to 400 were sent; e. The number of requests for which a response could not be sent at all. Ideally, the value of this measure should be 0.
	Error: Indicates the percentage of error responses that were sent by this application during the last measurement period	Percent	Ideally, the value of this measure should be 0.

1.1.2 GlassFish Sessions Test

To understand how much load is imposed by user sessions on each of the web applications deployed on the GlassFish server, and to determine the nature (eg., persisted sessions, rejected sessions, etc.) and the current state of these sessions (whether activated, passivated, created, etc.), use the **GlassFish Sessions** test. This way, the most popular web applications on the server can be isolated, and those applications that reject sessions too often can be identified.

Purpose	To understand how much load is imposed by user sessions on each of the web applications deployed on the GlassFish server, and to determine the nature (eg., persisted sessions, rejected sessions, etc.) and the current state of these sessions (whether activated, passivated, created, etc.
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each web application deployed on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Current active sessions: Indicates the number of sessions that are currently active on this application.	Number	This is a good indicator of the current workload of the application. You can compare the value of this measure across applications to know which applications is the most popular in terms of the number of user sessions open on it.
	Activated sessions: Indicates the number of sessions to this application that were activated during the last measurement period.	Number	
	Passivated sessions: Indicates the number of sessions to this application that were passivated during the last measurement period.	Number	
	Expired sessions: Indicates the number of sessions to this application that expired during the last measurement period.	Number	This measure indicates the number of sessions that remained inactive on the application for a duration in excess of the *session-timeout* value configured at the web application level. If the value of this measure is unreasonably high, you may want to increase the *session-timeout* value, so as to minimize the frequency of session expiry. If the *session-timeout* value is set to 0 or less, then sessions to that web application will never expire.
	Persisted sessions: Indicates the number of sessions to this application that persisted during the last measurement period.	Number	GlassFish Server provides high availability session persistence through **failover** of HTTP session data and stateful session bean (SFSB) session data. Failover means that in the event of a server instance or hardware failure, another server instance in a cluster takes over a distributed session.
	Rejected sessions: Indicates the number of sessions to this application that were rejected during the last measurement period.	Number	A low value is desired for this measure.
	Sessions created: Indicates the number of sessions to this application that were created during the last measurement period.	Number	A session manager automatically creates new session objects whenever a new session starts.

1.1.3 GlassFish Thread Pool Test

The Virtual Machine for the Java platform (Java Virtual Machine) or JVM machine) can support many threads of execution simultaneously. To help performance, GlassFish Server maintains one or more thread pools. It is possible to assign specific thread pools to connector modules, to network listeners, or to the Object Request Broker (ORB).

One thread pool can serve multiple connector modules and enterprise beans. Request threads handle user requests for application components. When GlassFish Server receives a request, it assigns the request to a free thread from the thread pool. The thread executes the client's requests and returns results.

Administrators will have to continuously observe the request load on their applications, periodically check how the thread pool services these requests, and accordingly decide whether/not to resize the thread pool commensurate to the current and anticipated load. The **GlassFish Thread Pool** test provides administrators with the load and thread pool usage insights they require to take these crucial sizing decisions.

Purpose	Provides administrators with the load and thread pool usage insights they require to take crucial sizing decisions
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each thread pool configured on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Core threads: Indicates the number of core threads currently in this pool.	Number	
	Current threads: Indicates the current number of threads in this pool.	Number	
	Current busy threads: Indicates the number of threads in this pool that are currently processing requests.	Number	A high value for this measure is indicative of a high load on this applications deployed on the GlassFish application server. This measure is also useful for determining usage trends. For example, it can show the time of day and the day of the week in which you usually reach peak thread count. In addition, the creation of too many threads can result in out of memory errors or thrashing. By watching this metric, you can reduce excessive memory consumption before it's too late.
	Maximum threads: Indicates the maximum number of threads allowed in this thread pool.	Number	This measure reports the value of the **Max Thread Pool Size** parameter of the GlassFish Enterprise server. This parameter specifies the maximum number of simultaneous requests the server can handle. The default value is 5. When the server has reached the limit or request threads, it defers processing new requests until the number of active requests drops below the maximum amount. Increasing this value will reduce HTTP response latency times.
	Total executed tasks: Indicates the total number of tasks executed by the threads in this pool during the last measurement period.	Number	This measure is a good indicator of how busy the pool was during the last measurement period.

