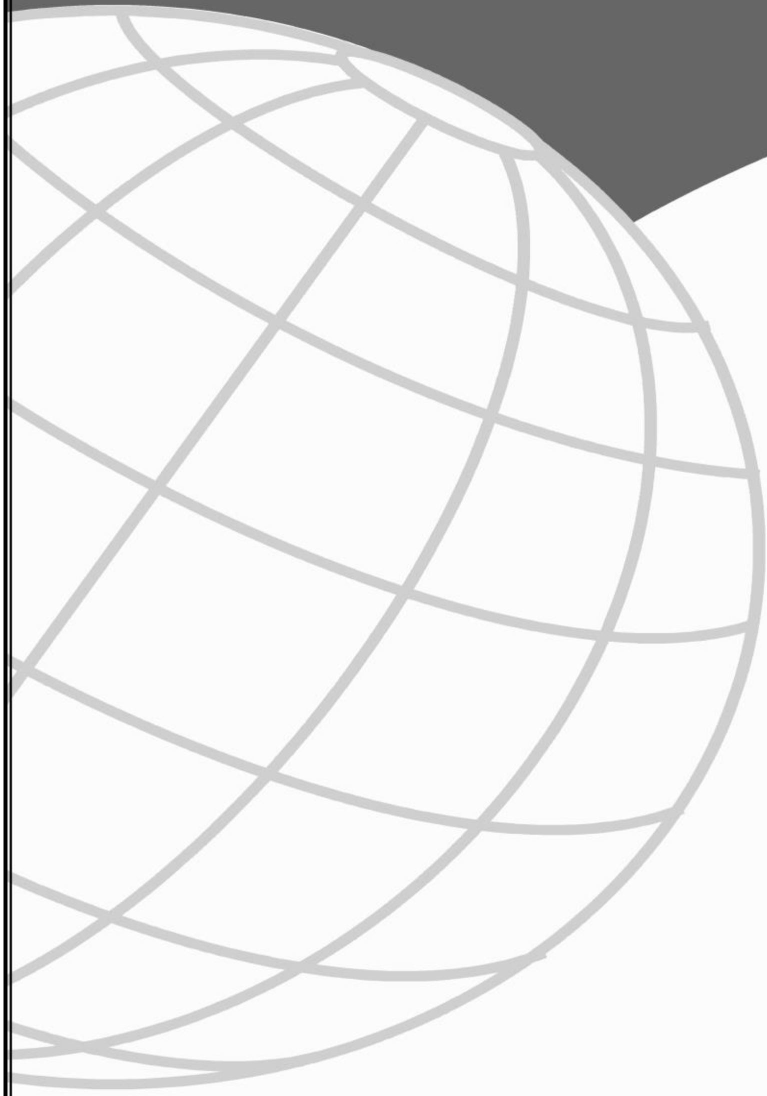


Enabling Service Excellence

Monitoring the Tuxedo PIA

eG Enterprise v6.0

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 2003, 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

Table of Contents

Table of Figures

1

Monitoring the Tuxedo PIA

PeopleSoft Internet Architecture (PIA), leverages a number of internet technologies and concepts to deliver simple, ubiquitous access to PeopleSoft applications and enable the open flow of information between systems. Using PeopleSoft Internet Architecture as the foundation, customers will be able to provide a wide range of end users with access to PeopleSoft applications over the web, as well as more easily integrate their PeopleSoft applications with existing internal systems and external trading partner systems.

The PeopleSoft Internet Architecture is comprised of these main server types:

- RDBMS
- Tuxedo Application Server(s)
- Web server(s)

The servers facilitate connections and process requests from:

- PeopleTools Development Environment: A Windows workstation running a development tool, such as PeopleSoft Application Designer.
- Browser: A supported browser type and version displaying a PeopleSoft application or administrative interface.
- Remote system: A PeopleSoft or third-party system integrated through PeopleSoft Integration Broker's service oriented architecture (SOA).

The following diagram illustrates the relationship between the elements of the PeopleSoft Internet Architecture.

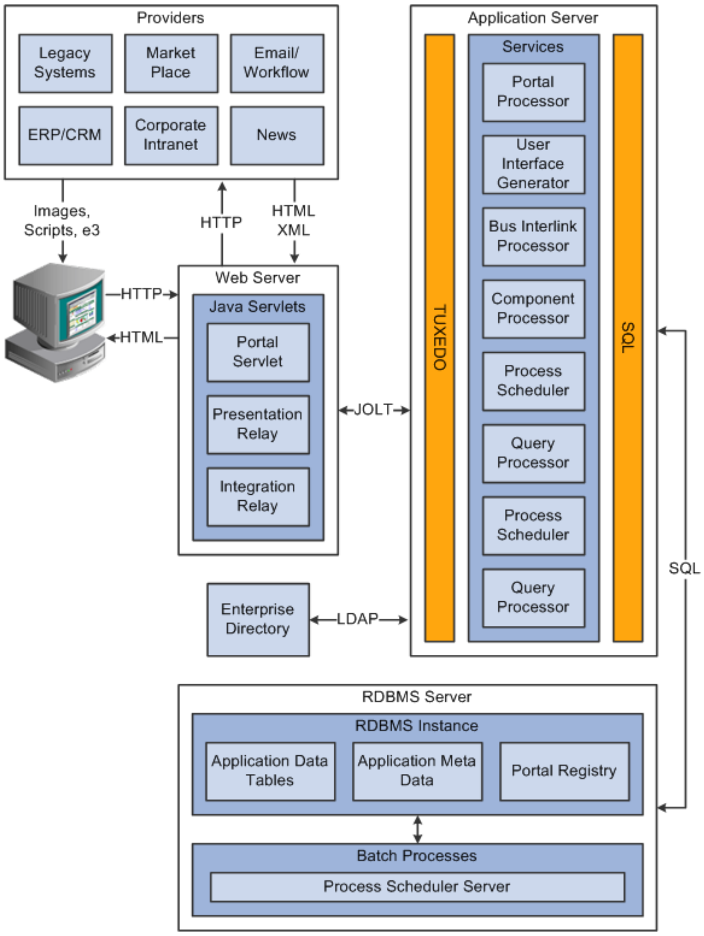


Figure 1.1: How the PeopleSoft Internet Architecture operates

As is evident from the image, application requests are received by the web server using three PeopleSoft servlets – namely, the Presentation Relay Servlet, the Integration Relay Servlet, and the Portal Servlet. When a PeopleSoft application sends a request, it sends a service name and a set of parameters. Any request so received is then forwarded by the web server to the Tuxedo application server. The application server then queues the transaction request to a specific server process that is designed to handle certain services.

An application server consists of numerous PeopleSoft server processes, grouped in domains. When you boot an application server domain, it starts the set of server processes associated with that domain. Each server process establishes a persistent connection to a PeopleSoft database, and this connection acts as a generic SQL pipeline that the server process uses to send and receive data. Listeners, handlers, and queues in a domain receive requests, route requests, store requests, monitor requests, and return request responses.

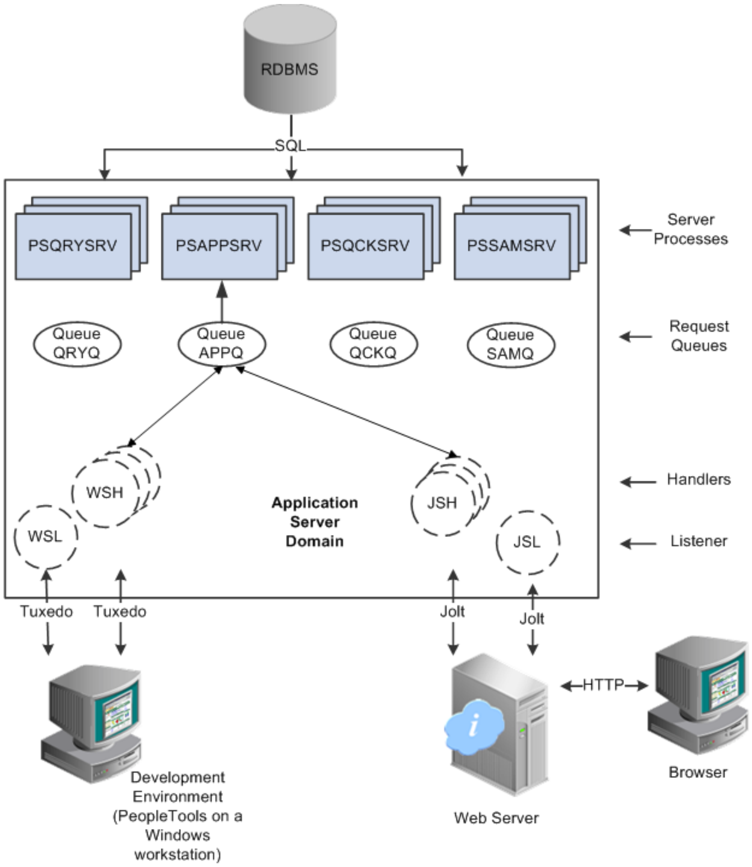


Figure 1.2: How an application domain processes requests

The proper functioning of all these crucial elements – be it the server processes, services, listeners, handlers, and queues in a domain – is essential to ensure that the domain effectively handles application requests, and that users have an above-par experience with PeopleSoft applications at all times.

eG Enterprise provides a specialized *Tuxedo PIA* model (see Figure 1.2) for monitoring the Tuxedo Application Server that is at the 'heart' of the PeopleSoft Internet Architecture. Using this model, processing bottlenecks within an application domain can be rapidly detected and the reason for these bottlenecks, pinpointed.

