Using the Conductor

Table Of Contents

Conductor

Overview of the QALoad Conductor

Use the *QA*Load Conductor to configure, run, and monitor a load test that utilizes the scripts created in the Script Development Workbench. The Conductor controls the *QA*Load Players and manages tests while they are running. When the Conductor process stops for any reason during a load test, the associated Player processes automatically terminate.

The primary windows in the Conductor are:

Conductor Start Page - This is the page that appears when you first open the Conductor. From the Start Page, you can open an existing Conductor session, create a new session, configure the Conductor, or open the Test Configuration Wizard.

Once you create or open a session, the main Conductor window appears. The Conductor's interface is dynamic — it changes depending whether you are setting up a test or running a test.

Test Setup Interface - This is the first window that appears when you open a session in the Conductor. Before running a test, you must set up general test options, configure Player workstations, assign compiled test scripts to Players, and set up monitoring options. You then save the test setup in a file called a session ID. Once you have configured and saved a test session ID, you can reuse it without re-entering any test information.

The Test Setup Interface presents two ways you can enter information about your test:

- The Visual Designer is the Conductor's main window. The Visual Designer graphically displays a collection of icons and nodes that represent your test session. Use this to enter information about your test and set up the machines and scripts for the test.

- The Grid View window presents another way to perform these functions. The Grid View is a dockable window that contains two tabs in which you can enter the information for your test session.

Runtime Window - While a test is running, the Conductor interface changes to the Runtime Window, which facilitates monitoring of individual machines and Players, and displays real-time test results. You can view default graphs of performance data that are created for you by the Conductor and create custom graphs based on the data being collected during the test. You can save custom graph layouts in the session ID file and reuse them in future tests.

Using the Conductor Start Page

When you first start the Conductor, the Start Page displays. This provides a concise selection of options to help you start defining or modifying a Conductor session.

From the Start Page, you can perform the following tasks:

Recent Session

In the Recent Sessions section, you can:

> Create a new test session
>
> Open an existing session

Tasks

In the Tasks section, you can:

> Open the Test Configuration Wizard
>
> Create monitoring tasks
>
> Edit monitoring tasks
>
> Discover and verify Player machines
>
> Configure the Conductor session options

Help

Use the selections in the Help section to access the QALoad online help.

> **Note**: If you select the **Don't show this panel again** option, you'll skip the start page and go directly to the Conductor Session Startup dialog box when you open the Conductor. To display the Start Page again when you open the Conductor, you must open a session, go to **Tools**>**Options**>**Startup**, and select **Show Start Page**.

About Setting Up a Test

When you set up a load test, you set options related to general Conductor behavior as well as information about your specific test environment. Before you can successfully set up a load test, you must have recorded and compiled one or more test scripts. For information about recording a test script, see Developing Scripts.

Determining General Conductor Behavior

General Conductor options you set are applicable for all tests run until you change the options. Conductor options are related to the following:

> Viewing options for real-time results
>
> Global Player options
>
> Player machine performance data
>
> Options for runtime reporting
>
> And more...

All of the above information, and more, can be configured on the Conductor's Options dialog box.

Setting Up a Specific Test Session

To prepare the Conductor for a specific test, save information and parameters specific to that test into a reusable session ID file (.id). You must enter the following types of information to set up a test's session ID file:

> General information about the test such as a description, the size of the database, the length of the test, and any notes or comments
>
> Information about the test script(s) included in the test, including script name, middleware/protocol type, pacing, whether to include external data, and so on
>
> Information about the workstations where the *QA*Load Players reside, including which script is assigned to each workstation, how many virtual users are assigned to each workstation, the machine name, and so on

All of the above information can be entered and saved when you set up a test from the Conductor's main window or by using the Test Configuration Wizard. Once you open the new session in the Conductor, you can do the following:

> (Optional) configuration for server monitoring
>
> (Optional) integration with other products

Generating Random Number Seeds

Random number seeds are used to inject random delays in script execution for each load test. The seed (or value) is automatically generated by *QA*Load. The random value used within the end of transaction function is used to generate the pacing time. The Player uses a system-generated sequence of numbers, so that each virtual user (VU) has its own seed value.

Setting Up a Test Session

You can enter all the information necessary for your session ID file in the Conductor using one of the following methods:

Grid View window

Visual Designer

Test Configuration Wizard

Configuring the Conductor

There are several settings for the Conductor that you should review before beginning your load test.

To set Conductor session options that are not specific to one test:

1. Do one of the following:

   - From the Conductor Start Page, click **Session Options** in the **Tasks** area.

     OR

   - With a session open, click **Tools>Options**.

2. On the Options dialog box, set options related to post-test activity, warnings and prompts, runtime grids, timing settings, interface refresh intervals, Conductor/Player communications, monitoring intervals, and more. For detailed descriptions of the options that are available, see Options dialog box.

3. When you are finished, click **OK** to save your changes. Any options you set apply to all tests until you change them.

Checking Out Virtual User Licenses

If you are licensed to run multiple copies of the Conductor, for example so different work groups have access to *QA*Load, you can check out virtual user licenses before running a load test to ensure that enough are available for your test run.

If you do not choose to check out your licenses before starting a test, *QA*Load prompts you after you start the test and attempts to check out the appropriate number of licenses. We recommend that you check your licenses out manually before starting so you can be sure you have enough virtual users available before beginning your test run.

To check out virtual user licenses:

1. From the Conductor menu, select **Tools>Licensing**. The License Information dialog box appears.

    - If you are licensed for concurrent licensing (multiple Conductors) the Conductor queries your license server to determine how many licenses are currently available, and returns the results to this dialog box. Go to step 2.

    - If you have a node-locked license (a single Conductor), then most of the options on this dialog box are unavailable, as you will not need to, or be able to, check out virtual user licenses. All virtual users for which you are licensed are available only to this Conductor. Click **OK** to return to your test setup.

2. In the Licensing Operations area, select **Check Out Virtual User Licenses**, then type how many virtual user licenses you want to check out in the **Number of Licenses** field.

3. Click **Check Out**. The licenses are checked out to your Conductor, and are unavailable to any other Conductor workstations on the network.

When you are done using your licensed virtual users, check them back in so they are once again available to other Conductor workstations on your network.

To check in virtual user licenses:

1. From the Conductor menu, choose **Tools>Licensing**. The License Information dialog box appears.

2. If you have licenses checked out, the Check in Virtual User License option is automatically selected for you.

3. Click **Check In**. The licenses are made available to other Conductor workstations on the network.

Running the Conductor from the Command Line

The following procedure describes how to run the Conductor from the command line.

To start the Conductor from the command prompt:

Type `conductor <session_file_name> /l /e /a /t`

The applicable parameters are defined in the following table.

Parameter	Definition
`/l` (Optional)	Creates a log file showing error messages and test status.
`/e` (Optional)	Exits the Conductor when the test completes.
`/a` (Optional)	Launches Analyze when the test completes.
`/t` (Optional)	Executes Conductor at a set time. Valid time formats are `/txx:xx` or `/txx/xx/xx /txx:xx`.

The Conductor **start page** appears.

Setting Up a Test

Creating a New Session

To create a new load test session:

1. Do one of the following:

    - From the Start page, click **New**.

    - From the Visual Designer window, click **File>New**. A confirmation dialog box prompts you to Save or discard changes to your current session.

    The Create New Session dialog box appears.

2. In the **Name** field, type a name for the new session.

3. (Optional) In the **Description** field, type a description for the new session.

4. (Optional) Select **Launch Test Configuration Wizard** to start the Test Configuration Wizard, which walks you through creating a load test session. If you do not select this option, the new test session opens in Conductor's Visual Designer, where you can manually configure the load test session.

Opening an Existing Session

When you open the Conductor, the Conductor Start Page appears. Use these procedures to open an existing session.

> **Note**: If you choose not to display this page, the Conductor opens to the Visual Designer window. You can choose to show the Start Page again in the Startup page of the Conductor Sessions Options dialog box.

To create a new session from the Conductor Start page:

1. Do one of the following:

    - In the Recent Sessions area, click the name of the session to open.

    OR

    - Click **Open Session**, then select the name of the appropriate saved session, and click **Open**.

2. The Conductor's Visual Designer opens with the session you selected displayed. You can make any changes to the session or run the load test.

To open an existing session in the Visual Designer:

1. From the Visual Designer window, click **File>Open**. A confirmation dialog box prompts you to Save or discard changes to your current session.

2. Select the name of the appropriate saved session, and click **Open**. The selected test session opens in Conductor's Visual Designer, where you can manually configure the load test session.

About the Test Configuration Wizard

The Test Configuration Wizard consists of a series of screens that guide you through the process of setting up a load test. Use the Test Configuration Wizard to select scripts, configure the script's transaction settings, select players or groups to assign to the test, and specify desired Virtual User configurations. You also can add scripts to an existing session using the Test Configuration Wizard. Once you setup your load test with the Test Configuration Wizard, you can run the test immediately without additional configuration.