	Thread busy: Indicates the percentage of threads in this pool that are currently in use.	Percent	This measure is computed using the following formula: *(Current busy threads / Maximum threads)*100* A high value is indicative of a busy pool. A value close to 100% indicates excessive utilization of threads in a pool. If the value of this measure grows closer to 100% over time, it indicates that the pool is rapidly running out of threads to service the request load. You may then have to fine-tune the thread pool size using the **Max Thread Pool Size** parameter and the **Min Thread Pool Size** parameter. The **Max Thread Pool Size** parameter specifies the maximum number of simultaneous requests the server can handle. The default value is 5. When the server has reached the limit or request threads, it defers processing new requests until the number of active requests drops below the maximum amount. Increasing this value will reduce HTTP response latency times. In practice, clients frequently connect to the server and then do not complete their requests. In these cases, the server waits a length of time specified by the **Timeout** parameter. Also, some sites do heavyweight transactions that take minutes to complete. Both of these factors add to the maximum simultaneous requests that are required. If your site is processing many requests that take many seconds, you might need to increase the number of maximum simultaneous requests. Adjust the thread count value based on your load and the length of time for an average request. In general, increase this number if you have idle CPU time and requests that are pending; decrease it if the CPU becomes overloaded.

			Suitable Request **Max Thread Pool Size** values range from 100 to 500, depending on the load. If your system has extra CPU cycles, keep incrementally increasing thread count and monitor performance after each incremental increase. When performance saturates (stops improving), then stop increasing thread count. The **Min Thread Pool Size** property specifies the minimum number of threads the server initiates upon startup. The default value is 2. **Min Thread Pool Size** represents a hard limit for the maximum number of active threads that can run simultaneously, which can become a bottleneck for performance. Specifying the same value for minimum and maximum threads allows GlassFish Server to use a slightly more optimized thread pool. This configuration should be considered unless the load on the server varies quite significantly.
	Pool utilization: Indicates the percentage of threads in the pool that are active.	Percent	This measure is computed as a ratio of the value of the *Current threads* measure and the *Maximum threads* measure.

1.1.4 GlassFish Transaction Service Test

A **transaction** is a series of discreet actions in an application that must all complete successfully. By enclosing one or more actions in an indivisible unit of work, a transaction ensures data integrity and consistency.

If all these actions complete successfully, the transaction is **committed**. If any action fails, all changes from the preceding steps are rolled back. This type of event is called a **rollback**. A normal transaction ends in either a committed state or a rolled back state.

The transaction manager makes it possible to commit and roll back distributed transactions.

By monitoring the transaction manager, you can determine the transaction load on the GlassFish Enterprise server, detect transaction rollbacks, and receive regular updates on transaction status. The **Transaction Service** test does all the above.

Purpose	Helps determine the transaction load on the GlassFish Enterprise server, detect transaction rollbacks, and receive regular updates on transaction status
Target of the	A GlassFish server

test	
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for the GlassFish Enterprise server being monitored

<table>
<tr><th>Measurements made by the test</th><th>Measurement</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td rowspan="3"></td><td>Active transactions:
Indicates the number of transactions that were active on the server during the last measurement period.</td><td>Number</td><td>This is a good indicator of the transaction load on the server.</td></tr>
<tr><td>Rollback transactions:
Indicates the number of transactions that were rolled back during the last measurement period.</td><td>Number</td><td>Rollbacks are resource-intensive operations. Ideally therefore, the value of this measure should be low.</td></tr>
<tr><td>Committed transactions:
Indicates the number of transactions that were committed during the last measurement period.</td><td>Number</td><td></td></tr>
<tr><td></td><td>Transaction state:
Indicates whether the transactions are currently in a frozen state or not.</td><td></td><td>If a transaction that you intend to rollback is active, then you will have to freeze that transaction before attempting the rollback. Freezing a transaction avoids the possibility of that transaction completing before the rollback request is issued.

This measure reports the value True if the transaction is currently frozen and reports the value False if the transaction is currently active.

The numeric values that correspond to the measure values described above are listed in the table below:
<table><tr><th>Measure Value</th><th>Numeric Value</th></tr><tr><td>True</td><td>1</td></tr><tr><td>False</td><td>0</td></tr></table>
Note:

This measure reports the Measure Value listed in the table above to indicate the state of transactions. However, in the graph of this measure, the transaction state is indicated using only the Numeric Values listed in the above table.</td></tr>
</table>

1.2 The GlassFish Connection Pools Layer

A connection pool is a cache of database connections maintained so that the connections can be reused when future requests to the database are required. With the help of the **GlassFish ConnectionPool** test mapped to it, this layer reveals how each connection pool configured on the server utilizes the connections in it.