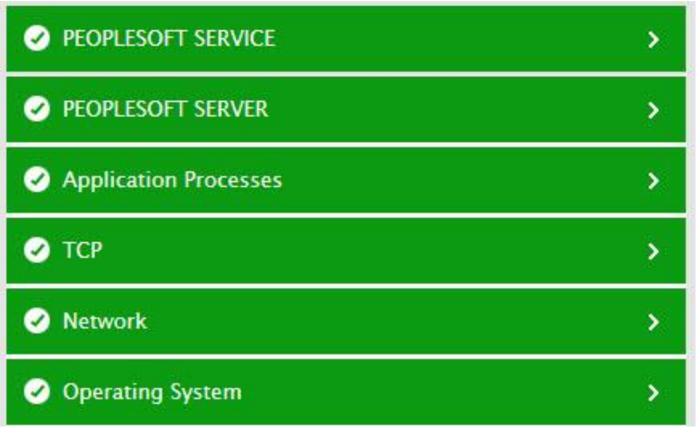


Figure 2.1: The layer model of a Tuxedo PIA

Each layer of Figure 2.1 is mapped to tests that run native PeopleSoft Server Administration (PSADMIN) commands on a single application domain in the target application server and pull out metrics. To run these commands, each of these tests should be configured with the following:

- The full path to the directory in which the **psadmin.exe** resides;
- The name of the application domain to be monitored;

Note

- The *Tuxedo PIA* can only be monitored in an agent-based manner.
- For every application domain that you want monitored, you will have to manage a separate *Tuxedo PIA* component in eG.

Once the tests run and collect the desired metrics, administrators can use these domain-specific metrics to find quick and accurate answers to the following persistent performance queries:

- What is the transaction load on the monitored domain?
- Have any transactions to the domain aborted? If so, which ones are these?
- Is any transaction to the domain about to abort? Which is this?
- What is the current workload of the domain? Which client is imposing the maximum load on the domain?
- Has the domain been configured with adequate server process instances, services, and interfaces to handle the load?
- Are all instances of a server process busy servicing requests presently? Are there idle server process instances in the domain?
- Which is the busiest service in the domain? How quickly is this service processing the requests it receives?
- Is any service unavailable?
- Have too many pending requests been enqueued in any request queue in the domain? Which queue is this and which server process is using this queue? How are the requests in this queue impacting the overall workload of the domain? Should more instances of this server process be configured to reduce the queue length, and consequently, the workload?
- Is the Tuxedo application server able to connect to the database? Are there sufficient number of free connections in the database connection pool to facilitate the connection?
- What is the current workload of the conversational server process (PSSAMSRV) in the monitored domain? Which are the conversations that are contributing to this workload? Has the PSSAMSRV server process been configured right to handle this load?
- Have any errors/warnings been captured by the **APPSRV** log files and **TUXLOG** files?

The sections that follow discuss the top 2 layers of Figure 2.1, as the rest of the layers have already been discussed in the *Monitoring Unix and Windows Servers* document.

1.1 The Application Processes Layer

Besides monitoring the PeopleSoft-related processes running on the application server host, the tests mapped to this layer also check the Application Server log and Tuxedo log files for error events.

1.1.1 Tuxedo PIA Application Logs Test

The **APPSRV.LOG** file contains PeopleTools specific logging information. Certificate authentication logs, including information about mismatched distinguished names and certificates that are not in the database, is contained within this log file. In addition, the log file also shows the current service count for an AppServer process, and when a specific process was recycled/spawned. By periodically scanning these log files for specific patterns of log entries, administrators can quickly capture certificate errors, know when a process was spawned, and determine which services are offered by which server process. This is exactly what the **Tuxedo PIA Application Logs** test does. This test periodically searches the **APPSRV.LOG** file for specific patterns of error/general information messages and reports the count and nature of such messages, so as to enable administrators to troubleshoot better and to easily obtain configuration details.

Purpose	Periodically searches the **APPSRV.LOG** file for specific patterns of error/general information messages and reports the count and nature of such messages, so as to enable administrators to troubleshoot better and to easily obtain configuration details
Target of the test	A Tuxedo application server
Agent deploying the test	An internal 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 at which the server listens

4. **ALERTFILE** - This test monitors the **APPSRV.LOG** files for errors and configuration information. Typically, these log files will be available in the <**PSHome>\appserv\<domain>\LOGS** directory and will be named in the following format: *APPSRV_<mmdd>.log*, where *<mmdd>* refers to the month and date of creation of the log file. In the **LOGS** directory therefore, you will find multiple *APPSRV.log* files, one for every day of the month. To enable the test to monitor a specific **APPSRV.LOG** file, you need to provide the full path to that log file against **ALERTFILE**. For instance, if **<PSHome>** (on Windows) is *C:\ps*, the name of the PeopleSoft domain is *psdomain*, and the log file you want to monitor was created on the 16th of April, your **ALERTFILE** specification will be: *C:|ps|appserv|psdomain|LOGS|APPSRV_0416.log*. Multiple log files can also be monitored as a comma-separated list. For instance, your **ALERTFILE** specification can be: *C:|ps|appserv|psdomain|LOGS|APPSRV_0416.log,C:|ps|appserv|psdomain|LOGS|APPSRV_0420.log*. Specific log file name patterns can also be specified. For example, to monitor all the log files created in the months of March and April (i.e., in the 3rd and 4th months), the parameter specification can be, : *C:|ps|appserv|psdomain|LOGS|APPSRV_03*.log, C:|ps|appserv|psdomain|LOGS|APPSRV_04*.log*. Here, '*' indicates leading/trailing characters (as the case may be).

   Your **ALERTFILE** specification can also be of the following format: *Name@logfilepath_or_pattern*. Here, *Name* represents the display name of the path being configured. For instance, to monitor all **APPSRV** log files that were created in the months of March and April, your specification can be: *Marchlogs@C:|ps|appserv|psdomain|LOGS|APPSRV_03*.log,Aprillogs@C:|ps|appserv|psdomain|LOGS|APPSRV_04*.log*. In this case, the display names 'Marchlogs' and 'Aprillogs' will alone be displayed as descriptors of this test.

   Every time this test is executed, the eG agent verifies the following:

   - Whether any changes have occurred in the size and/or timestamp of the log files that were monitored during the last measurement period;

   - Whether any new log files (that match the **ALERTFILE** specification) have been newly added since the last measurement period;

   If a few lines have been added to a log file that was monitored previously, then the eG agent monitors the additions to that log file, and then proceeds to monitor newer log files (if any). If an older log file has been overwritten, then, the eG agent monitors this log file completely, and then proceeds to monitor the newer log files (if any).

5. **SEARCHPATTERN** - Enter the specific patterns of alerts to be monitored. The pattern should be in the following format: *<PatternName>:<Pattern>*, where *<PatternName>* is the pattern name that will be displayed in the monitor interface and *<Pattern>* is an expression of the form – **expr** or expr or **expr or expr**, etc. A leading '*' signifies any number of leading characters, while a trailing '*' signifies any number of trailing characters.

For example, say you specify *ServerProcess:PSAPPSRV.7948** in the **SEARCHPATTERN** text box. This indicates that "ServerProcess" is the pattern name to be displayed in the monitor interface. "PSAPPSRV.7948*" indicates that the test will monitor only those lines in the alert log which begin with the phrase *PSAPPSRV.7948.* Similarly, if your pattern specification reads: *lpAction:*null*, then it means that the pattern name is "lpAction" and that the test will monitor those lines in the alert log which end with the term "null".

A single pattern may also be of the form e1+e2, where + signifies an OR condition. That is, the <*PatternName*> is matched if either e1 is true or e2 is true.

Multiple search patterns can be specified as a comma-separated list. For example: *ServerProcess:PSAPPSRV.7948*,lpAction:*null*

If the **ALERTFILE** specification is of the format *Name@logfilepatterns*, then the descriptor for this test in the eG monitor interface will be of the format: *Name:PatternName*. On the other hand, if the **ALERTFILE** specification consists only of a comma-separated list of log files/patterns, then the descriptors will be of the format: *LogFile/Pattern:PatternName*.

If you want all the messages in a log file to be monitored, then your specification would be: *<PatternName>:**.

6. **LINES** - Specify two numbers in the format x:y. This means that when a line in the alert file matches a particular pattern, then x lines before the matched line and y lines after the matched line will be reported in the detail diagnosis output (in addition to the matched line). The default value here is 0:0. Multiple entries can be provided as a comma-separated list.