You can use the Test Configuration Wizard to:

Create a New Test Session Using the Test Configuration Wizard

Modifying an Existing Test Session Using the Test Configuration Wizard

Creating a Test Session Using the Test Configuration Wizard

The Test Configuration Wizard guides you through the process of setting up a load test. Use the series of screens in the Test Configuration Wizard to:

1. Select scripts
2. Configure the script's transaction settings
3. Select players or groups to assign to the test
4. Specify desired Virtual User configurations

You can open the Test Configuration Wizard from the Conductor Start page or with a test session open in the Conductor.

To access the Test Configuration Wizard from the Conductor Start page:

1. Do one of the following:

   - In the Tasks area, click **Test Configuration Wizard** to open the first screen of the wizard.

   OR

   - In the Recent Sessions area, click **New** to open the Create New Session dialog box. Select the **Launch Test Configuration Wizard** option.

     ■ **Note**: If you do not select the Launch Test Configuration Wizard option, the new test session opens in Conductor's Visual Designer, where you can manually configure the load test session.

2. In the **Name** field, type a name for the new session.

3. (Optional) In the **Description** field, type a description for the new session.

4. Click **Next** to start configuration of your test session.

To access the Test Configuration Wizard with a session open in Conductor:

Click **File>New**, then follow steps 2 through 4 above.

Modifying a Test Session Using the Test Configuration Wizard

You can use the Test Configuration Wizard to add a script to an open test session by following the steps for each screen in the Test Configuration Wizard.

To start the process of adding a script using the Test Configuration Wizard:

On the toolbar, click the **Test Configuration Wizard** button. The Select Script to Configure dialog box appears.

Anticipating Error Conditions

You know before beginning a load test that errors are a possibility, but you may not always want them to stop your progress during testing.

*QA*Load helps anticipate error conditions and determine, before running the test, how Players react to non-fatal errors. By setting one option, you can instruct a Player to continue as if no error was encountered, stop running immediately, or restart at the beginning of the transaction.

Note: When the Conductor process stops for any reason during a load test, the associated Players automatically terminate.

Managing Large Amounts of Test Data

With a large number of virtual users, it is possible to create a timing file containing hundreds of thousands of timing records for each checkpoint. Attempting to graph just a few of those checkpoints can slow down *QA*Load Analyze considerably.

For example, if a timing file contained 250,000 timing records for each data point, attempting to graph even one checkpoint means that *QA*Load Analyze must paint 250,000 lines on the graph. Since most monitors only have 1024 pixels across the screen, the 250,000 data points would mostly be plotted atop one another and the results would be unreadable.

Now imagine attempting to graph the data of several data points of that size. The sheer amount of data could easily overwhelm a workstation. And every time you move the window, resize the window, or right-click on the graph, *QA*Load Analyze has to re-draw the graph. You could conceivably spend enormous amounts of time simply attempting to graph data.

To make large amounts of data manageable, *QA*Load Analyze provides an option that allows you to determine how to thin data. That is, how to determine how many data points to plot.

When your test is running and your Conductor is collecting timing information from your Player machines, the sheer amount of data can take up more of your resources than you would like to expend. Use *QA*Load's Timing Data Thinning option to thin the amount of timing data being transferred back to the Conductor during the test so that your test can run longer without stressing your resources.

Removing Used Datapool Records After a Test

You can remove used datapool records from a Central datapool after a test completes by setting the Strip Datapool function before you run the test. Use this function when running a test where you have data in the external datapool that can only be used once by one virtual user at a time. (For example, when running transactions that have unique data constraints.) When activated, the Strip Datapool function marks each piece of data in the datapool that is used during your test. When the test is over, the Strip Datapool function prompts you to remove the identified used data from the datapool. If you run the test again, only new data is used for your subsequent test.

To use the Strip Datapool function:

1. With the current test's session ID file open, do one of the following:

   - Click the script icon for the appropriate script to display the Script Properties panel on the right-hand of the window. In the **External Data** area, click **Central Datapool** field and click the **browse [...]** button in the adjacent field.

   OR

   - In Grid View window's **Script Assignment** tab, click the **browse [...]** button in the **External Data** field of the appropriate script. When the Script Properties dialog box appears, click **Central Datapool**.

2. In the Central Datapool dialog box, click the **browse [...]** button to select the Central Datapool file, then select the **Strip** check box. Click **OK**.

3. At the end of your test, a Strip Datapools prompt appears asking if you wish to go to the Strip Datapools screen. Click **Yes**.

4. The Strip Central Datapool dialog box appears with the Strip option selected. Click the **Strip** button.

5. When you are finished, click **Close**.

Validating Scripts in Conductor

Before running a test, you should run your script in a simple test to ensure that it runs without errors. You can validate UNIX or Win32 scripts in the Conductor.

Enabling Expert User

When you enable the Expert User, this VU collects more detailed information about requests that are made while the script is running. Every main request and subrequest logs the amount of server and network time used. This helps diagnose why page loads may be taking longer than expected. You enable Expert User from the Conductor, either before or during a load test. Expert User uses the existing custom counter support so Conductor can graph the custom counter information. Once the load test is complete, you can view the data in Analyze.

 Note: Currently, Expert User capability is provided only for the WWW middleware.

You can enable the Expert User:

Before a load test begins from the Visual Designer or Grid View windows.

During a load test from the Runtime window.

To enable Expert User before the load test begins:

1. Do one of the following:

    ▪ Using the **Visual Designer**:

    a.   Click an individual player machine in the script test setup node to display the Player Properties panel on the right-hand side of the window.

    b.   Click the **browse [...]** button in the **Expert User Options** field to open the Expert User Options dialog box.

OR

    ▪ Using the **Grid View** window:

    a.   Click the **Machine Assignment** tab.

    b.   In the **Middleware** field for the appropriate WWW script, click the **browse [...]** button to display the Expert User Options dialog box.

2.   Click **Enable Expert User timings**.

3.   In the **Virtual User Number** field, type the Virtual User (VU) number to represent the Expert User. The default VU number is zero (0).

4.   Click **OK**.

To enable Expert User during the load test:

1. In the Runtime window, click **Actions>Set Expert User Options**. The **Update Expert User Options** dialog box displays listing all the scripts that support Expert User counters.

2. Click the scripts in which you want to enable Expert User, then click **OK**.

> 🔷 **Note**: Selecting Expert User Scripts at the top level enables or disables expert user on all associated scripts.

Overview of Expert User

Expert User provides an easy, logical guide for drilling down to the root performance problems for applications. It enables you to break web pages down into their individual components, providing more detailed response time data. Response time for each component is broken into network and server time.

More detailed information helps troubleshoot application performance problems. The ability to see timing files on a component level can spotlight where the majority of time is being spent. A breakdown of network and server times per component can identify areas for improvement in either the network or server hardware or configuration, or in application performance.

The main functionality is provided by a special virtual user (VU). When you enable the Expert User, this VU collects more detailed information about requests that are made while the script is running. Every main request and subrequest logs the amount of server and network time used. This helps diagnose why page loads may be taking longer than expected. For example, a particular subrequest, such as css, gif, html, and so forth, may be taking more time to download from the server than other requests. Expert User data can show you this. It also can help you determine whether the problem is a network or a server problem.

You enable Expert User from the Conductor, either before or during a load test. Expert User uses the existing custom counter support so Conductor can graph the custom counter information.

Once the load test is complete, you can view the data in Analyze. The Analyze Workspace includes an Expert User tab, from which you can access detail reports and graphs on server and network data. The pre-defined reports include an Expert User report.

> 🔷 **Note**: Currently, Expert User capability is provided only for the WWW middleware.

Assigning Scripts to the Test Session

Script Assignment

Use the **Assign Scripts** button on the Visual Designer toolbar, or the **Script Assignment** tab in the Grid View window to set up any scripts that have previously been recorded and compiled. Any script you add here is included in your load test, and one virtual user is automatically assigned to your script. After setting up your scripts here, you must assign additional virtual users to your script from the **Properties** pane in the Visual Designer, or from the **Machine Assignment** tab in the Grid View window.

You can also add scripts to a test session using the Test Configuration Wizard. For more information, see About the Test Configuration Wizard.

Adding a Script to a Test

To add a script to a test session:

1. On the **Visual Designer** toolbar, click **Assign Scripts...**

   OR

   In the **Grid View** window, click the **Script Assignment tab**, then click **New** in the toolbar. The Assign Scripts dialog box displays.

2. In the **Middleware Type** box, select your middleware type.

3. From the list of available scripts that appears in the Available Scripts pane, highlight a script name and click ➡ . The script is moved to the Selected Scripts pane.

   🔷 **Note**: To select more than one script, hold down the **Ctrl** key and click then scripts to select, the click ➡ . To select all scripts in the Available Scripts pane, click ⬌ .