Figure 4: The tests mapped to the GlassFish Connection Pools layer

1.2.1 GlassFish ConnectionPool Test

A connection pool is a cache of database connections maintained so that the connections can be reused when future requests to the database are required. Connection pools are used to enhance the performance of executing commands on a database. Opening and maintaining a database connection for each user, especially requests made to a dynamic database-driven website application, is costly and wastes resources. In connection pooling, after a connection is created, it is placed in the pool and it is used over again so that a new connection does not have to be established. Connection pooling also cuts down on the amount of time a user must wait to establish a connection to the database. As opposed to this, the lack of adequate connections in the pool can prolong the time taken to connect to a database. To analyze the impact of connection pools on the database connection time, use the **GlassFish ConnectionPool** test. This test auto-discovers the connection pools configured on the GlassFish server, tracks the usage of connections in each pool, and reports whether/not sufficient connections are available in the pools to handle the connection requests to the GlassFish server. This way, the test points you to connection pools that are starved for connections and helps you assess how this impacts connection requests to the GlassFish server.

Purpose	Auto-discovers the connection pools configured on the GlassFish server, tracks the usage of connections in each pool, and reports whether/not sufficient connections are available in the pools to handle the connection requests to the GlassFish server
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each connection pool on the Java application being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Total beans in cache: Indicates the number of EJBs in the EJB cache.	Number	
	Used connections: Indicates currently used connections in this pool.	Number	A high value indicates a large number of connections to the database.
	Free connections: Indicates the total number of free connections in this pool.	Number	
	Connection waiting time: Indicates waiting time of the last request that was serviced in the pool.	Secs	
	Average connection waiting time: Indicates average waiting time per successful request to this pool.	Secs	The effectiveness of a connection pool can be measured by observing how much it reduces the connection time. A low value for this measure is hence a clear indicator that the connection pool is effective in reducing the wait time of connection requests. A high value on the other hand, could indicate that the pool is running out of connections often, causing connection requests to wait for long periods of time to be processed.
	Connections timedout: Indicates the total number of connections in this pool that timed out between the start time and the last sample time.	Number	
	Potential connection leak: Indicates currently available potential connection leaks.	Number	Ideally, the value of this measure should be 0.
	Waiting queue length:: Indicates the number of connection requests to this pool that are currently in waiting queue.	Number	A consistent increase in the value of this measure could indicate that there are not enough connections in the pool to service the connection requests received by the GlassFish server.
	Failed connections: Indicates number of connections in this connection pool that failed validation from the start time until the last sample time.	Number	Ideally, the value of this measure should be 0.

	Released connections: Indicates the current number of logical connections released to this pool.	Number	
	Destroyed connections: Indicates the number of physical connections in this pool that were destroyed since the last reset.	Number	

1.3 The GlassFish Servlets Layer

The tests mapped to this layer monitors the ability of the GlassFish server to process requests for JSPs and servlets. In the event of a slowdown of one/more web applications deployed on the server, you can use these tests to figure out where the slowdown occurred - while processing requests for JSPs? or while processing servlets? - and also isolate the JSP/servlet that is delaying the processing.