   If you give *1:1* as the value for **LINES**, then this value will be applied to all the patterns specified in the **SEARCHPATTERN** field. If you give *0:0,1:1* as the value for **LINES** and if the corresponding value in the **SEARCHPATTERN** field is like *ServerProcess:PSAPPSRV.7948*,lpAction:*null*, then:

   0:0 will be applied to *ServerProcess:PSAPPSRV.7948** pattern

   1:1 will be applied to *lpAction:*null* pattern

7. **EXCLUDEPATTERN** – Provide a comma-separated list of patterns to be excluded from monitoring in the **EXCLUDEPATTERN** text box. For example: *PSAPPSRV.19826*,PSAPPSRV.7949** . By default, this parameter is set to 'none'.

8. **UNIQUEMATCH** - By default, the **UNIQUEMATCH** parameter is set to **FALSE**, indicating that, by default, the test checks every line in the log file for the existence of each of the configured **SEARCHPATTERNS**. By setting this parameter to **TRUE**, you can instruct the test to ignore a line and move to the next as soon as a match for one of the configured patterns is found in that line. For example, assume that *ServerProcess:*PSAPPSRV.7948*,lpAction:*null** is the **SEARCHPATTERN** that has been configured. If **UNIQUEMATCH** is set to **FALSE**, then the test will read every line in the log file completely to check for the existence of messages embedding the strings '*PSAPPSRV.7948*' and '*null*'. If both the patterns are detected in the same line, then the number of matches will be incremented by 2. On the other hand, if **UNIQUEMATCH** is set to **TRUE**, then the test will read a line only until a match for one of the configured patterns is found and not both. This means that even if the strings '*PSAPPSRV.7948*' and '*null*' follow one another in the same line, the test will consider only the first match and not the next. The match count in this case will therefore be incremented by only 1.

9. **ROTATINGFILE** - This flag governs the display of descriptors for this test in the eG monitoring console.

   If this flag is set to **true**, then the descriptors of this test will be displayed in the following format: *Directory_containing_monitored_file:<SearchPattern>*. For instance, if the **ALERTFILE** parameter is set to *C:\ps\appserv\psdomain\LOGS\APPSRV_0415.log*, and **ROTATINGFILE** is set to **true**, then, your descriptor will be of the following format: *C:\ps\appserv\psdomain\LOGS:<SearchPattern>*. On the other hand, if the **ROTATINGFILE** flag had been set to **false**, then the descriptors will be of the following format: *<FileName>:<SearchPattern>* - i.e., *APPSRV_0415.LOG:<SearchPattern>* in the case of the example above.

10. **CASESENSITIVE** - This flag is set to **No** by default. This indicates that the test functions in a 'case-insensitive' manner by default. This implies that, by default, the test ignores the case of your **ALERTFILE** and **SEARCHPATTERN** specifications. If this flag is set to **Yes** on the other hand, then the test will function in a 'case-sensitive' manner. In this case therefore, for the test to work, even the case of your **ALERTFILE** and **SEARCHPATTERN** specifications should match with the actuals.

11. **ROLLOVERFILE** - By default, this flag is set to **false**. Set this flag to **true** if you want the test to support the 'roll over' capability (if any) of the specified **ALERTFILE**. A roll over typically occurs when the timestamp of a file changes or when the log file size crosses a pre-determined threshold. When a log file rolls over, the errors/warnings that pre-exist in that file will be automatically copied to a new file, and all errors/warnings that are captured subsequently will be logged in the original/old file. In such a scenario, since the **ROLLOVERFILE** flag is set to **false** by default, the test by default scans only the original/old file for new log entries and ignores the new file. On the other hand, if the flag is set to **true**, then the test will scan both old and the rolled-over file for new entries.

    If you want this test to support the 'roll over' capability described above, the following conditions need to be fulfilled:

    - The **ALERTFILE** parameter has to be configured only with the name and/or path of one/more alert files. File patterns should not be specified in the **ALERTFILE** text box.

    - The roll over file name should be of the format: "<**ALERTFILE**>.1", and this file must be in the same directory as the **ALERTFILE**.

12. **OVERWRITTENFILE** - By default, this flag is set to **false**. Set this flag to **true** if log files do not 'roll over' in your environment, but get overwritten instead. In such environments typically, new messages that are captured will be written into the log file that pre-exists and will replace the original contents of that log file; unlike when 'roll over' is enabled, no new log files are created for new entries in this case. If the **OVERWRITTENFILE** flag is set to **true**, then the test will scan the new entries in the log file for matching patterns. However, if the flag is set to **false**, then the test will ignore the new entries.

13. **ENCODEFORMAT** – By default, this is set to *none*, indicating that no encoding format applies by default. However, if the test has to use a specific encoding format for reading from the specified **ALERTFILE** , then you will have to provide a valid encoding format here. Where multiple log files are being monitored, you will have to provide a comma-separated list of encoding formats – one each for every log file monitored - - eg., *UTF-8, UTF-16*. Make sure that your encoding format specification follows the same sequence as your **ALERTFILE** specification. In other words, the first encoding format should apply to the first alert file, and so on.

> **Note:**
>
> If your **ALERTFILE** specification consists of file patterns that include wildcard characters (eg., *C:\ps\appserv\psdomain\LOGS\APPSRV_03*.log, C:\ps\appserv\psdomain\LOGS\APPSRV_04*.log*), then such configurations will only be supported in the ANSI format, and not the UTF format.

14. **USEUTF8** – By default, this flag is set to **No**, implying that the test does not use the **UTF-8** format to read from the log files configured for monitoring. However, if the test has to use only the **UTF-8** format for reading from all the log files configured for monitoring in your **ALERTFILE** specification, then set this flag to **Yes**.
15. **USEUTF16** – By default, this flag is set to **No**, implying that the test does not use the **UTF-16** format to read from the log files configured for monitoring. However, if the test has to use only the **UTF-16** format for reading from all the log files configured for monitoring in your **ALERTFILE** specification, then set this flag to **Yes**.

> **Note:**
>
> If your **ALERTFILE** specification consists of file patterns that include wildcard characters (eg., *C:\ps\appserv\psdomain\LOGS\APPSRV_03*.log, C:\ps\appserv\psdomain\LOGS\APPSRV_04*.log*), then such configurations will only be supported in the ANSI format, and not any of the UTF (UTF-8 or UTF-16) formats.

16. **DD FREQUENCY** - Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD FREQUENCY**.
17. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option.

The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:

- The eG manager license should allow the detailed diagnosis capability
- Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Outputs of the test	One set of results for every **ALERTFILE** and **SEARCHPATTERN** pair configured		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**Number of messages:** Indicates the number of messages matching this **SEARCHPATTERN** found in this **ALERTFILE.**	Number	Use the detailed diagnosis of this measure to view these messages completely.

1.1.2 PeopleSoft Tuxedo Log Monitor Test

The TUXLOG files enable administrators to trace the Tuxedo component for troubleshooting information. Using the **PeopleSoft Tuxedo Log Monitor** Test, administrators can periodically scan these log files for specific patterns of errors/warnings, and can efficiently troubleshoot them.

Purpose	Enables administrators to periodically scan TUXLOG files for specific patterns of errors/warnings, and efficiently troubleshoot them
Target of the test	A Tuxedo application server
Agent deploying the test	An internal 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 at which the server listens
4. **ALERTFILE** - This test monitors the **TUXLOG** files for errors/warnings. Typically, these log files will be available in the <**PSHome>\appserv\<domain>\LOGS** directory and will be named in the following format: *TUXLOG.mmddyy*, where *<mmddyy>* refers to the month, date, and year of creation of the log file. In the **LOGS** directory therefore, you will find multiple **TUXLOG** files, one for every day of the month. To enable the test to monitor a specific **TUXLOG** file, you need to provide the full path to that log file against **ALERTFILE**. For instance, if **<PSHome>** (on Windows) is *C:\ps*, the name of the PeopleSoft domain is *psdomain*, and the log file you want to monitor was created on the 16th of April, 2015, your **ALERTFILE** specification will be: *C:|ps|appserv|psdomain|LOGS|TUXLOG.041615*. Multiple log files can also be monitored as a comma-separated list. For instance, your **ALERTFILE** specification can be: *C:|ps|appserv|psdomain|LOGS|TUXLOG.041615, C:|ps|appserv|psdomain|LOGS|TUXLOG.042015*. Specific log file name patterns can also be specified. For example, to monitor all the log files created in the months of March and April (i.e., in the 3rd and 4th months), the parameter specification can be, : C:|ps|appserv|psdomain|LOGS|TUXLOG.03*15,C:|ps|appserv|psdomain|LOGS|TUXLOG.04 *15. Here, '*' indicates leading/trailing characters (as the case may be).