4. Click **Next** to display the script transaction configuration dialog box.

5. In the Selected Scripts pane, highlight a script.

6. In the **Transactions** field, specify the maximum number of transactions that you want each virtual user running this script to run. Once a workstation executes the number you specify, script execution continues with the line following the End_Transaction command rather than jumping to the beginning of the transaction loop.

7. Enter a value, in seconds, in the **Pacing** field. Pacing is the time interval between the start of a transaction and the start of the next transaction for each virtual user running a script.

8. Enter a value in the **Sleep Factor %** field to specify the percentage of any originally-recorded delay to preserve in the script (for example, a value of 80 means preserve 80% of the original delay). Valid values are 0-1000, or Random. The default value is 100%.

9. To apply these options to all scripts in the Selected Scripts pane, click **Apply this to all selected scripts**.

10. Click **Finish** to save your changes to the current session ID file.

Replacing a Script in a Test

You can replace a script in a test session using the Visual Designer or the Grid View window. Use the following procedure to replace a script in a test:

To replace a script using the Visual Designer:

1. In the Conductor's menu bar, click **Actions>Replace Script**. The Select Replacement Script dialog box appears.

2. In the **Middleware Type** field, select the middleware environment of the replacement script.

3. Select the replacement script, then click **OK**.

4. Assign the script to Player machines, if necessary.

To replace a script using the Grid View window:

1. In the Grid View window, click the Machine Assignment tab.

2. In the Script Name field, select the script to replace, then select the new script from the drop-down dialog box.

3. Assign the script to Player machines, if necessary.

Removing a Script or a Player from a Test

To remove a script from a test:

1. Right-click on the script icon for the appropriate script, then select **Remove Script**.

2. In the confirmation dialog box, click **Remove**.

To remove all scripts from a test:

1. Click **Actions>Remove All Scripts**.

2. In the confirmation dialog box, click **Remove**.

To remove a single Player from a test:

1. In the script test setup node, right-click on the Player to remove, then select **Remove Selected**.

2. In the confirmation dialog box, click **Remove**.

To remove all Players from a test:

1. Click **Actions>Remove All Players/Groups**.

2. In the confirmation dialog box, click **Remove**.

Assigning Player Machines

Machine Assignment

Use the **Assign Players/Groups** button on the Visual Designer toolbar, or use the **Machine Assignment** tab in the Grid View window to open the Assign Players/Groups dialog box and assign scripts to specific Player workstations. You also can drag and drop individual Player machines in selected script test setup nodes in the Visual Designer.

Saving Machine Configurations

After configuring the machines to use for a load test, you can save the machine configuration information into a configuration file (.cfg) that can be reused in later tests. This saves you significant time setting up later tests. A configuration file includes information about which machines on the network were used as Player machines. You can save multiple configurations under different names. By default, when first using *QA*Load, the Conductor uses a configuration file named `Default.cfg`. The Conductor saves any changes to your machine configurations to this file unless you save your configuration to a new file with a different name.

You can open or save .cfg files from the **Manage Players/Groups** dialog box. The .cfg field always displays the active configuration.

To create a new, empty .cfg file:

1. On the Conductor toolbar, click **Tools>Manage Players**. The Manage Players/Groups dialog box displays.

2. Click **File>New>Player** or click the New icon [icon] on the toolbar.

3. In the Player Information area of the window, type a name in the **Machine [hostname or IP]** field.

4. Click **File>Save as**... On the Save As dialog box, specify a name for the new file and click **Save**.

5. Add the necessary Player and agent machines using the fields and buttons on the **Manage Players/Groups** dialog box. The machines you configure are saved automatically to the file you just created.

To rename the current .cfg file:

1. On the Manage Players/Groups dialog box, click **File>Save As...**

2. On the Save As dialog box, specify a name for the new file and click **Save**.

3. Make any necessary changes to the configuration. Your changes are saved automatically to the file you just created.

To open a previously created .cfg file:

1. On the Manage Players/Groups dialog box, click **File>Open**. The Open Machine Configuration dialog box displays.

2. Choose the .cfg file to open.

Note: The .cfg file only stores information about Player machines. It does not store information specific to a test, such as script names or settings. Test specific information is saved in the session ID file. A session ID file for a specific test saves the name of the .cfg file associated with that test, and opens it automatically when the session ID file is opened. You can change the .cfg file at any time without being concerned about the session ID file.

Removing a Script or a Player from a Test

To remove a script from a test:

1. Right-click on the script icon   for the appropriate script, then select **Remove Script**.

2. In the confirmation dialog box, click **Remove**.

To remove all scripts from a test:

1. Click **Actions>Remove All Scripts**.

2. In the confirmation dialog box, click **Remove**.

To remove a single Player from a test:

1. In the script test setup node, right-click on the Player to remove, then select **Remove Selected**.

2. In the confirmation dialog box, click **Remove**.

To remove all Players from a test:

1. Click **Actions>Remove All Players/Groups**.

2. In the confirmation dialog box, click **Remove**.

Assigning Scripts to Player Workstations

Once you've added scripts to your test session, you must assign scripts to Player workstations and set up Virtual User configurations. You can use:

the drag and drop method from the Players/Groups panel

the toolbar in the Visual Designer

the Machine Assignment tab in the Grid View

> **Note**: You cannot run multiple OFS scripts with different Forms Environment settings or different Connection Mode settings on the same player.

To assign players using the drag and drop method from the Players/Groups panel:

> **Note**: Use the Player Properties panel in the Visual Designer to set the optional Expert User options for WWW scripts.

1. Select a machine or group from the list in the **Players/Groups** window, then drag and drop it into the appropriate script test setup node. The Configure All Players/Groups dialog box appears.

2. Use the arrow keys in each field to set the following options:

   - **Starting VUs** - Number of Virtual Users to launch the script on the machine when the test begins.

   - **Ending VUs** - Number of Virtual Users at the end of the test.

   - **VU increment** - Number of virtual users to add at intervals throughout the test.

   > **Note**: If you add incremental virtual users, you must designate the time interval and the ending virtual users.

   - **Time interval** - Time interval at which incremental virtual users should be added to the test.

   - **Mode** - The test mode for the machine. You can select **thread** based or **process** based.

3.   To use these options for all players and groups in the script, click **Assign to all selected players and groups**.

4.   Click **Finish**.

5.   Click **File>Save** in the Conductor main menu.

To assign players using the toolbar in the Visual Designer:

> **Note**: Use the Player Properties panel in the Visual Designer to set the optional Expert User options for WWW scripts.

1.   Click the appropriate script icon in the Visual Designer.

2.   Click the **Assign Players/Groups** button [icon] in the toolbar. The Assign Players/Groups dialog box appears.

3.   Select the Player machine or group, then click **Next** to display the Configure All Players/Groups dialog box.

4.   Click the arrows in the fields to set the following options:

- **Starting VUs** - Number of Virtual Users to launch the script on the machine when the test begins.

- **Ending VUs** - Number of Virtual Users at the end of the test.

- **VU increment** - Number of virtual users to add at intervals throughout the test.

> **Note**: If you add incremental virtual users, you must designate the time interval and the ending virtual users.

- **Time interval** - Time interval at which incremental virtual users should be added to the test.

- **Mode** - The test mode for the machine. You can select **thread** based or **process** based.

5.   To use these options for all players and groups in the script, click **Assign to all selected players and groups**.

6.   Click **Finish**.

7.   Click **File>Save** in the Conductor main menu.

To assign players using the Grid View:

1.   In the **Machine Assignment** tab, click the down arrow in the **Player Name** field, then select a player from the list.

2.   (Optional - WWW only) In the **Middleware** field, click the **browse [...]** button to open the Expert User Options dialog box. If you enable the Expert User, select the Virtual User number to represent the Expert User.

3.   Click each of the following fields to set options for:

- **Starting VUs** - Type the number of Virtual Users to launch the script on the machine when the test begins.

- **Ending VUs** - Type the number of Virtual Users at the end of the test.

- **VU Increment** - Use the arrows in the field to set the number of virtual users to add at intervals throughout the test.

- **Timing Interval** - Click the **browse [...]** button in the field to open the Set Time Interval dialog box. Set the time interval at which incremental virtual users should be added to the test.

> **Note**: If you add incremental virtual users, you must designate the time interval and the ending virtual users.

- **Mode** - Use the arrows in the field to select the test mode for the machine. You can select **thread** based or **process** based.

4.   Click **File>Save** in the Conductor main menu.

Setting Script Properties

Enabling and Disabling ApplicationVantage Mode

Use the Script Properties panel to enable ApplicationVantage mode. This option is only available if Compuware ApplicationVantage is installed on the Player machine.

> **Note**: Setup the ApplicationVantage settings using the **Tools** menu, then selecting **Manage Players** to open the Manage Players/Groups dialog box.