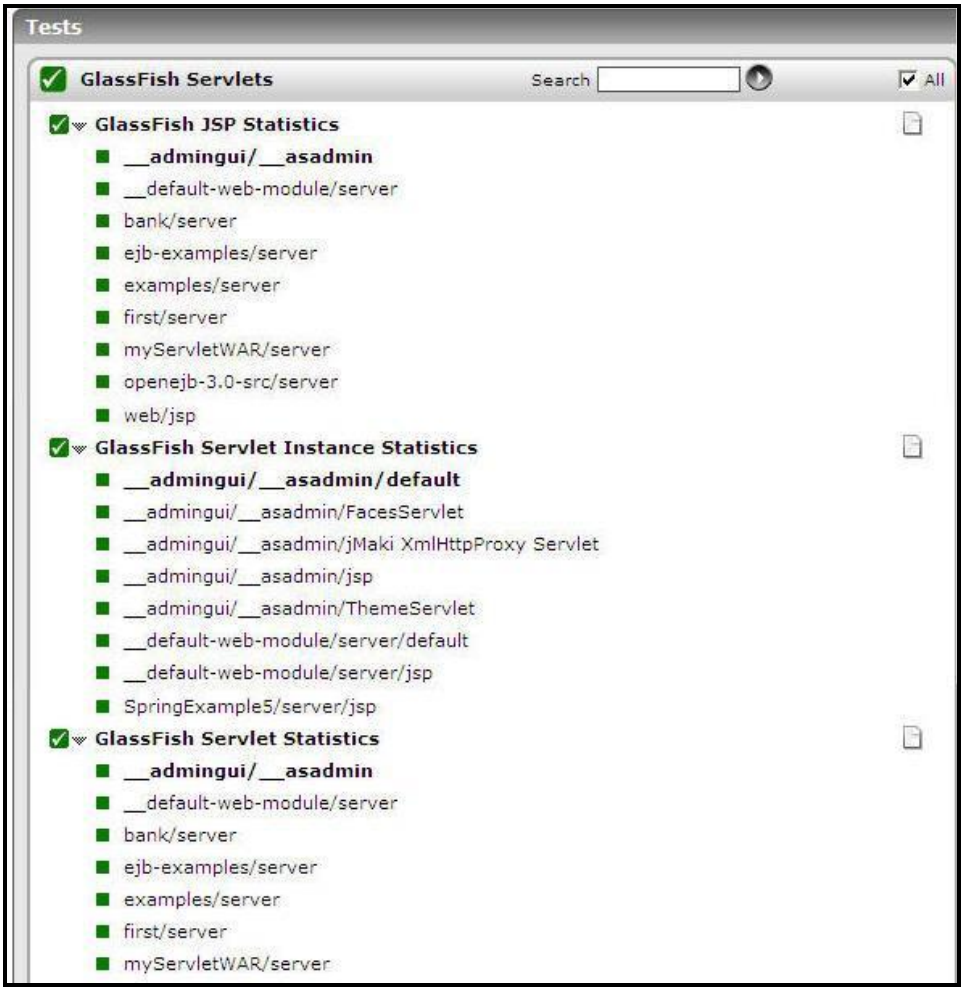

Figure 5: The tests mapped to the GlassFish Servlets layer

1.3.1 GlassFish JSP Statistics Test

To know how well the GlassFish server processes requests for JSP pages and to promptly capture JSP errors, use the **GlassFish JSP Statistics** test.

Purpose	Reports how well the GlassFish server processes requests for JSP pages and promptly captures JSP errors
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each JSP processed by the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Active jsps: Indicates the number of JSPs that are currently active on this application.	Number	
	Jsp loaded: Indicates the number of JSPs that were loaded by this application during the last measurement period.	Number	This counter gets incremented the first time a JSP is accessed, and its corresponding servlet is loaded and initialized.
	Jsp reloaded: Indicates the number of JSPs that were reloaded by this application during the last measurement period.	Number	This counter gets incremented whenever a JSP whose source code has changed since it was first deployed is accessed again and recompiled, and its corresponding servlet is reloaded and reinitialized.
	Jsp errors: Indicates the number of JSP errors experienced by this application during the last measurement period.	Number	Ideally, the value of this measure should be 0.

1.3.2 GlassFish Servlet Instance Statistics Test

Whenever users to a web application deployed on the GlassFish Enterprise server complain of a slowdown, administrators need to instantly determine what is causing the slowdown. The **GlassFish Servlet Statistics** test will enable administrators to figure out if the servlets used by the problem application is contributing to the slowdown or not. If so, then, administrators can use the **GlassFish Servlet Instance Statistics** test to isolate that servlet, which is the processing bottleneck. This test auto-discovers the servlets used by each web application that is installed on the GlassFish server, and reports the processing time of and errors experienced by each servlet, so that problematic servlets can be quickly and accurately isolated, and the root-cause of the problems investigated.

By default, this test monitors only those servlets that are part of a servlet group. To configure servlet groups, click on the 'Click here' hyperlink above the parameters of this test.