   Your **ALERTFILE** specification can also be of the following format: *Name@logfilepath_or_pattern*. Here, *Name* represents the display name of the path being configured. For instance, to monitor all **TUXLOG** log files that were created in the months of March and April, your specification can be: Marchlogs@ C:|ps|appserv|psdomain|LOGS|TUXLOG_03*15,Aprillogs@C:|ps|appserv|psdomain|LOGS| TUXLOG_04*15. In this case, the display names 'Marchlogs' and 'Aprillogs' will alone be displayed as descriptors of this test.

   Every time this test is executed, the eG agent verifies the following:

   - Whether any changes have occurred in the size and/or timestamp of the log files that were monitored during the last measurement period;
   - Whether any new log files (that match the **ALERTFILE** specification) have been newly added since the last measurement period;

   If a few lines have been added to a log file that was monitored previously, then the eG agent monitors the additions to that log file, and then proceeds to monitor newer log files (if any). If an older log file has been overwritten, then, the eG agent monitors this log file completely, and then proceeds to monitor the newer log files (if any).

5. **SEARCHPATTERN** - Enter the specific patterns of alerts to be monitored. The pattern should be in the following format: *<PatternName>:<Pattern>*, where *<PatternName>* is the pattern name that will be displayed in the monitor interface and *<Pattern>* is an expression of the form – **expr** or expr or **expr or expr**, etc. A leading '*' signifies any number of leading characters, while a trailing '*' signifies any number of trailing characters.

For example, say you specify Error:ERROR:* in the **SEARCHPATTERN** text box. This indicates that "Error" is the pattern name to be displayed in the monitor interface. "ERROR:*" indicates that the test will monitor only those lines in the alert log which begin with the phrase *ERROR:.* Similarly, if your pattern specification reads: Fail:*failed, then it means that the pattern name is "Fail" and that the test will monitor those lines in the alert log which end with the term "failed".

A single pattern may also be of the form e1+e2, where + signifies an OR condition. That is, the <*PatternName*> is matched if either e1 is true or e2 is true.

Multiple search patterns can be specified as a comma-separated list. For example: Error:ERROR:*, Fail:*failed

If the **ALERTFILE** specification is of the format *Name@logfilepatterns*, then the descriptor for this test in the eG monitor interface will be of the format: *Name:PatternName*. On the other hand, if the **ALERTFILE** specification consists only of a comma-separated list of log files/patterns, then the descriptors will be of the format: *LogFile/Pattern:PatternName*.

If you want all the messages in a log file to be monitored, then your specification would be: <*PatternName*>:*.

6. **LINES** - Specify two numbers in the format x:y. This means that when a line in the alert file matches a particular pattern, then x lines before the matched line and y lines after the matched line will be reported in the detail diagnosis output (in addition to the matched line). The default value here is 0:0. Multiple entries can be provided as a comma-separated list.

   If you give *1:1* as the value for **LINES**, then this value will be applied to all the patterns specified in the **SEARCHPATTERN** field. If you give *0:0,1:1* as the value for **LINES** and if the corresponding value in the **SEARCHPATTERN** field is like Error:ERROR:*, Fail:*failed, then:

   0:0 will be applied to Error:ERROR:* pattern

   1:1 will be applied to Fail:*failed pattern

7. **EXCLUDEPATTERN** – Provide a comma-separated list of patterns to be excluded from monitoring in the **EXCLUDEPATTERN** text box. For example: JOLT_CAT*,LIBTUX_CAT*. By default, this parameter is set to 'none'.

8. **UNIQUEMATCH** - By default, the **UNIQUEMATCH** parameter is set to **FALSE**, indicating that, by default, the test checks every line in the log file for the existence of each of the configured **SEARCHPATTERNS**. By setting this parameter to **TRUE**, you can instruct the test to ignore a line and move to the next as soon as a match for one of the configured patterns is found in that line. For example, assume that Error:*ERROR:*, Fail:*failed* is the **SEARCHPATTERN** that has been configured. If **UNIQUEMATCH** is set to **FALSE**, then the test will read every line in the log file completely to check for the existence of messages embedding the strings '*ERROR:*' and '*failed*'. If both the patterns are detected in the same line, then the number of matches will be incremented by 2. On the other hand, if **UNIQUEMATCH** is set to **TRUE**, then the test will read a line only until a match for one of the configured patterns is found and not both. This means that even if the strings '*ERROR:*' and '*failed*' follow one another in the same line, the test will consider only the first match and not the next. The match count in this case will therefore be incremented by only 1.

9. **ROTATINGFILE** - This flag governs the display of descriptors for this test in the eG monitoring console.

   If this flag is set to **true**, then the descriptors of this test will be displayed in the following format: *Directory_containing_monitored_file:<SearchPattern>*. For instance, if the **ALERTFILE** parameter is set to *C:\ps\appserv\psdomain\LOGS\TUXLOG.042015*, and **ROTATINGFILE** is set to **true**, then, your descriptor will be of the following format: *C:\ps\appserv\psdomain\LOGS:<SearchPattern>*. On the other hand, if the **ROTATINGFILE** flag had been set to **false**, then the descriptors will be of the following format: *<FileName>:<SearchPattern>* - i.e., *TUXLOG.042015:<SearchPattern>* in the case of the example above.

10. **CASESENSITIVE** - This flag is set to **No** by default. This indicates that the test functions in a 'case-insensitive' manner by default. This implies that, by default, the test ignores the case of your **ALERTFILE** and **SEARCHPATTERN** specifications. If this flag is set to **Yes** on the other hand, then the test will function in a 'case-sensitive' manner. In this case therefore, for the test to work, even the case of your **ALERTFILE** and **SEARCHPATTERN** specifications should match with the actuals.

11. **ROLLOVERFILE** - By default, this flag is set to **false**. Set this flag to **true** if you want the test to support the 'roll over' capability (if any) of the specified **ALERTFILE**. A roll over typically occurs when the timestamp of a file changes or when the log file size crosses a pre-determined threshold. When a log file rolls over, the errors/warnings that pre-exist in that file will be automatically copied to a new file, and all errors/warnings that are captured subsequently will be logged in the original/old file. In such a scenario, since the **ROLLOVERFILE** flag is set to **false** by default, the test by default scans only the original/old file for new log entries and ignores the new file. On the other hand, if the flag is set to **true**, then the test will scan both old and the rolled-over file for new entries.

    If you want this test to support the 'roll over' capability described above, the following conditions need to be fulfilled:

    - The **ALERTFILE** parameter has to be configured only with the name and/or path of one/more alert files. File patterns should not be specified in the **ALERTFILE** text box.

    - The roll over file name should be of the format: "<**ALERTFILE**>.1", and this file must be in the same directory as the **ALERTFILE**.

12. **OVERWRITTENFILE** - By default, this flag is set to **false**. Set this flag to **true** if log files do not 'roll over' in your environment, but get overwritten instead. In such environments typically, new messages that are captured will be written into the log file that pre-exists and will replace the original contents of that log file; unlike when 'roll over' is enabled, no new log files are created for new entries in this case. If the **OVERWRITTENFILE** flag is set to **true**, then the test will scan the new entries in the log file for matching patterns. However, if the flag is set to **false**, then the test will ignore the new entries.

13. **ENCODEFORMAT** – By default, this is set to *none*, indicating that no encoding format applies by default. However, if the test has to use a specific encoding format for reading from the specified **ALERTFILE** , then you will have to provide a valid encoding format here. Where multiple log files are being monitored, you will have to provide a comma-separated list of encoding formats – one each for every log file monitored - - eg., *UTF-8, UTF-16*. Make sure that your encoding format specification follows the same sequence as your **ALERTFILE** specification. In other words, the first encoding format should apply to the first alert file, and so on.