To enable ApplicationVantage mode from the Script Properties panel:

Click the script icon for the appropriate script to display the Script Properties panel on the right-had side of the window. In the Script Properties panel, click the **ApplicationVantage Mode** field, then use the arrow key to enable or disable ApplicationVantage mode. Selecting:

True enables ApplicationVantage mode

False disables ApplicationVantage mode

Setting External Data Options

Use the external data options to add or remove attached files, specify a central datapool file, or specify local datapool files used by your script. You can setup external options using the Script Properties panel in the Visual Designer, or using the Grid View window.

To assign a Local Datapool file used by the script:

> **Note**: A local datapool file must reside in the directory \QALoad\Datapools on the Player workstation.

1.   Do one of the following:

- In the Visual Designer, click the script icon for the appropriate script to display the Script Properties panel on the right-hand side of the window, then click the **browse [...]** button in the **Local Datapools** field. The Script Properties dialog box appears with the Attached Files window displayed.

    OR

- In the Grid View window, select the **Script Assignment tab**, then click the **browse [...]** button in the **External Data** column. The Script Properties dialog box appears. Click **Local Datapools**.

2.   Click **Add**. The Choose a Local Datapool File dialog box appears.

3.   Select a file, and click **Open**.

> **Note**: If adding a shared datapool to a script using remote machines, it is possible to overwrite datapools. This will counteract the shared datapool automatic stripping. To avoid this, use the FTP Transer functionality in the Workbench.

4.   Repeat steps 2 and 3 to include additional datapool files.

5.   Click **OK**.

To assign attached files used by the script:

1.   Do one of the following:

   - Open the Script Properties panel for the appropriate script, then click the **browse [...]** button in the **Attached Files** field. The Script Properties dialog box appears with the Attached Files window displayed.

OR

   - In the Grid View window, select the **Script Assignment tab**, then click the **browse [...]** button in the **External Data** column. The Script Properties dialog box appears. Click **Attached Files**.

2.   Click **Add**. The Choose a file to attach dialog box appears.

3.   Select a file, and click **Open**.

To assign a Central Datapool file used by the script:

1.   Do one of the following:

   - Open the Script Properties panel for the appropriate script, then click the **browse [...]** button in the **Central Datapool** field. The Script Properties dialog box appears with the Central Datapool window displayed.

OR

   - In the Grid View window, select the **Script Assignment tab**, then click the **browse [...]** button in the **External Data** column. The Script Properties dialog box appears. Click **Central Datapool**.

2.   Click **Browse**. The Choose a Central Datapool File dialog box appears.

3.   Select a file, and click **Open**.

4.   (Optional) Select **Rewind** to rewind the records from this datapool at the end of a test. This enables you to reuse the records in a subsequent test of this session.

5.   (Optional) Select **Strip** to remove the datapool records from the test so that they cannot be used again.

Setting Middleware Options for Citrix and SAP

You can set the following custom options for Citrix and SAP middlewares:

Middleware Type	Option	Description
Citrix	Hide Graphical User Interface for Citrix Users	Runs Citrix in "windowless" mode.
SAPGUI	Hide Graphical User Interface for SAP Users	Runs SAP in "windowless" mode.

You can setup Citrix and SAP middleware options using the Script Properties panel in the Visual Designer or in the Grid View window.

To set the middleware options in the Visual Designer's Script Properties panel:

1. Click the appropriate Citrix or SAP script icon   . The Script Properties panel appears on the right-hand of the window.

2. Select the **Hide Citrix/SAP graphical user interface** field, and use the arrow key to select the desired middleware option. You can select:

   - **True** - to hide the graphical user interface
   - **False** - to display the graphical user interface

3. Click **OK**.

To set the middleware options in the Grid View window:

1. In the Script Assignment tab, select the appropriate Citrix or SAP script.

2. In the Middleware column, click the **browse [...]** button. The Middleware Options dialog box appears.

3. Select **Hide graphical user interface during replay** to run the script in windowless mode.

4. Click **OK**.

Setting Debugging Options for a Script

If you encountered errors while validating or testing a script, use *QA*Load's debugging options to monitor the Player(s) that generated errors while they are running or after the test.

You can watch a virtual user execute a script on a Player Workstation while it is running. To monitor selected virtual users at runtime, enable the Debug Trace option before you run your test. Each virtual user for which you enabled Debug Trace displays messages on its assigned Player workstation indicating which commands are being executed.

You can instruct the Conductor to generate and save details about the script execution of selected virtual users by enabling Logfile Generation before you run your test. This applies to Citrix, ODBC, Oracle, Oracle Forms Server, SAP, Winsock, or WWW only.

You can set the Debug Options using the Script Properties panel in the Visual Designer, or using the Grid View window.

To enable the Debug options:

1. Do one of the following:

   - In the Visual Designer, click the script icon   for the appropriate script to display the Script Properties panel on the right-hand side of the window, then click the **browse [...]** button in the **Debug Options** field. The Debug window of the Script Properties dialog box appears.

     OR

   - In the Grid View window, select the **Script Assignment tab**, then click the **browse [...]** button in the **Debug Options** column, for the appropriate script. The Debug window of the Script Properties dialog box appears.

2. On the Debug Options dialog box, you can choose the following options:

   - To enable the Debug Trace option: in the Debug Trace Virtual User Range area, choose which virtual users (if any) to monitor. You can choose None or All Virtual Users, or choose Virtual

> User(s) and then type the numbers assigned to the virtual users you want to monitor. You can monitor individual virtual users or ranges of virtual users.
>
> - To enable Logfile Generation: in the Logfile Generation Virtual User Range area, choose which virtual users (if any) to monitor. You can choose None or All Virtual Users, or choose Virtual User(s) and then type the numbers assigned to the virtual users you want to monitor. You can monitor individual virtual users or ranges of virtual users.

3. Click **OK** to save your changes.

4. From the Conductor's main menu, click **File>Save** to save your test session ID.

5. Run your test as usual.

Note: Some log files are generated automatically when you run a test in the Script Development Workbench or Player.

Setting Error Handling Options

You can configure the behavior of a virtual user when an error occurs during the load test so that the test aborts or continues executing. You can set the Error Handling Options using the Script Properties panel in the Visual Designer, or using the Grid View window.

To set the Error Handling options:

1. Do one of the following:

   - In the Visual Designer, click the script icon [icon] for the appropriate script to display the Script Properties panel on the right-hand side of the window, then click the **browse [...]** button in the **Error Handling** field. The Error Handling page of the Script Properties dialog box appears.

     OR

   - In the Grid View window, click the **Script Assignment** tab, then click the **browse [...]** button in the **Error Handling** field for the appropriate script . The Error Handling page of the Script Properties dialog box appears.

2. Select the option you want applied to the script. You can select:

   - Abort, stopping further execution of transactions. Use this option when errors will make the virtual user invalid for executing more transactions.

   - Continue executing and ignore the error. Select this option when errors are not critical to the performance of the load test

   - (WWW, SAPGUI, and Citrix scripts only) Restart the transaction from the beginning. When you select this option, you must type a number for the attempts made to restart the transaction in the **Maximum restart attempts** field.

     Note:The transaction count increases for each transaction that is restarted.

3. (Optional) Select **Apply Error Handling to all scripts in this session**. This ensures that all scripts in the session respond to errors the same way.

Setting Script Pacing

Script Pacing is the time interval between the start of a transaction and the beginning of the next transaction on each workstation running the script. For example: if a transaction is designed to duplicate the process of someone handling incoming telephone calls and those calls arrive at a rate of 40 per

hour/per person, set the pacing rate at 90 seconds. The default pacing value is one second, which enables the Conductor to control runaway virtual users.

> **Note**: *QA*Load randomly schedules transactions so that each transaction executes on an average according to this predetermined rate. When a transaction completes faster than its pacing rate, *QA*Load delays the execution of the next transaction for that workstation so that proper pacing is met. Since we do not normally time events according to this predetermined rate, *QA*Load randomly accelerates or delays the pacing on a workstation-by-workstation basis. However, on the average, *QA*Load provides pacing according to the value that you assign.

You can set the Pacing rate using the Script Properties panel in the Visual Designer, or using the Grid View window.

To set the Pacing rate:

1. Do one of the following:

   - In the Visual Designer, click the script icon for the appropriate script to display the Script Properties panel on the right-hand side of the window, then click the **browse [...]** button in the **Pacing** field. The Set Script Pacing dialog box appears.

     OR

   - In the Grid View window, click the **Script Assignment** tab, then click the **browse [...]** button in the **Pacing** field. The Set Script Pacing dialog box appears.

2. Use the arrows in each field to set the pacing rate. You can set Hours, Minutes, Seconds, and Milliseconds.

3. Click **OK**.

Setting the Service Level Threshold

Use the service level threshold to specify a response time that is used to compare against incoming response time data. The threshold appears as a horizontal line in the runtime Graphs view.

Service level thresholds are similar to thresholds that can be specified for real-time graphs in the Conductor, but are limited to transaction time and must be specified before a test runs.