Purpose	Auto-discovers the servlets used by each web application that is installed on the GlassFish server, and reports the processing time of and errors experienced by each servlet, so that problematic servlets can be quickly and accurately isolated, and the root-cause of the problems investigated
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each servlet group/servlet used by each web application deployed on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Requests processed: Indicates the number of requests to this application that were processed by this servlet group/servlet during the last measurement period.	Number	This measure is a good indicator of the load on each servlet. Compare the value of this measure across the servlets of an application to identify which servlet is experiencing very high load.
	Avg processing time: Indicates the time taken by this servlet group/servlet to process requests.	Secs	A low value is desired for this measure. A consistent rise in the value of this measure could indicate a processing bottleneck, which in turn may affect application performance. Compare the value of this measure across servlets (associated with a particular application) to know which servlet processing requests too slowly.
	Maximum time: Indicates the maximum time taken by this servlet group/servlet to process requests.	Secs	
	Servlet errors: Indicates the number of error responses that were sent by this servlet group/servlet during the last measurement period.	Number	This count includes the following: f. The number of requests for which responses with response code that is greater than or equal to 400 were sent; g. The number of requests for which a response could not be sent at all. Ideally, the value of this measure should be 0.
	Error: Indicates the percentage of error responses that were sent by this servlet group/servlet during the last measurement period	Percent	Ideally, the value of this measure should be 0.

1.3.3 GlassFish Servlet Statistics Test

For each web application that is deployed on a GlassFish server, this test monitors the processing ability of the servlets used by that application, and reports whether the servlets are responsible for any application slowdowns noticed by users.

Purpose	For each web application that is deployed on a GlassFish server, this test monitors the processing ability of the servlets used by that application, and reports whether the servlets are responsible for any application slowdowns noticed by users.

Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each web application deployed on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Active servlets: Indicates the number of servlets of this application that are actively servicing requests currently.	Number	
	Servlet processing time: Indicates the time taken by the servlets of this application to process requests during the last measurement period.	Secs	A low value is desired for this measure. A consistent rise in the value of this measure could indicate that the servlets are taking too long to process requests.
	Servlets loaded: Indicates the number of servlets loaded into this application during the last measurement period.	Number	

1.4 The GlassFish EJBs Layer

The tests mapped to this layer enable monitoring of the following:

- Usage of EJB caches;
- EJB methods and their execution times;
- Usage of beans in the EJB pools;
- The Stateless and Stateful session bean containers and the nature of methods invoked on them.

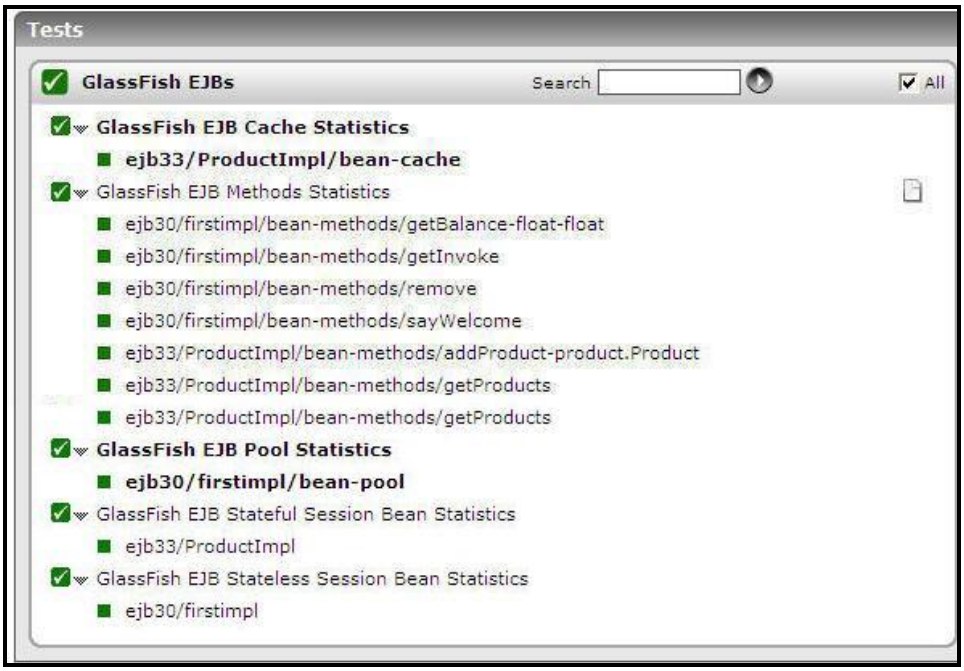

Figure 6: The tests mapped to the GlassFish EJBs layer

1.4.1 GlassFish EJB Cache Statistics

To improve the performance and the scalability of the EJB container, the GlassFish server caches EJBs. A bean in the cache represents the ready state in the EJB lifecycle. This means that the bean has an identity (for example, a primary key or session ID) associated with it.