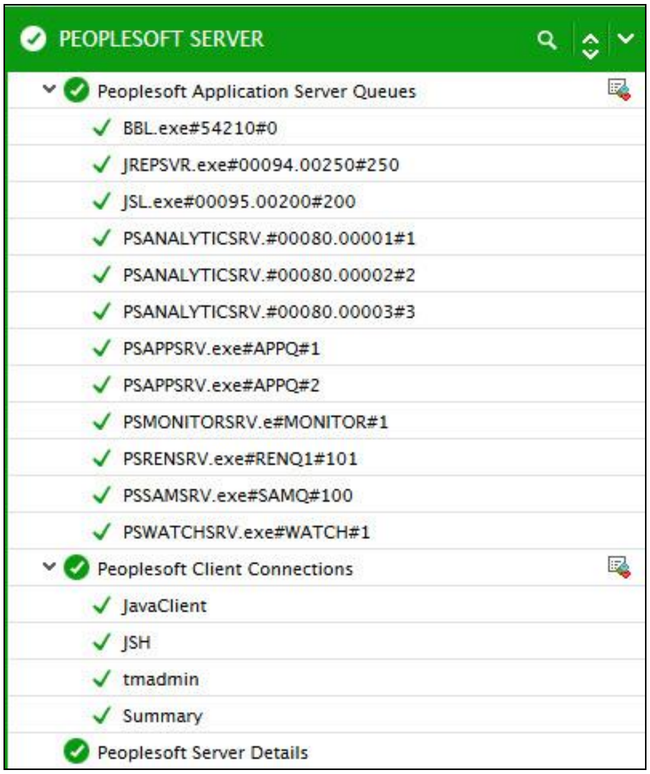
<table>
<tr><td></td><td colspan="3">**Note:**
If your **ALERTFILE** specification consists of file patterns that include wildcard characters (eg., *C:\ps\appserv\psdomain\LOGS\TUXLOG.03*15, C:\ps\appserv\psdomain\LOGS\TUXLOG.04*15*), then such configurations will only be supported in the ANSI format, and not the UTF format.
14. **USEUTF8** – By default, this flag is set to **No**, implying that the test does not use the **UTF-8** format to read from the log files configured for monitoring. However, if the test has to use only the **UTF-8** format for reading from all the log files configured for monitoring in your **ALERTFILE** specification, then set this flag to **Yes**.
15. **USEUTF16** – By default, this flag is set to **No**, implying that the test does not use the **UTF-16** format to read from the log files configured for monitoring. However, if the test has to use only the **UTF-16** format for reading from all the log files configured for monitoring in your **ALERTFILE** specification, then set this flag to **Yes**.
Note:
If your **ALERTFILE** specification consists of file patterns that include wildcard characters (eg., *C:\ps\appserv\psdomain\LOGS\TUXLOG.03*15, C:\ps\appserv\psdomain\LOGS\TUXLOG.04*15*), then such configurations will only be supported in the ANSI format, and not the UTF format.
16. **DD FREQUENCY** - Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD FREQUENCY**.
17. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option.
The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:
• The eG manager license should allow the detailed diagnosis capability
• Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.</td></tr>
<tr><td>**Outputs of the test**</td><td colspan="3">One set of results for every **ALERTFILE** and **SEARCHPATTERN** pair configured</td></tr>
<tr><td>**Measurements made by the**</td><td>**Measurement**</td><td>**Measurement Unit**</td><td>**Interpretation**</td></tr>
</table>

test	**Number of messages:** Indicates the number of messages matching this **SEARCHPATTERN** found in this **ALERTFILE.**	Number	Use the detailed diagnosis of this measure to view these error messages completely.

1.2 The Tuxedo PIA Server Layer

With the help of the tests mapped to this layer, administrators can:

- Know how many request queues the monitored domain consists of, and which queue has the maximum number of enqueued requests;
- Identify the server process that is unable to process requests quickly, and is hence enqueing many requests in its queue;
- Determine the current workload of the domain and the clients who are contributing to this load;
- Ascertain the current status of the server processes in the domain.



PEOPLESOFT SERVER

- ✓ Peoplesoft Application Server Queues
  - ✓ BBL.exe#54210#0
  - ✓ JREPSVR.exe#00094.00250#250
  - ✓ JSL.exe#00095.00200#200
  - ✓ PSANALYTICSRV.#00080.00001#1
  - ✓ PSANALYTICSRV.#00080.00002#2
  - ✓ PSANALYTICSRV.#00080.00003#3
  - ✓ PSAPPSRV.exe#APPQ#1
  - ✓ PSAPPSRV.exe#APPQ#2
  - ✓ PSMONITORSRV.e#MONITOR#1
  - ✓ PSRENSRV.exe#RENQ1#101
  - ✓ PSSAMSRV.exe#SAMQ#100
  - ✓ PSWATCHSRV.exe#WATCH#1
- ✓ Peoplesoft Client Connections
  - ✓ JavaClient
  - ✓ JSH
  - ✓ tmadmin
  - ✓ Summary
- ✓ Peoplesoft Server Details

Figure 1.3: The tests mapped to the Tuxedo PIA Server layer

1.2.1 Tuxedo PIA Server Load Test

Typically, when configuring an application server, administrators can set the "weight" of the load, or amount of requests, that need to be directed to a particular application server. Weight values are integers 1–10, with 1 being low and 10 being a heavy load. Servers that can handle extra work can take heavy loads, while servers that are either less powerful or are being used in other capacities can take lower loads.

When monitoring the queues used by an application server domain, it is not only important to know the number of requests enqueued in each queue at any given point in time, but also how this request count translates into "weighted load" on the domain.

By measuring the weighted load on each queue used by an application server domain, administrators can analyze how load-intensive every queue is, and isolate that queue that imposes the maximum load on the domain. This is where the **Tuxedo PIA Server Load** test helps. This test auto-discovers the queues in an application server domain, and reports the requests handled and the weighted load per queue. In the process, the test pinpoints that queue which contributes the most to the load on the domain, and thus leads administrators to the type of transactions that are being the most load-intensive.

Purpose	Auto-discovers the queues in an application server domain, and reports the requests handled and the weighted load per queue. In the process, the test pinpoints that queue which contributes the most to the load on the domain, and thus leads administrators to the type of transactions that are being the most load-intensive.		
Target of the test	A Tuxedo application server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST** - Host name of the server for which the test is to be configured 3. **PORT** - Enter the port to which the specified **HOST** listens. The default port is 12345. 4. **PSADMIN HOME** – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the **psadmin.exe**. To enable the test to run the **psadmin.exe**, you need to configure the test with the full path to the directory in which the **psadmin.exe** resides. Therefore, in the **PSADMIN HOME** text box, specify the location of the **psadmin.exe**. Typically, this will be the install directory of the Tuxedo PIA. For instance, on Windows, your specification can be *C:|ps|appsrv* and on Unix, your specification can be: *opt/ps/appsrv*. 5. **DOMAIN** - An application server **DOMAIN** is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server **DOMAIN** it needs to monitor.		
Outputs of the test	One set of results for every *Server_process:Queue_ID:Machine_ID* auto—discovered from the monitored application server domain.		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretatione**

test	**Requests handled:** Indicates the number of requests currently handled by this queue.	Number	This is a good indicator of the current queue length. A consistent rise in the value of this measure is a cause for concern, as it indicates that many requests to the corresponding server process are pending processing. This hints at a probable processing bottleneck in the server.
	Server load: Indicates the weighted load imposed by this queue on the server.	Number	This is a weighted measure of the server requests. Some requests could have a different weight than others. By default, the workload is always 50 times the number of requests. Compare the value of this measure across queues to know which queue is imposing the maximum load on the domain.
	Request rate: Indicates the rate at which requests were handled by this queue.	Requests/Sec	
	Load rate: Indicates the load rate of this server queue.	Load/Sec	

1.2.2 Tuxedo PIA Client Connections Test

Monitoring the clients that are communicating with the Tuxedo application server domain and their experience with the domain, will accurately point administrators to clients who are imposing the maximum load on the domain and the clients whose experience with the domain is below-par. This is exactly what the **Tuxedo PIA Client Connections** test does. This test automatically discovers the clients who are currently connected to the server domain, reports the number of active sessions for each client, and also reveals the number of transactions of every client that have been committed and aborted. In the process, the test pinpoints those clients who have launched too many sessions on the domain; in the event that the server domain chokes, such clients could be considered guilty of overloading the domain. Additionally, the test reveals which client's transactions were aborted often, rendering such clients unable to complete their transactions; this in turn may disrupt the business flow and adversely impact experience with the server domain.

Purpose	Automatically discovers the clients who are currently connected to the server domain, reports the number of active sessions for each client, and also reveals the number of transactions of every client that have been committed and aborted. In the process, the test pinpoints those clients who have launched too many sessions on the domain; in the event that the server domain chokes, such clients could be considered guilty of overloading the domain. Additionally, the test reveals which client's transactions were aborted often, rendering such clients unable to complete their transactions; this in turn may disrupt the business flow and adversely impact experience with the server domain.
Target of the test	A Tuxedo application server
Agent	An internal agent

deploying the test			
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST** - Host name of the server for which the test is to be configured 3. **PORT** - Enter the port to which the specified **HOST** listens. The default port is 12345. 4. **PSADMIN HOME** – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the **psadmin.exe**. To enable the test to run the **psadmin.exe**, you need to configure the test with the full path to the directory in which the **psadmin.exe** resides. Therefore, in the **PSADMIN HOME** text box, specify the location of the **psadmin.exe**. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be *C:|ps|appsrv* and on Unix, your specification can be: *opt/ps/appsrv*. 5. **DOMAIN** - An application server **DOMAIN** is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server **DOMAIN** it needs to monitor. 6. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the On option. To disable the capability, click on the Off option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.		
Outputs of the test	One set of results for every client currently connected to the server domain being monitored		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**Number of clients:** Indicates the number of sessions currently active for this client.	Number	This is a good indicator of the workload currently imposed by a client on the application server domain. In the event of an overload, you can compare the value of this measure across clients to know which client is contributing to the overload by launching many sessions on the domain. You can also use the detailed diagnosis of this measure to know which users are communicating with the application server domain via the client, the duration of each user's session with the domain, and the current status of the session. With the help of this information, administrators can in no time isolate those sessions that are idle, but are unnecessarily consuming resources and adding to the server load. By closing such sessions, administrators can conserve server resources and reduce load.
	Average begins: Indicates the average number of transactions started by this client.	Number	
	Average commits : Indicates the average number of transactions committed by this client.	Number	
	Average aborts: Indicates the average number of transactions of this client that were aborted.	Number	Ideally, the value of this measure should be low. A high value could indicate the incidence of many transactional errors that are causing the transactions to abort. You may want to compare the value of this measure across clients to know which client's transactions aborted the most, thus leading administrators to those clients who may not have had a good experience with the server.