Set the Service Level Threshold in the Visual Designer's Script Properties panel.

To set the service level threshold:

1. In the Visual Designer, click the script icon for the appropriate script to display open the Script Properties panel on the right-hand side of the window.

2. In the **Service Level Threshold** field, click the **browse [...]** button to open the Set Service Level Threshold dialog box.

3. Use the arrow keys in the Hours, Minutes, and Seconds fields to set the threshold.

4. Click **OK**.

Setting the Sleep Factor Percentage

Use the Sleep Factor % field to specify the percentage of any originally recorded delay to preserve in the script. *QA*Load records the actual delays between requests and inserts the DO_SLEEP command in the script to mimic those delays when the script is played back in a test. You can maintain the exact length of the recorded delays at playback, or shorten them by entering a smaller percentage of the originally recorded delay to play back. For example, if you recorded a delay of 10 seconds, then DO_SLEEP (10); is written to your script. Then, if a Sleep Factor of 50% is specified here, the Player sleeps for 5 seconds at that statement when the test is executed.

Valid values for Sleep Factor % are 0-1000%, and Random. Random causes the Player to sleep for a randomly selected duration between 0 and 100. A value of 100% causes the script to execute at exactly the same speed at which it was recorded; therefore, you can simulate the performance of faster users by specifying a lower Sleep Factor % value.

> **Hint**: Enter a value of zero during unit testing to eliminate the actual sleeps from the script. After you unit test the script, you can restore the original recorded delays by changing the Sleep Factor to a higher percentage.

You can set the sleep factor percentage using the Script Properties panel in the Visual Designer, or using the Grid View window.

To set the Sleep Factor percentage:

1. Do one of the following:

   - In the Visual Designer, click the script icon for the appropriate script to display the Script Properties panel on the right-hand side of the window.

   OR

   - In the Grid View window, click the **Script Assignment tab**, then click the **Sleep Factor %** field for the appropriate script.

2. In the **Sleep Factor %** field, do one of the following:

   - Click the arrow in the field to select **Random** or **100**

   OR

   - Type the number representing the percent of the originally recorded delay to keep in the script.

3. In the Conductor, click **File>Save**.

Setting Options for Large Amounts of Timing Data

Your load test probably includes a large number of checkpoints and virtual users in order to adequately test your system. When your test is running and your Conductor is collecting timing information from your Player machines, the sheer amount of data can take up more of your resources than you'd like to expend. Use *QA*Load's Timing Data Thinning option to thin the amount of timing data being transferred back to the Conductor during the test so that your test can run longer without stressing your resources.

You can set the Timing Options using the Script Properties panel in the Visual Designer, or using the Grid View window.

To open the Timing Options dialog box:

In the Visual Designer:

1. Click the script icon ![icon] for the appropriate script to display the Script Properties panel on the right-hand side of the window.

2. In the **Timing Options** field in the Script Properties panel, click the **browse [...]** button. The Timing page of the Script Properties dialog box appears.

In the Grid View window:

0. Click the **Script Assignment** tab.

1. In the **Timing Options** column, click the **browse [...]** button. The Timing page of the Script Properties dialog box appears.

To set timing options:

In the Timing Options area of the dialog box, select options for checkpoints and for custom counter data collection. For more information on these options, refer to Timing Options.

To thin timing data:

1. In the Timing Data Thinning area of the dialog box, choose one or both of the following:

   ▪ **Thin counter timing data by** to control the amount of counter timing data that is collected and saved in the timing file. Do not select this option if you want to collect all available timing data for counters.

   ▪ **Thin checkpoint timing data by** to control the amount of checkpoint timing data that is collected and saved in the timing file. Do not select this option if you want to collect all available timing data for checkpoints.

3. Do one of the following:

   ▪ Select **Script** to thin data by script. In the **Summary interval** field, type the number of seconds between each data collection.

   ▪ Select **Virtual User** to thin data by virtual user. In the **Summary interval** field, type the number of seconds between each data collection.

   Note: Thinning by script minimizes the amount of data collected.

4. The average is sent to the Conductor for inclusion in the timing file, rather than every value.

5. Click **OK**.

6. Save your changes to your test session ID file by choosing **File>Save** from the Conductor main menu.

Setting the Number of Transactions

Use the Transactions field to set the number of transactions that each Virtual User running the script should run. Once the workstation executes the number of transactions that you specify, script execution continues with the line following the End Transaction command rather than jumping to the beginning of the transaction loop.

You can set the number of transactions using the Script Properties panel in the Visual Designer, or using the Grid View window.

To set the number of transactions in the Visual Designer:

1. Click the script icon for the appropriate script. The Script Properties panel displays on the right-hand side of the window.

2. In the **Transactions** field, type the number of transactions that each Virtual User should run.

3. From the Conductor menus, click **File>Save**.

To set the number of transactions in the Grid View window:

1. Click the Script Assignment tab.

2. In the **Transactions** column for the appropriate script, use the arrow keys to select the number of transactions that each Virtual User should run.

3. From the Conductor menus, click **File>Save**.

Setting Player Properties

Specify Virtual User Configurations

Once you've selected the scripts and configured the transaction settings, you must assign Player machines and groups, and set the Virtual User configurations. You can set the Virtual User configurations using:

the toolbar in the script test setup node in the Visual Designer

the Player Properties panel in the Visual Designer

the Machine Assignment tab in the Grid View

the toolbar in the main Visual Designer window

To set the Virtual User configurations in the script test setup node:

1. Click the appropriate Player in the script test setup node in the Visual Designer.

2. Click the **Configure All** button in the script test setup node toolbar. The Configure All Players/Groups dialog box appears.

3. Click the arrows in the fields to set the following options:

- **Starting VUs** - Number of Virtual Users to launch the script on the machine when the test begins.

- **Ending VUs** - Number of Virtual Users at the end of the test.

- **VU increment** - Number of virtual users to add at intervals throughout the test.

■ **Note**: If you add incremental virtual users, you must designate the time interval and the ending virtual users.

- **Time interval** - Time interval at which incremental virtual users should be added to the test.

- **Mode** - The test mode for the machine. You can select **thread** based or **process** based.

5. To use these options for all players and groups in the script, click **Assign to all selected players and groups**.

6. Click **Finish**.

7. Click **File>Save** in the Conductor main menu.

To set the Virtual User configurations using the Player Properties panel of the Visual Designer:

1. Click the individual Player in the script test setup node in the Visual Designer. The Player Properties panel displays on the right-hand side of the window.

2. In the **[Starting VUs]** field, type the number of Virtual Users to launch the script on this machine when the test begins.

3. (Optional) In the **[VU Increment]** field, type the number of virtual users that should be added at intervals. When you fill in this field, you must also fill in the Time Interval and Ending VUs fields.

4. In the **Ending VUs** field, type the number of Virtual Users at the end of the test.

5. In the **Mode** field, use the arrow to select the test mode, process-based or thread-based, for this Player.

6. (Optional) In the **Time Interval** field, click the browse **[...]** button to display the Set Time Interval dialog box.

 Note: If you filled in the VU Increment field, you must fill in the Time Interval.

7. Use the arrows in the **Hours**, **Minutes**, and **Seconds** fields to set the time interval at which incremental Virtual Users should be added to the test, then click **OK**.

8. Click **File>Save** in the Conductor main menu.

To set the Virtual User configurations using the Grid View:

1. In the **Machine Assignment** tab, click the down arrow in the **Player Name** field, then select a player from the list.

2. (Optional - WWW only) In the **Middleware** field, click the **browse [...]** button to open the Expert User Options dialog box. If you enable the Expert User, select the Virtual User number to represent the Expert User.

3. Click each of the following fields to set options for:

   - **Starting VUs** - Type the number of Virtual Users to launch the script on the machine when the test begins.

   - **Ending VUs** - Type the number of Virtual Users at the end of the test.

   - **VU Increment** - Use the arrows in the field to set the number of virtual users to add at intervals throughout the test.

   - **Timing Interval** - Click the **browse [...]** button in the field to open the Set Time Interval dialog box. Set the time interval at which incremental virtual users should be added to the test.

    Note: If you add incremental virtual users, you must designate the time interval and the ending virtual users.

   - **Mode** - Use the arrows in the field to select the test mode for the machine. You can select **thread** based or **process** based.

4. Click **File>Save** in the Conductor main menu.

To set the Virtual User Configurations using the Visual Designer toolbar:

1. Select the appropriate script test setup node, then click **Review Virtual Users** in the Visual Designer toolbar.

2. Use the columns in the Review Virtual Configuration dialog box to set the Virtual User configurations for each Player listed.

Changing the Number of Virtual Users

Change the number of virtual users assigned to a script using the Visual Designer or on the Machine Assignment tab of the Grid View window.

To change the number of virtual users using the Visual Designer:

1. Do one of the following:

   - In the Visual Designer window, select a player machine in the appropriate script node. The Player Properties panel displays on the right-hand of the window. Type a new value in the **Starting VUs** column of the Player Properties panel.