Beans moving out of the cache have to be passivated or destroyed according to the EJB lifecycle. Once passivated, a bean has to be activated to come back into the cache.

Any incoming request using these "ready" beans from the cache avoids the overhead of creation, setting identity, and potentially activation. It is hence good to cache as many beans as possible.

Using this test, you can monitor the usage of the EJB cache and accurately identify caches that are grossly under-utilized. You can also diagnose the probable cause for the ineffective cache utilization - is it because adequate beans are not available in the cache? if so, why? - is it because the cache size is too small and needs to be resized to accommodate more beans or to reduce cache overflow? In the process, you can even determine the success/failure of stateful session bean passivation attempts.

Purpose	Monitors the usage of the EJB cache and accurately identify caches that are grossly under-utilized
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each cache configured on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Total beans in cache: Indicates the number of EJBs in this EJB cache.	Number	
	Cache hits: Indicates current number of hits in this EJB Cache.	Number	A high value is desired for this measure. A low value indicates that adequate beans are not available in the cache to service requests to the server. This in turn could increase the processing overheads of the requests to the server.
	Cache misses: Indicates the number of times a current user request failed to find an EJB in this EJB cache instance.	Number	A low value is desired for this measure.
	Cache hits ratio: Indicates the percentage of user requests that were serviced by the beans in this cache.	Percent	A high value is desired for this measure. A low value indicates that adequate beans are not available in the cache to service requests to the server. This in turn could increase the processing overheads of the requests to the server. Under such circumstances, you may want to alter the size of the cache by modifying the *max-cache-size* and *resize-quantity* parameters. The *max-cache-size* governs the maximum number of beans in the cache. Increasing this size can increase the capacity of the cache, and thus enable it to hold additional beans. In case of entity beans, you need to tune this parameter depending on the usage of a particular entity bean. While beans that are used less (for example, an order that is created and never used after the transaction is over) are to be cached less, the beans that are used frequently (for example, an item in the inventory that gets referenced very often), are to be cached more in numbers. The *resize-quantity* governs the number of beans to be created or deleted when the cache is serviced by the server. By increasing this quantity, you can ensure that additional beans are created to serve additional user requests.

	StatefulSessionBean passivations: Indicates the number of StatefulSessionBean passivations that are currently occurring in this cache.	Number	Beans moving out of the cache have to be passivated or destroyed according to the EJB lifecycle. Once passivated, a bean has to be activated to come back into the cache. Typically, bean passivation occurs when the *max-cache-size* setting is exceeded and the cache overflows. A bean is also passivated if it outlives the *cache-idle-timeout-in-seconds* settings.
	Passivations success: Indicates the number of passivations that suceeded currently in this cache.	Number	
	Passivations errors: Indicates the number of errors that occurred when passivating the beans in this cache.	Number	Ideally, the value of this measure should be 0.
	Removed sessions: Indicates the number of expired session beans that were currently removed from this cache.	Number	If a stateful session bean remains idle in the backup store for a duration beyond the configured *removal-timeout-in-seconds*, then it is removed from the backup store and will not be accessible to the client. The default value is 60 minutes.

1.4.2 GlassFish EJB Method Statistics

This test auto-discovers the methods invoked by the GlassFish server, reports the time taken by the server to execute each method, and reveals errors that may have occurred during method execution. When application users complaint of a slowdown, then this test provides the method-level insight that is necessary for troubleshooting the slowdown. With the help of this test, administrators can quickly figure out which method is taking too long to execute and which methods are throwing exceptions often, and will thus be able to zero-in on the root-cause of the application slowdown.

Purpose	Auto-discovers the methods invoked by the GlassFish server, reports the time taken by the server to execute each method, and reveals errors that may have occurred during method execution
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each method invoked by the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Success methods: Indicates the number of invocations of this method that were successful during the last measurement period.	Number	
	Error methods: Indicates the number of invocations of this method that resulted in errors during the last measurement period.	Number	Ideally, the value of this measure should be 0.
	Error: Indicates the percentage of this method's invocations that resulted in errors..	Percent	A very low value is desired for this measure. Too many errors in method execution can adversely impact application performance.
	Execution time: Indicates the time taken for this method to execute during the last measurement period.	Secs	A very low value is desired for this measure. A high value is indicative of problems in method execution, which can cause slowdowns in application performance.
	Statistics: Indicates the number of times this method was invoked in the last measurement period.	Number	

1.4.3 GlassFish EJB Pool Statistics Test

A bean in the pool represents the pooled state in the EJB lifecycle. This means that the bean does not have an identity. The advantage of having beans in the pool is that the time to create a bean can be saved for a request. The container has mechanisms that create pool objects in the background, to save the time of bean creation on the request path.