1.2.3 Tuxedo PIA Server Details Test

Using this test, administrators can figure out how many server processes in the domain are currently servicing requests, how many are active and ready for requests but have not been assigned requests yet, and the number of server processes that are not even ready for requests. With the help of the detailed diagnosis of this test, administrators can swiftly identify which server process is in which state presently, the number of requests being handled by each server process currently, and the total number of requests per server process for which processing is complete. Using this information, administrators can determine the following:

- Which server processes are idle?

- Which server processes are active, but are not processing any request?
- Which server process is handling too many requests currently?

<table>
<tr><td><b>Purpose</b></td><td>Using this test, administrators can figure out how many server processes in the domain are currently servicing requests, how many are active and ready for requests but have not been assigned requests yet, and the number of server processes that are not even ready for requests.</td></tr>
<tr><td><b>Target of the test</b></td><td>A Tuxedo application server</td></tr>
<tr><td><b>Agent deploying the test</b></td><td>An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td>1. <b>TEST PERIOD</b> - How often should the test be executed
2. <b>HOST</b> - Host name of the server for which the test is to be configured
3. <b>PORT</b> - Enter the port to which the specified <b>HOST</b> listens. The default port is 12345.
4. <b>PSADMIN HOME</b> – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the <b>psadmin.exe</b>. To enable the test to run the <b>psadmin.exe</b>, you need to configure the test with the full path to the directory in which the <b>psadmin.exe</b> resides. Therefore, in the <b>PSADMIN HOME</b> text box, specify the location of the <b>psadmin.exe</b>. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be <i>C:|ps|appsrv</i> and on Unix, your specification can be: <i>opt/ps/appsrv</i>.
5. <b>DOMAIN</b> - An application server <b>DOMAIN</b> is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server <b>DOMAIN</b> it needs to monitor.
6. <b>DETAILED DIAGNOSIS</b> - To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the On option. To disable the capability, click on the Off option.
The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:
• The eG manager license should allow the detailed diagnosis capability
• Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.</td></tr>
<tr><td><b>Outputs of the test</b></td><td>One set of results for the application server domain being monitored</td></tr>
</table>

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Active servers: Indicates the number of server processes that are active, but are not processing any request currently.	Number	Use the detailed diagnosis of this measure to know which are the active server processes.
	Running servers: Indicates the number of server processes that are actively servicing requests presently.	Number	Use the detailed diagnosis of this measure to know which server processes are running currently.
	Idle servers: Indicates the number of server processes that are not even ready for servicing any requests currently.	Number	Use the detailed diagnosis of this measure to know which are the idle server processes.
	Other servers: Indicates the number of server processes that are in any other state.	Number	Use the detailed diagnosis of this measure to know which server processes are in a state other than *Active, Running, or Idle*.

1.2.4 Tuxedo PIA Database Connections Test

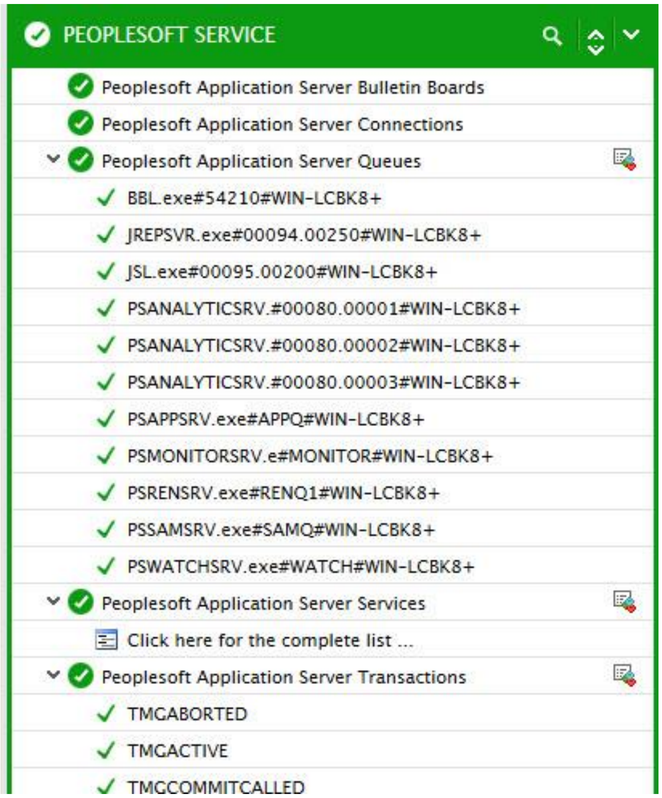
If the Tuxedo application server is unable to connect to the database owing to the absence of adequate connections in the connection pool, then critical server operations may fail. Using the **Tuxedo PIA Database Connections** test, administrators can periodically monitor the usage of each connection pool and promptly detect when a pool runs short of connections. This way, the test provides useful pointers to resizing connection pools.

Purpose	Helps administrators periodically monitor the usage of each connection pool and promptly detect when a pool runs short of connections. This way, the test provides useful pointers to resizing connection pools.
Target of the test	A Tuxedo application server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST** - Host name of the server for which the test is to be configured 3. **PORT** - Enter the port to which the specified **HOST** listens. The default port is 12345. 4. **PSADMIN HOME** – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the **psadmin.exe**. To enable the test to run the **psadmin.exe**, you need to configure the test with the full path to the directory in which the **psadmin.exe** resides. Therefore, in the **PSADMIN HOME** text box, specify the location of the **psadmin.exe**. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be *C:\ps\appsrv* and on Unix, your specification can be: *opt/ps/appsrv*. 5. **DOMAIN** - An application server **DOMAIN** is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server **DOMAIN** it needs to monitor.		
Outputs of the test	One set of results for every *Group_name:Pool_name* auto-discovered from the application server domain being monitored		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Max connection pool size: Indicates the maximum number of connections in this pool.	Number	
	Current connection pool size: Indicates the current size of this pool.	Number	
	Used connections: Indicates the number of used connections relative to the size of the pool.	Number	
	Free connections: Indicates the percentage of unused connections in this pool.	Percent	A high value is desired for this measure. If the value is low, it indicates abnormal usage of the connection pool. You may want to consider resizing the pool, so that sufficient connections are always available in the pool.

1.3 The Tuxedo PIA Service Layer

Using the tests mapped to this layer, administrators can determine the current configuration of the domain, the load on the conversational server in the domain, the status of transactions to the domain, the length of the queues in the domain, and the status and load of the services in the domain.



PEOPLESOFT SERVICE

Peoplesoft Application Server Bulletin Boards
Peoplesoft Application Server Connections
Peoplesoft Application Server Queues
✓ BBL.exe#54210#WIN-LCBK8+
✓ JREPSVR.exe#00094.00250#WIN-LCBK8+
✓ JSL.exe#00095.00200#WIN-LCBK8+
✓ PSANALYTICSRV.#00080.00001#WIN-LCBK8+
✓ PSANALYTICSRV.#00080.00002#WIN-LCBK8+
✓ PSANALYTICSRV.#00080.00003#WIN-LCBK8+
✓ PSAPPSRV.exe#APPQ#WIN-LCBK8+
✓ PSMONITORSRV.e#MONITOR#WIN-LCBK8+
✓ PSRENSRV.exe#RENQ1#WIN-LCBK8+
✓ PSSAMSRV.exe#SAMQ#WIN-LCBK8+
✓ PSWATCHSRV.exe#WATCH#WIN-LCBK8+
Peoplesoft Application Server Services
Click here for the complete list ...
Peoplesoft Application Server Transactions
✓ TMCABORTED
✓ TMCACTIVE
✓ TMCCOMMITCALLED

Figure 1.4: The tests mapped to the Tuxedo PIA Service layer

1.3.1 Tuxedo PIA Bulletin Boards Test

When any Tuxedo application server domain is booted, the first process to be started is the Bulletin Board Liaison process, called BBL. This process is the heart of the application server. It reads the configuration of the domain from a binary configuration file, which in a PeopleSoft domain is called PSTUXCFG. The BBL then establishes a shared memory segment, referred to as the Bulletin Board (BB) or Management Information Base (MIB), some message queues (determined by the specifications set out in the configuration file), and two semaphores. The Bulletin Board holds all the information about the rest of the application server domain including servers, services and their loads and priorities, clients and transactions. It is used as a form of database by the application server processes to determine how they should behave.