   OR

   - Click **Review Virtual Users** in the Visual Designer toolbar, then type a new value in the **Starting VUs** column for the selected script. Click **OK**.

2. If you have assigned incremental virtual users, change the values in the **VU Increment** column and the **Ending VUs** column to determine how many virtual users to add at the interval specified in the **Time Interval** column.

3. Select **File>Save** to save your changes to the current session ID file, or **File>Save As** to save them to a new session ID file.

To change the number of virtual users using the Grid View:

1. In the **Machine Assignment** tab, type a new value in the **Starting VUs** column for the selected script.

2. If you have assigned incremental virtual users, change the values in the **VU Increment** column and the **Ending VUs** column to determine how many virtual users to add at the interval specified in the **Time Interval** column.

3. Select **File>Save** to save your changes to the current session ID file, or **File>Save As** to save them to a new session ID file.

Managing Players and Groups

Overview of Players and Groups

You can configure the various machines and agents that will participate in a load test from a single screen. Click **Tools>Manage Players** to display the Manage Players/Groups dialog box, where you can configure Player Machines, Player Groups, and ApplicationVantage settings information from a single screen.

You should use this option to update Player or Agent information whenever a Player or Agent is added to the test network, removed from the test network, or the network address of a Player or Agent has changed.

You can collect Player machines into logical groups using the Group Membership options in the dialog box.

Player Agents

Player machines execute the virtual users that perform the transactions recorded in your test scripts. You can view information on Player machines from either the Visual Designer or the Grid View window. If no Player machines are listed, you can retrieve information from Player machines on the local network, or you can add Player machines manually.

Adding Player Machines to a Test Session

Follow these instructions to add a Player workstation to your pool of available Players in a test's session ID file.

To add a new Player machine:

1. From the Conductor's main menu, click **Tools>Manage Players**. The **Manage Players/Groups** dialog box displays.

2. Click **File>New>Player**. A new page displays in the detail area of the dialog box, where you enter information and settings for the new Player.

To enter information and settings for the new Player:

Enter information for the new Player in the following areas of the dialog box.

In the **Player Information** area:

1. In the **Machine (hostname or IP)** field, type a name for the Player Machine.

2. In the **Communications [TCP] port** field, type the port number the Conductor should use to communicate (using TCP) with this machine during a test. The default is 3032.

3. Click **Verify** to check that the machine is active. The Player or Agent returns the operating system, processor type, amount of memory, and the maximum threads and processes available on the machine.

Note: If a Player does not respond, a message box appears indicating that the Player is not responding. If the Player is not responding, one of the following scenarios is likely:

- The host name and/or port number you entered may not be correct. Check your parameters and network connections, then try to send another request.

- The Player is not running. Start the Player and then try to send another request.

Expand the **Machine Settings** area (Optional):

4. Select desired options to ping the host before attempting connection to the player, generate IP spoof data, or override the default machine settings.

5. Close the Machine Settings.

Expand the **Group Membership** area (Optional):

6. In the Available Groups pane, select a group to which you want this Player Machine added, then click **Add.**

7. The selected groups are moved to the Member of Groups pane.

8. Close the Group Membership.

Note: You also can add a Player to a Group by dragging and dropping the Player from the list in the Players area to the appropriate Group in the Groups area, or by dragging and dropping a Group in there Groups area to a Player int he Players area.

In the **ApplicationVantage Settings** area:

Note: The fields on this tab are available only if ApplicationVantage is installed on the Player Machine and **ApplicationVantage Mode** is selected when you choose a script in the **Script Assignment** tab.

From the drop-down list in the **NIC Name** field, select the Network Interface Card (NIC) that is used by the machine, if necessary.

To save the Player machine:

Click **Save,** then click **OK**. The Player Machine appears in the **Manage Players and Groups** dialog box in the All Player Machines and Groups tree.

Editing a Player Machine

From the Conductor's main menu, click **Tools>Manage Players**. The Manage Player Machines and Groups dialog box displays. Use the following procedure to edit Player Machines.

To edit a Player machine:

Select an individual Player Machine in the Players list.

In the **Player Information** area:

1. In the Communications port field, type the port number the Conductor should use to communicate (using TCP) with this machine during a test. The default is 3032.

2. Click **Verify** to check that the machine is active. The Player or Agent returns the operating system, processor type, and amount of memory on the machine.

In the **Machine Settings** area:

1. Open the **Machine Settings** area.

2. Make any necessary selection or changes here, then close the Machine Settings.

In the **Group Membership** area (Optional):

1. Open the **Group Membership** area.

2. Add the Player to appropriate groups by selecting a group in the **Available Groups** pane and clicking **Add.**

3. **C**lose the Group Membership.

In the **ApplicationVantage Settings** area:

> **Note**: These fields are enabled only if Application Vantage is installed on the Player machine.

In the **NIC Name** field, select the Network Interface Card (NIC) that is used by the machine.

Save the edits to the Player machine:

Click **Save** to save your settings for this player machine, then click **OK**.

To delete a Player machine:

Select the Player machine in the **Players** panel, then click **Edit>Delete**.

Discovering and Verifying Player Machines

In Conductor, you can retrieve information from Player machines on the local network by doing the following:

To discover and information on Player machines:

1. Click **Tools>Manage Players**. The Manage Players/Groups dialog box displays.

2. Click **Actions>Discover Players**. *QA*Load Conductor queries the network for available Player workstations and adds the results under Player Machines in the **Players** area.

To verify a Player machine:

1. In the Players area of the screen, select the Player machine to verify.

2. In the **Communications [TCP] port** field in the Player Information area, click **Verify**. A confirmation dialog box appears when the verification is successful.

Viewing Information on Player Machines

In Conductor, you can retrieve information from Player machines on the local network by doing the one of following:

To view information on Player machines in the Visual Designer:

In the script node, select the individual Player you want to view. The information on the Player appears in the Properties panel on the right-hand side of the window.

To view information on the Player in the Grid View:

Click **View>Grid Window** to display the Grid View window, then click the **Machine Assignment** tab to display information on the Player.

Adding a Group

You can combine players into logical groups using the following procedure:

To create a group:

1. From Conductor's main menu, click **Tools>Manage Players**. The Manage Players/Groups dialog box displays.

2. Click **File>New> Group**. The Group Information dialog box appears.

3. In the **Name** field, type a name for the group.

4. In the **Description** field, type a description for the group.

To add Players to the group:

1. In the Available Players panel, select the player or players to add to the group.

2. Click **Add**. The Player is moved to the Member Players pane.

   ▣ **Note**: You can select more than one machine by holding down the **Ctrl** key and selecting each Player Machine to select. Select all the available Player Machines by clicking **Add All**.

3. Click **Save**, then click **OK**.

Editing a Group

Use the following procedure to edit a group.

To edit a group:

1. On Conductor's main menu, select **Tools>Manage Players**. The Manage Players/Groups dialog box displays.

2. Select a group in the **Groups** panel to display the Group Information window.

3. Use the fields in this dialog box to change the Group Name or Description.

4. To add a Player Machine to the group, use the procedures for adding a Player machine to a group.

To remove a Player from the group:

1. Select the Player machine in the **Member Players** panel, then click **Remove**.

2. Click **Save** to save your changes, then click **OK** to return to the main Conductor window.

Integration and Server Monitoring

Server and Performance Monitoring

*QA*Load integrates several mechanisms for merging load test response time data with server utilization data and performance metrics. Select the method that best suits your needs, or for which you are licensed (if applicable). Most methods produce data that is included in your load test timing results and processed in *QA*Load Analyze. The only exception is Compuware Application Vantage. Data captured from ApplicationVantage can be opened in ApplicationVantage, but not in *QA*Load.

This section briefly describes each method, and provides links to more detailed information about setting up a test that includes the appropriate method.

Remote Monitoring — allows you to monitor server utilization statistics from a remote machine without installing any software on the remote machine.

ServerVantage — integrates with your existing Compuware ServerVantage installation. You must be licensed for and have installed and configured the appropriate product in order to integrate with *QA*Load .

ApplicationVantage — collects test data that you can open in ApplicationVantage.

Vantage Analyzer - enables you to easily drill into specific problem transactions to determine the cause of bottlenecks in your production applications

See also:

Integration and Monitoring Requirements

Integration and Monitoring Requirements

Integration Requirements

ApplicationVantage

*QA*Load supports integration with Compuware ApplicationVantage 10.0 Service Pack 2, 10.1,l 10.2, and 11.0.

Integration with ApplicationVantage is supported on the Windows platform only.

ServerVantage

*QA*Load supports integration with Compuware ServerVantage (SVI Monitoring) 10.0, 10.1, 10.2, and 11.0.

*QA*Load supports integration with Compuware ServerVantage (Remote Monitoring) 10.1 Service Pack 1.5 only.