Stateless session beans and entity beans use the EJB pool. By continuously monitoring the usage of these beans in the pools, you can accurately isolate the pools that are over-utilized - i.e., pools that do not enough free beans to service future requests - and under-utilized - i.e., pools that have too many beans than required. This test enables administrators to achieve the same. Based on the usage reports of this test, administrators can tune the pool size, so that abnormal usage patterns can be avoided.

Purpose	Continuously monitors the usage of beans in the pools, and helps administrators accurately isolate the pools that are over-utilized - i.e., pools that do not enough free beans to service future requests - and those that are under-utilized - i.e., pools that have too many beans than required
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each EJB pool configured on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Total beans in pool: Indicates the number of beans that are currently available in this pool.	Number	
	Total waiting threads: Indicates the number of threads that are currently waiting for a free bean in this pool.	Number	A well-tuned pool is one that has enough free beans to service all user requests, and does not keep any request thread waiting for a free bean. Ideally therefore, this measure should report the value 0 for a pool. A non-zero value indicates that the corresponding pool is running out of beans. This could be because the pool is improperly sized. You may then have to resize the pool. For this, you can use any of the following parameters: ➢ *steady-pool-size*: This governs the initial and minimum number of beans maintained in the pool. Set this property to a number greater than zero for a moderately loaded system. Having a value greater than zero ensures that there is always a pooled instance to process an incoming request. ➢ *max-pool-size*: This governs the maximum number of connections that can be created to satisfy client requests. Set this property to be representative of the anticipated high load of the system. An very large pool wastes memory and can slow down the system. A very small pool is also inefficient due to contention. ➢ *resize-quantity*: This governs the number of beans to be created or deleted in the pool when the requests are being serviced by the server. Be sure to re-calibrate the pool resize quantity when you change the maximum pool size, to maintain an equilibrium. Generally, a larger maximum pool size should have a larger pool resize quantity.

	Loaded JMS messages: Indicates the number of messages that are currently loaded into a JMS session in this pool during the last measurement period.	Number	The message-driven bean container uses the JMS service integrated into the GlassFish Server for message-driven beans that are JMS clients. The container manages a pool of message-driven beans for the concurrent processing of a stream of messages. A message-driven bean is a client to a Connector inbound resource adapter.
	Created beans in pool: Indicates the number of beans created in this pool during the last measurement period.	Number	
	Destroyed beans in pool: Indicates the number of beans destroyed in this pool during the last measurement period.	Number	If one/more beans in a pool violate the *pool-idle-timeout-in-seconds* setting of that pool - i.e., if the beans were idle for a duration longer than the *pool-idle-timeout-in-seconds* configuration - then such beans are destroyed, provided they are stateless session beans or message driver beans.

1.4.4 GlassFish EJB Stateful Session Bean Statistics Test

The **stateful container** manages the stateful session beans, which, by definition, carry the client-specific state. There is a one-to-one relationship between the client and the stateful session beans. At creation, each stateful session bean (SFSB) is given a unique session ID that is used to access the session bean so that an instance of a stateful session bean is accessed by a single client only. Stateful session beans are managed using cache.

This test monitors the stateful container and reports the number and nature of methods that were invoked on the container. In the process, the test reports the status of the stateful session beans - whether they have just been created, are in the 'ready' state, are in the 'passivate' state, or have been removed from the container.

Purpose	Monitors the stateful container and reports the number and nature of methods that were invoked on the container. In the process, the test reports the status of the stateful session beans - whether they have just been created, are in the 'ready' state, are in the 'passivate' state, or have been removed from the container
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each stateful session bean container configured on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Create method calls: Indicates the number of times the create method was invoked on this container.	Number	
	Ready method calls: Indicates the number of times in the last measurement period the 'Ready' method was invoked on this container.	Number	A bean in the cache represents the ready state in the EJB lifecycle. This means that the bean has an identity (for example, a primary key or session ID) associated with it.
	Beans in passivate state: Indicates the number of beans in this container that were in the passivate state during the last measurement period.	Number	Beans moving out of the cache have to be passivated or destroyed according to the EJB lifecycle. Once passivated, a bean has to be activated to come back into the cache. Typically, bean passivation occurs when the *max-cache-size* setting is exceeded and the cache overflows. A bean is also passivated if it outlives the *cache-idle-timeout-in-seconds* settings.
	Remove method calls: Indicates the number of number of times the 'remote' method was invoked on the beans in this container during the last measurement period.	Number	If a stateful session bean remains idle in the backup store for a duration beyond the configured *removal-timeout-in-seconds*, then it is removed from the backup store and will not be accessible to the client. The default value is 60 minutes.