By monitoring how requests to a domain are utilizing the server processes, services, and interfaces registered with the Bulletin Board, administrators can ascertain the load on the domain, and whether the domain has been sized right to handle the load. This is exactly what the **Tuxedo PIA Bulletin Boards** test helps administrators achieve. This test reports the current configuration of the domain by reading the bulletin board. In addition, the test also keeps an eye on the usage of the server processes, services, and interfaces that the domain has been configured with, and proactively alerts administrators to probable processing bottlenecks in the domain, along with accurate pointers to the cause of the bottleneck – is it because the domain does not have enough server processes? services? Or interfaces? Based on the test’s findings, administrators can then proceed to reconfigure the domain (if required) to ensure rapid processing of requests.

Purpose	Reports the current configuration of the domain by reading the bulletin board. In addition, the test also keeps an eye on the usage of the server processes, services, and interfaces that the domain has been configured with, and proactively alerts administrators to probable processing bottlenecks in the domain, along with accurate pointers to the cause of the bottleneck – is it because the domain does not have enough server processes? services? Or interfaces? Based on the test’s findings, administrators can then proceed to reconfigure the domain (if required) to ensure rapid processing of requests		
Target of the test	A Tuxedo application server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST** - Host name of the server for which the test is to be configured 3. **PORT** - Enter the port to which the specified **HOST** listens. The default port is 12345. 4. **PSADMIN HOME** – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the **psadmin.exe**. To enable the test to run the **psadmin.exe**, you need to configure the test with the full path to the directory in which the **psadmin.exe** resides. Therefore, in the **PSADMIN HOME** text box, specify the location of the **psadmin.exe**. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be *C:|ps|appsrv* and on Unix, your specification can be: *opt/ps/appsrv*. 5. **DOMAIN** - An application server **DOMAIN** is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server **DOMAIN** it needs to monitor.		
Outputs of the test	One set of results for the application server domain being monitored		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Total servers: Indicates the total number of server processes registered with the bulletin board.	Number	A server process is an executable code that receives and processes incoming transaction requests. The number of server processes of each type is configurable and typically varies within a domain, depending on the

	Used servers : Indicates the number of server processes that are actively processing requests.	Number	requirements of your application or the main purpose of the domain. To gauge the adequacy of the server process configuration, you need to use the **Used Servers** and **Free servers** measures. ` If the **Used servers** count is very high and the percentage of **Free servers** is low, it implies that the load on the domain is high, but the domain has not been configured with sufficient server processes to handle the load. If the percentage of free server processes is allowed to dip further, request queues on the domain wll only grow longer, delaying request processing or even bringing it to a standstill. To avoid such adversities, you have to probe deeper to identify the server process types that are in demand and configure more server processes of that type in the domain.
	Free servers : Indicates the percentage of server processes that are currently idle.	Percent	
	Total services: Indicates the total number of services registered with the bulletin board.	Number	The server process carries out a request by making calls to a service, such as MgrGetObject. The server process waits for the service to complete, then returns information to the device that initiated the request, such as a browser. A high value for the **Used services** measure indicates that many services are in use, which in turn implies that the load on the domain is high. If the percentage of **Free services** is also low, it indicates that the domain has not been configured with adequate services to complete the requests. In such situations, you will have to consider increasing the number of services configured for the domain.
	Used services : Indicates the number of services that are actively processing requests to the domain.	Number	
	Free services : Indicates the percentage of services idle in the domain.	Percent	
	Total interfaces : Indicates the number of network interfaces registered with the bulletin board.	Number	
	Used interfaces : Indicates the number of network interfaces currently in use in the domain.	Number	

	Free interfaces : Indicates the percentage of network interfaces idle in the domain.	Percent	A high value is desired for this measure. If the value of this measure is low and the value of the **Used interfaces** measure is close to that of the **Total interfaces** measure, it indicates that the domain has very few interfaces left for handling any additional traffic. Under such circumstances, you may want to assign more network interfaces to the domain.
	Number of queues : Indicates the current number of queues in the domain.	Number	While a server process waits for a service to complete, other transaction requests wait in a queue until the current service completes. A service may take a fraction of a second to complete or several seconds, depending on the type and complexity of the service. When the service completes, the server process is then available to process the next request in the corresponding queue. The existence of multiple queues in a domain could be sign of a processing bottleneck in the domain, caused by poor domain configuration. If the value of this measure is high therefore, you may want to check the value of the **Free servers, Free services,** and **Free interfaces**, to figure out what resource the domain requires to ensure that there are no hiccups in request processing.
	Number of groups : Indicates the number of groups currently available in the domain.	Number	

1.3.2 Tuxedo PIA Server Connections Test

The PSSAMSRV application server process running within an application server domain is what Tuxedo refers to as a conversational server. This server process is capable of conversing with other servers by submitting requests to them, or receiving requests from them. It provides only one service, *SqlAccess*. This server is used to administer the process scheduler tables, and to allocate version numbers on PeopleSoft objects during development and upgrade. When a job is submitted, or when job and process scheduler statuses are updated in the process monitor, the SQL to perform those updates are submitted directly by PSSAMSRV.

By tracking connections to the PSSAMSRV application server process in a domain over time, an administrator can assess the typical load on the server process. Based on this assessment, the administrators can, if required, fine-tune the settings of the PSSAMSRV so as to enhance its request-handling capability. This is where the **Tuxedo PIA Server Connections** test helps. This test periodically reports the current connection load on the PSSAMSRV application server process. By using this test to monitor how the load on PSSAMSRV varies with time, administrators can gauge the typical load on the PSSAMSRV application server process and can accordingly figure out whether/not additional PSSAMSRV application server process instances will be required for handling the load. Additionally, administrators can use the detailed diagnosis of this test to know which two server processes are conversing using the conversational server process in the domain.

Purpose	Periodically reports the current connection load on the PSSAMSRV application server process. By using this test to monitor how the load on PSSAMSRV varies with time, administrators can gauge the typical load on the PSSAMSRV application server process and can accordingly figure out whether/not additional PSSAMSRV application server process instances will be required for handling the load.		
Target of the test	A Tuxedo application server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST** - Host name of the server for which the test is to be configured 3. **PORT** - Enter the port to which the specified **HOST** listens. The default port is 12345. 4. **PSADMIN HOME** – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the **psadmin.exe**. To enable the test to run the **psadmin.exe**, you need to configure the test with the full path to the directory in which the **psadmin.exe** resides. Therefore, in the **PSADMIN HOME** text box, specify the location of the **psadmin.exe**. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be *C:|ps|appsrv* and on Unix, your specification can be: *opt/ps/appsrv*. 5. **DOMAIN** - An application server **DOMAIN** is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server **DOMAIN** it needs to monitor. 6. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the On option. To disable the capability, click on the Off option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.		
Outputs of the test	One set of results for the application server domain being monitored		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**No of connections:** Indicates the number of conversational connections currently open on the Tuxedo server.	Number	This is a good indicator of the current workload of the PSSAMSRV application server process. To know the details of the current connections, use the detailed diagnosis of this test. This reveals the *Originator group ID, the Original logical machine ID, Receiver group ID and logical machine ID, Data transmitted by originator and receiver, and service name*. With the help of the detailed metrics, administrators can identify which originator-receiver pair is transmitting the maximum data. If the value of this measure increases consistently, and the PSSAMSRV application server process appears to service requests slowly, it could be because enough server instances are not available to handle the heavy load. It is mandatory that at least one instance of the PSSAMSRV application server process be started when a domain is booted. Depending upon the variations in the value of this measure over time, you may want to increase the number of instances to be started at boot time, so that the load is balanced and serviced quickly.

1.3.3 Tuxedo PIA Server Queues Test

When a Tuxedo application server process carries out a request, it makes a call to a service, such as MgrGetObject. The server process waits for the service to complete, then returns information to the device that initiated the request, such as a browser. While a server process waits for a service to complete, other transaction requests wait in a queue until the current service completes. A service may take a fraction of a second to complete or several seconds, depending on the type and complexity of the service. When the service completes, the server process is then available to process the next request in the corresponding queue.

If the length of a queue keeps increasing, it is a sign that the corresponding server process is unable to process requests quickly. Therefore, to be able to swiftly spot a probable processing bottleneck in an application server domain, administrators must monitor the queues in the domain continuously, isolate those queues that are growing in length, and identify which server process is using that queue. This can be achieved using the **Tuxedo PIA Server Queues** test. This test auto-discovers the queues used by each server process in a domain, and for every queue, reports the number of requests in the queue and the number of instances of that server process sharing the queue. This way, the test rapidly leads administrators to a potential slowdown in request processing by a domain, and helps them accurately isolate which server process is contributing to the slowdown.