ClientVantage

QALoad supports integration with Compuware ClientVantage. QALoad is packaged with the current version of ClientVantage at time of release.

Vantage Analyzer

QALoad supports integration with Compuware Vantage Analyzer 10.1 Service Pack 1.

Monitoring Requirements

In addition to the integration requirements, your system may need to meet specific requirements to support remote monitoring.

JVM Requirements

Oracle Application Server (AS), JVM, SAP, WebLogic, WebSphere, and WebSphere MQ monitoring all require JVM installed on the Conductor machine.

For Oracle AS monitoring, if monitoring Oracle AS 10g, you must use Java Virtual Machine 1.5.

For SAP monitoring, you must use JVM 1.4.

For WebLogic monitoring version 7 and earlier versions, use JVM 1.3. For WebLogic version 8.1, use JVM 1.4. For WebLogic 9, use JVM 1.5. You may also use the JVM that is distributed with the WebLogic Application Server.

For WebSphere monitoring, you must use the JVM provided with the WebSphere client or server.

For WebSphere MQ monitoring, you must use JVM 1.4.

File Installation Requirements

Oracle Application Server (AS), SAP, WebLogic, WebSphere, WebSphere MQ, WMI, and Cold Fusion monitoring require the following files.

Oracle AS

For Oracle AS 10g, you must store copies of the dms.jar, xmlparserv2.jar, ons.jar, and optic.jar files from the monitored Oracle AS server on the Conductor machine.

SAP Monitoring

The SAP files listed below must be placed on the Conductor machine:

librfc32.dll

sapjco.jar

sapjcorfc

To obtain these files, install the **SAP Java Connector package (JCo)** on the Conductor machine. The JCo package is available from SAP. Add the location of the files, to the Path System Variable of the Conductor machine. For more information, refer to the *Requirements for SAP Remote Monitoring* topic in the ServerVantage Reconfigure Agent Online Help.

WebLogic Monitoring

Using the Conductor

The **weblogic.jar** file must be placed on the Conductor machine. Copy the jar file from the lib directory of the WebLogic application server to a separate directory in the Conductor machine. If you are monitoring WebLogic version 8.1, copy the **weblogic.jar** and **webservices.jar** files to the same directory. If you are monitoring WebLogic 9.x, copy the **weblogic.jar** and **wljmxclient.jar** file to the same directory. For more information, refer to Requirements for WebLogic Remote Monitoring in the ServerVantage Reconfigure Agent Online Help.

WebSphere Monitoring

Note the following requirements for WebSphere monitoring:

> The WebSphere client files must be installed on the Conductor machine. Installing the WebSphere Application Server Admin Server software on the Conductor machine provides the necessary client files. Note the directory path of the **WebSphere\AppServer\Java** files. For more information, refer to Requirements for WebSphere Remote Monitoring in the ServerVantage Reconfigure Agent Online Help.

> The Java Home for the monitoring task must be setup for compatible Java version; for example, **WebSphere\AppServer\Java**.

> If authentication is required and soap.client.props and ssl.client.props authentication files are installed in a custom directory, you must also place copies of the files in **WebSphere\AppServer\properties**.

WebSphere MQ Monitoring

The WebSphere client files listed below must be placed in a directory on the Conductor machine:

> com.ibm.mq.jar

> com.ibm.mq.pcf.jar

> connector.jar

The files may be obtained from the installation of the WebSphere Application Server Admin Server software on the Conductor machine. If the installation does not include the com.ibm.mq.pcf.jar file, obtain the file from the IBM Support Pac MS0B. See "http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg24000668&loc=en_US&cs=utf-8&lang=en".

For more information, refer to *Configuring WebSphere MQ for Remote Monitoring* in the ServerVantage Reconfigure Agent Online Help.

WMI Monitoring

WMI security must be enabled on the monitored server machine and the WMI service must be started. For more information, refer to Configuring WMI for Remote Monitoring in the ServerVantage Reconfigure Agent Online Help.

Cold Fusion Monitoring

Performance Monitoring must be enabled from the Cold Fusion Administrator Page – Debugging Settings of the monitored server machine. Cold Fusion is available under Windows Registry monitoring.

Rstat Monitoring

ServerVantage Agent must be installed with QALoad.

Host Verification for *QA*Load Monitoring

> Ensure host accessibility. Add an entry for the monitored machine to the system *hosts* file of the Conductor machine. Consult the network administrator for more information.

> Test host availability. Type the following command at the *Run* command: *ping* `<monitored machine name>`.

Remote Monitoring

Overview of Remote Monitoring

Remote Monitoring enables you to extract data from Windows Registry, JVM, Oracle Application Server (AS), SAP, SNMP, Compuware ServerVantage, Compuware Vantage Analyzer, WebLogic, WebSphere, WebSphere MQ, Rstat, and WMI counters on the servers under stress without installing any software on the servers.

> **Note**: Select counters for monitor types in the application.

To use Remote Monitoring:

- You must have login access to the machines you want to monitor.
- You must select the servers and counters to monitor on the machines you identify using the monitoring options on Conductor's Manage Monitoring Tasks window.
- To collect SNMP counters, SNMP must be enabled on the Remote Monitor machine. Refer to your operating system help for information about enabling SNMP.
- To collect Windows registry counters, you must have a valid sign-on for the servers under test.
- For requirements for Oracle AS, SAP, WebLogic, WebSphere, WebSphere MQ, and WMI, see Integration and Monitoring Requirements.

*QA*Load uses the default ports 7790 and 7788 when it communicates with the ServerVantage agent and client. You can override the default ports if your ServerVantage installation requires it.

While your test is running, *QA*Load collects the appropriate counter data and writes it to your timing file where you can view it in Analyze after the test. What counters are available?

You can simplify the configuration process by creating or applying pre-defined monitoring templates. A monitoring template is a predefined group of counters not associated with a specific machine.

To set up Remote Monitoring, see Creating a New Monitoring Task.

Monitoring Counters

About Counters and Instances

You use counters and, in some cases, specific instances of counters when you monitor servers.

Counters

Counters are the numeric data values that are collected when monitoring servers. Counters exist for components such as processor, memory, processes, hard disk, and cache, with a set of counters that measure statistical information. For Windows, a large number of performance counters are provided by the operating system registry and Windows server applications. Registry counters can monitor external components of the environment such as databases, applications, and printers.

Many of the counters that are collected are points in time data values, such as Process\thread count. Some counters are cumulative, such as server logon errors, and some are averages, such as the page faults per second in Job Object Details.

In addition to the numeric value counters, a set of extended data counters is provided for a number of key performance indicators. These extended data counters can provide intelligent data points that have associated textual data for the numeric value. For example, the extended CPU usage counter's intelligent datapoint shows the top 10 processes consuming CPU at that time.

Instances

When you select a counter to monitor, the available instances, or occurrences, for that counter appear. Counters can have several instances or no instances. For example, if a system has multiple processors, then the Processor counter has multiple instances. For counters with multiple instances, a list of the available instances for that counter is presented. Many counters also have an instance called _Total, which is an aggregate of the individual instances.

Counters for an object, such as processor, have instances that are numbered, beginning with 0 (zero). A machine with a single processor has an instance of _Total and 0. A dual-processor machine has instances of _Total, 0, and 1. Other instances are based on what is currently running on the server, and the instance list displays these for each process name or service name that is active.

Some instances represent the most recent value for the resource, for example, Processes. This is the number of processes in the computer at the time of data collection. Other instances are average values between the last two measurements.

Windows NT Registry Server Counters

*QA*Load supports the following MS Windows NT Server counter categories:

Counter Category	Description
Active Server Pages	This object type handles the Active Server Pages device on your system.
Browser	This object type displays Browser Statistics.
Cache	The Cache object type manages memory for rapid access to files. Files on Windows NT are cached in main memory in units of pages. Main memory not being used in the working sets of processes is available to the Cache for this purpose. The Cache preserves file pages in memory for as long as possible to permit access to the data through the file system without accessing the disk.
Context Index	This object type handles the Content Index.
Context Index Filter	This object type handles the Content Index Filter.
ICMP	The ICMP object type includes the counters that describe the rates that ICMP Messages are received and sent by a certain entity using the ICMP protocol. It also describes various error counts for the ICMP protocol.
IP	This object type includes those counters that describe the rates that IP datagrams are received and sent by a certain computer using the IP protocol. It also describes various error counts for the IP protocol.
LogicalDisk	A LogicalDisk object type is a partition on a hard or fixed disk drive and assigned a drive letter, such as C. Disks can be partitioned into distinct sections where they can store file, program, and page data. The disk is read to retrieve these items and written to record changes to them.
Memory	The Memory object type includes those counters that describe the