1.4.5 GlassFish EJB Stateless Session Bean Statistics Test

The **stateless container** manages stateless session beans, which, by definition, do not carry client-specific states. All session beans (of a particular type) are considered equal. A stateless session bean container uses a bean pool to service requests.

This test monitors the stateless containers and reports the number and nature of methods that were invoked on each container. In the process, the test reports the status of the stateless session beans - whether they have just been created, or are in the 'ready' state, in the 'passivate' state, or have been removed from the container.

Purpose	Monitors the stateless containers and reports the number and nature of methods that were invoked on each container. In the process, the test reports the status of the stateless session beans - whether they have just been created, or are in the 'ready' state, in the 'passivate' state, or have been removed from the container
Target of the test	A GlassFish server
Agent deploying the test	An internal/remote agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST -** The host for which the test is to be configured 3. **PORT -** The port number at which the specified **HOST** listens 4. **JMX REMOTE PORT** – To collect metrics from a GlassFish Enterprise server, the eG agent monitoring that server should be configured to use **JMX** to connect to the JRE used by the server and pull out the metrics of interest. By default, **JMX** support is enabled for the JRE used by the GlassFish Enterprise Server. The JMX connector listens on port 8686 by default. Therefore, type 8686 as the **JMX REMOTE PORT**. If JMX listens on a different port in your environment, then specify the same here. To know the port at which JMX listens, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The port number specified against the *port* parameter in the code block above refers to the **JMX REMOTE PORT**. 5. **JMX USER, JMX PASSWORD, AND CONFIRM PASSWORD** – By default, the JMX connector on the GlassFish Enterprise Server requires **authentication**. To enable the eG agent to use JMX therefore, you need to configure the agent with the credentials of a user who is authorized to use JMX. Typically, the GlassFish *administrator* has the right to use JMX. You can hence configure the **JMX USER** and **JMX PASSWORD** parameters with the credentials of the *administrator*. However, if you prefer not to expose the credentials of an *administrator* owing to security considerations, you can use the credentials of any other user with access rights to JMX. To know the name of such a user, open the **domain.xml** file in the **<GLASSFISH_INSTALL_DIR>\Sun\AppServer\domains\domain1\config** directory and look for the code block shown below: `<jmx-connector accept-all="false" address="0.0.0.0" auth-realm-name="admin-realm" enabled="true" name="system" port="8686" protocol="rmi_jrmp" security-enabled="false">` The user name specified against the *auth-realm-name* parameter in the code block above can be configured as the **JMX USER**, and the password of that user can be specified against **JMX PASSWORD**. Confirm the **JMX PASSWORD** you specify by retyping that password in the **CONFIRM PASSWORD** text box. 6. **JNDINAME** – 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. 7. **JMX PROVIDER** – This test uses a JMX Provider to access the MBean attributes of the GlassFish Enterprise server and collect metrics. Specify the package name of this JMX Provider here. By default, this is set to *com.sun.jmx.remote.protocol*. 8. **TIMEOUT** – Specify the duration (in seconds) for which this test should wait for a response from the GlassFish Enterprise server. If there is no response from the server beyond the configured duration, the test will timeout. By default, this is set to *240* seconds.
Outputs of the test	One set of results for each stateless session bean container configured on the GlassFish Enterprise server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Create method calls: Indicates the number of times the create method was invoked on this container.	Number	
	Ready method calls: Indicates the number of times in the last measurement period the 'Ready' method was invoked on this container.	Number	
	Remove method calls: Indicates the number of number of times the 'remote' method was invoked on the beans in this container during the last measurement period.	Number	

Chapter 2

Conclusion

This document has clearly explained how eG Enterprise monitors the **GlassFish Enterprise Server**. For more information on eG Enterprise, please visit our web site at www.eginnovations.com or write to us at sales@eginnovations.com.