Purpose	Auto-discovers the queues used by each server process in a domain, and for every queue, reports the number of requests in the queue and the number of instances of that server process sharing the queue. This way, the test rapidly leads administrators to a potential slowdown in request processing by a domain, and helps them accurately isolate which server process is

<table>
<tr><td></td><td colspan="3">contributing to the slowdown.</td></tr>
<tr><td><b>Target of the test</b></td><td colspan="3">A Tuxedo application server</td></tr>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> - How often should the test be executed
2. <b>HOST</b> - Host name of the server for which the test is to be configured
3. <b>PORT</b> - Enter the port to which the specified <b>HOST</b> listens. The default port is 12345.
4. <b>PSADMIN HOME</b> – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the <b>psadmin.exe</b>. To enable the test to run the <b>psadmin.exe</b>, you need to configure the test with the full path to the directory in which the <b>psadmin.exe</b> resides. Therefore, in the <b>PSADMIN HOME</b> text box, specify the location of the <b>psadmin.exe</b>. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be <i>C:|ps|appsrv</i> and on Unix, your specification can be: <i>opt/ps/appsrv</i>.
5. <b>DOMAIN</b> - An application server <b>DOMAIN</b> is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server <b>DOMAIN</b> it needs to monitor.</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every auto-discovered <i>Server_Process#Queue_name#Machine_ID</i> in the application server domain being monitored</td></tr>
<tr><td><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td></td><td><b>Servers connected:</b>
Indicates the number of instances of this server process that are currently connected to this queue.</td><td>Number</td><td>This is a good indicator of the number of instances of a server processes that are currently running in a domain.</td></tr>
<tr><td></td><td><b>Average queue length :</b>
Indicates the average length of this queue.</td><td>Number</td><td>A consistent rise in the value of this measure is a cause for concern, as it indicates that this server process is not able to process the requests in this queue quickly. Under such circumstances, you may need to add more instances of that particular server process to improve its processing capability.</td></tr>
</table>

	Service requests queued : Indicates the number of service requests in this queue.	Number	
	All requests in queue : Indicates the total number of requests in this queue.	Number	

1.3.4 Tuxedo PIA Services Test

Server processes invoke services to perform application logic and issue SQL to the RDBMS. When a PeopleSoft application sends a request to the application server, it sends a service name and a set of parameters, such as MgrGetObject and its parameters. Tuxedo then queues the transaction request to a specific server process that is designed to handle certain services. When a server process boots, it advertises to the system the predefined services it handles.

If the server process is down or is not ready to handle any requests, then all services it handles will also be unavailable; this can severely hamper request processing by the server. Moreover, if even a single service processes requests slowly, user experience with the application server will be adversely impacted. Using the **Tuxedo PIA Services** test, administrators can quickly detect such anomalies and initiate measures to correct them, so that requests are serviced promptly. Additionally, the test also periodically reports the count of requests handled by each service, thus highlighting the most popular/busiest services delivered by the application server domain.

Purpose	If the server process is down or is not ready to handle any requests, then all services it handles will also be unavailable; this can severely hamper request processing by the server. Moreover, if even a single service processes requests slowly, user experience with the application server will be adversely impacted. Using the **Tuxedo application server Services** test, administrators can quickly detect such anomalies and initiate measures to correct them, so that requests are serviced promptly. Additionally, the test also periodically reports the count of requests handled by each service, thus highlighting the most popular/busiest services delivered by the application server domain.
Target of the test	A Tuxedo application server
Agent deploying the test	An internal agent

<table>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">
1. <b>TEST PERIOD</b> - How often should the test be executed
2. <b>HOST</b> - Host name of the server for which the test is to be configured
3. <b>PORT</b> - Enter the port to which the specified <b>HOST</b> listens. The default port is 12345.
4. <b>PSADMIN HOME</b> – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the <b>psadmin.exe</b>. To enable the test to run the <b>psadmin.exe</b>, you need to configure the test with the full path to the directory in which the <b>psadmin.exe</b> resides. Therefore, in the <b>PSADMIN HOME</b> text box, specify the location of the <b>psadmin.exe</b>. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be <i>C:\ps\appsrv</i> and on Unix, your specification can be: <i>opt/ps/appsrv</i>.
5. <b>DOMAIN</b> - An application server <b>DOMAIN</b> is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server <b>DOMAIN</b> it needs to monitor.
</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every <i>group_name:server_process_name:service_name</i> associated with the monitored application server domain.</td></tr>
<tr><td><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretatione</b></td></tr>
<tr><td></td><td><b>Service status:</b>
Indicates the current status of this service.</td><td></td><td>The values that this measure can report and their corresponding numeric values are discussed in the table below:
<table>
<tr><th>Measure Value</th><th>Numeric Value</th></tr>
<tr><td>Available</td><td>100</td></tr>
<tr><td>Not available</td><td>0</td></tr>
</table>
<b>Note:</b>
By default, this measure reports the <b>Measure Value</b> mentioned above to indicate the current state of a server. The graph of this measure however, represents the same using the numeric equivalent only.</td></tr>
<tr><td></td><td><b>Requests handled:</b>
Indicates the number of requests handled by this service since the last measurement period.</td><td>Number</td><td>A very high value could indicate that the service is overloaded with requests. You can compare the value of this measure across services to know which services have the maximum number of requests; this way, you can find out which services are most popular or are the busiest.</td></tr>
</table>

	Request rate: Indicates the rate at which requests were handled by this service.	Requests/Sec	A consistent dip in the value of this measure is a cause for concern as it indicates that the service is unable to process requests quickly. This could hint at a processing slowdown.

1.3.5 Tuxedo PIA Server Transactions Test

A Tuxedo application server consists of numerous PeopleSoft server processes, grouped in domains. Each server process within a domain provides unique processing abilities, enabling the application server to respond effectively to a multitude of transaction requests generated throughout the PeopleSoft architecture.

When a PeopleSoft application sends a request to the application server, it sends a service name and a set of parameters, such as MgrGetObject and its parameters. The application server then queues the transaction request to a specific server process that is designed to handle certain services. The server process then establishes a persistent connection to a PeopleSoft database, and this connection acts as a generic SQL pipeline that the server process uses to send and receive SQL to process the transaction request.

If any of the transaction requests handled by a domain fail, administrators should be able to proactively detect the failure and prevent it from occurring, so that workflow is not disrupted and user productivity is not hit. The **Tuxedo PIA Server Transactions** test helps administrators with this and more! This test periodically checks the status of transactions on a configured application server domain, and reports how many transactions are in which state presently. This can reveal the number of active transactions on the domain, based on which the current load on the domain can be determined. In addition, the test also warns administrators of transactional errors that may cause transactions to abort, thus enabling administrators to take pre-emptive steps in this regard.

Purpose	Periodically checks the status of transactions on a configured application server domain, and reports how many transactions are in which state presently.
Target of the test	A Tuxedo application server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** - How often should the test be executed 2. **HOST** - Host name of the server for which the test is to be configured 3. **PORT** - Enter the port to which the specified **HOST** listens. 4. **PSADMIN HOME** – To collect metrics from a Tuxedo application server, this test runs PeopleSoft Server Administration (PSADMIN) commands on the target server. The first step towards this is to launch the **psadmin.exe**. To enable the test to run the **psadmin.exe**, you need to configure the test with the full path to the directory in which the **psadmin.exe** resides. Therefore, in the **PSADMIN HOME** text box, specify the location of the **psadmin.exe**. Typically, this will be the install directory of the Tuxedo application server. For instance, on Windows, your specification can be *C:\ps\appsrv* and on Unix, your specification can be: *opt/ps/appsrv*. 5. **DOMAIN** - An application server **DOMAIN** is the collection of server processes, supporting processes, and resource managers that enable connections to the database. A single application server machine can support multiple application server domains running on it. A server process is executable code that receives incoming transaction requests. The server process carries out a request by making calls to a service. Using a managed Tuxedo application server, you can monitor only those server processes and transactions that pertain to a single domain. This is why, the eG agent needs to be explicitly configured with the application server **DOMAIN** it needs to monitor. 6. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the On option. To disable the capability, click on the Off option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.		
Outputs of the test	One set of results for every transaction status code returned by the application server		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**Transactions:** Indicates the number of transactions that are in this status currently.	Number	If the value of this measure is high for the **TMGACTIVE** descriptor, it implies that many transactions are currently active on the domain. You may want to observe variations to this load over time, so that, if demand appears to be increasing, you can consider configuring multiple application server domains per database, for redundancy, fail-over, and performance reasons. On the other hand, if the value of this measure is very high for the **TMGABORTED** or the **TMGTOBEABORTED** descriptors, it means that many transactions have been aborted or are about to be aborted. This is a cause for concern, and requires further investigation. Use the detaied diagnosis of this measure to know the details of transactions in various states. This way, you can accurately pinpoint the transactions that are about to be aborted.

2

Conclusion

This document has described in detail the monitoring paradigm used and the measurement capabilities of the eG Enterprise suite of products with respect to **Tuxedo application server** and **PeopleSoft Process Scheduler**. For details of how to administer and use the eG Enterprise suite of products, refer to the user manuals.

We will be adding new measurement capabilities into the future versions of the eG Enterprise suite. If you can identify new capabilities that you would like us to incorporate in the eG Enterprise suite of products, please contact support@eginnovations.com. We look forward to your support and cooperation. Any feedback regarding this manual or any other aspects of the eG Enterprise suite can be forwarded to feedback@eginnovations.com.