	behavior of both real and virtual memory on the computer. Real memory is allocated in units of pages. Virtual memory can exceed real memory in size, causing page traffic as virtual pages are moved between disk and real memory.
Network Interface	The Network Interface Object Type includes those counters that describe the rates that bytes and packets are received and sent over a Network TCP/IP connection. It also describes various error counts for the same connection.
Objects	The Objects object type is a meta-object that contains information about the objects in existence on the computer. This information can be used to detect the unnecessary consumption of computer resources. Each object requires memory to store basic information about the object.
Paging File	This object displays information about the system's Page File(s).
PhysicalDisk	A PhysicalDisk object type is a hard or fixed disk drive. It contains 1 or more logical partitions. Disks are used to store file, program, and paging data. The disk is read to retrieve these items and written to record changes to them.
Process	The Process object type is created when a program is run. All the threads in a process share the same address space and have access to the same data.
Process Address Space	Process Address Space object type displays details about the virtual memory usage and allocation of the selected process.
Processor	The Processor object type includes as instances all processors on the computer. A processor is the part in the computer that performs arithmetic and logical computations, and initiates operations on peripherals. It executes (such as runs) programs on the computer.
Redirector	The Redirector is the object that manages network connections to other computers that originate from your own computer.
Server	The Server object type is the process that interfaces the services from the local computer to the network services.
Server Work Queues	The Server Work Queues object type handles explain text performance data.
SMTP Server	This object type handles the counters specific to the SMTP Server.
System	This object type includes those counters that apply to all processors on the computer collectively. These counters represent the activity of all processors on the computer.
TCP	The TCP object type includes the counters that describe the rates that TCP Segments are received and sent by a certain entity using the TCP protocol. In addition, it describes the number of TCP connections in each possible TCP connection state.
Telephony	This object type handles the Telephony System.

Thread	The Thread object type is the basic object that executes instructions in a processor. Every running process has at least one thread.
UDP	The UDP object type includes the counters that describe the rates that UDP datagrams are received and sent by a certain entity using the UDP protocol. It also describes various error counts for the UDP protocol.

For information on the registry counters refer to the documentation or developer network for that product or the developer kit provided with the product. For Microsoft products, refer to http://msdn.microsoft.com/library/default.aspx.

Windows Server Registry Counters

Remote Monitoring Agents can monitor the same Windows registry counters as PERFMON, the performance monitoring application available with the Windows operating system. To retrieve Windows Registry Counters, you must have access, via a user name and password, to the remote machine.

*QA*Load supports the following MS Windows counter categories:

Counter Category	Description
ACS/RSVP Service	RSVP or ACS service performance counters.
Active Server Pages	This object type handles the Active Server Pages device on your system.
Browser	The Browser performance object consists of counters that measure the rates of announcements, enumerations, and other Browser transmissions.
Cache	The Cache performance object consists of counters that monitor the file system cache, an area of physical memory that stores recently used data as long as possible to permit access to the data without reading from the disk. Because applications typically use the cache, the cache is monitored as an indicator of application I/O operations. When memory is plentiful, the cache can grow, but when memory is scarce, the cache can become too small to be effective.
IAS Accounting Clients	IAS Accounting Clients
IAS Accounting Server	IAS Accounting Server
IAS Authentication Clients	IAS Authentication Clients
IAS Authentication Server	IAS Authentication Server
ICMP	The ICMP performance object consists of counters that measure the rates at which messages are sent and received by using ICMP protocols. It also includes counters that monitor ICMP protocol errors.
IP	The IP performance object consists of counters that measure the rates at which IP datagrams are sent and received by using IP protocols. It also includes counters that monitor IP protocol errors.
LogicalDisk	The Logical Disk performance object consists of counters that monitor logical partitions of a hard or fixed disk drives. Performance Monitor

	identifies logical disks by their a drive letter, such as C.
Memory	The Memory performance object consists of counters that describe the behavior of physical and virtual memory on the computer. Physical memory is the amount of random access memory on the computer.. Virtual memory consists of the space in physical memory and on disk. Many of the memory counters monitor paging, which is the movement of pages of code and data between disk and physical memory. Excessive paging, a symptom of a memory shortage, can cause delays which interfere with all system processes.
NBT Connection	The NBT Connection performance object consists of counters that measure the rates at which bytes are sent and received over the NBT connection between the local computer and a remote computer. The connection is identified by the name of the remote computer.
Network Interface	The Network Interface performance object consists of counters that measure the rates at which bytes and packets are sent and received over a TCP/IP network connection. It includes counters that monitor connection errors.
Objects	The Object performance object consists of counters that monitor logical objects in the system, such as processes, threads, mutexes, and semaphores. This information can be used to detect the unnecessary consumption of computer resources. Each object requires memory to store basic information about the object.
Paging File	The Paging File performance object consists of counters that monitor the paging file(s) on the computer. The paging file is a reserved space on disk that backs up committed physical memory on the computer.
PhysicalDisk	The Physical Disk performance object consists of counters that monitor hard or fixed disk drive on a computer. Disks are used to store file, program, and paging data and are read to retrieve these items, and written to record changes to them. The values of physical disk counters are sums of the values of the logical disks (or partitions) into which they are divided.
Print Queue	Displays performance statistics about a Print Queue.
Process	The Process performance object consists of counters that monitor running application program and system processes. All the threads in a process share the same address space and have access to the same data.
Process Address Space	The Process Address Space performance object consists of counters that monitor memory allocation and use for a selected process.
Processor	The Processor performance object consists of counters that measure aspects of processor activity The processor is the part of the computer that performs arithmetic and logical computations, initiates operations on peripherals, and runs the threads of processes. A computer can have multiple processors. The processor object represents each processor as an instance of the object.
Redirector	The Redirector performance object consists of counter that monitor

	network connections originating at the local computer.
Server	The Server performance object consists of counters that measure communication between the local computer and the network.
Server Work Queues	The Server Work Queues performance object consists of counters that monitor the length of the queues and objects in the queues.
SMTP NTFS Store Driver	This object represents global counters for the Exchange NTFS Store driver.
SMTP Server	The counters specific to the SMTP Server.
System	The System performance object consists of counters that apply to more than one instance of a component processors on the computer.
TCP	The TCP performance object consists of counters that measure the rates at which TCP Segments are sent and received by using the TCP protocol. It includes counters that monitor the number of TCP connections in each TCP connection state.
Telephony	The Telephony System.
Thread	The Thread performance object consists of counters that measure aspects of thread behavior. A thread is the basic object that executes instructions on a processor. All running processes have at least one thread.
UCP	The UCP performance object consists of counters that measure the rates at which UCP datagrams are sent and received by using the UCP protocol. It includes counters that monitor UCP protocol errors.

For information on the registry counters refer to the documentation or developer network for that product or the developer kit provided with the product. For Microsoft products, refer to http://msdn.microsoft.com/library/default.aspx.

SAP R/3 Remote Extended Counters

The following extended SAP R/3 remote counters are provided. These counters extend the monitoring of your SAP R/3 system:

Active Servers	Page/Roll Area
Active Users	Page/Roll Area Max
Alerts	Process Monitoring
Buffer Statistics	Spool Queue
CCMS Monitoring	System Log Entries
Connection Test (SM59)	Top CPU Utilization
CPU Consumption	Top Load
Itemized Active Users	User Function Call
Itemized Job Status	Workload Statistic
Itemized Spool Queue	Work Processes

Job Status

Memory Usage

Number of Dumps

SNMP Counters

SNMP Counters

SNMP Remote Monitoring uses the SNMP service to provide network and system counters. SNMP counters can be retrieved from any machine that is running an SNMP server. QALoad uses the default SNMP port (161) and default Community (public). If necessary, you can enter a different port and community in the Configure Monitor Dialog screen when you create or edit an SNMP monitoring task and when you create or edit an SNMP monitoring template. Although SNMP does not require a user name and password, the SNMP agent must be configured to allow read-only access from the Conductor machine.

In addition to the standard SNMP counters supported by QALoad Remote Monitoring, you can create your own custom Object Identifier (OID) file with counters you define. See Customizing SNMP Counter Discovery for more information.

Standard SNMP Counters

Standard SNMP counters that are supported by *QA*Load Remote Monitoring are categorized below.

ICMP

icmpInMsgs/sec: the rate at which ICMP messages are received
icmpInErrors: the number of ICMP messages received having ICMP errors
IcmpInDestUnreachs: the number of ICMP Destination Unreachable messages received
IcmpInTimeExcds: the number of ICMP Time Exceeded messages received
IcmpInParmProbs: the number of ICMP Parameter Problem messages received

IcmpInSrcQuenchs: the number of ICMP Source Quench messages received
icmpInRedirects/sec: the rate at which ICMP Redirect messages are received
icmpInEchos/sec: the rate at which ICMP Echo messages are received
icmpInEchoReps/sec: the rate at which ICMP Echo Reply messages are received
icmpInTimestamps/sec: the rate at which ICMP Timestamp messages are received
icmpInTimestampReps/sec: the rate at which ICMP Timestamp Reply messages are received
icmpInAddrMasks: the number of ICMP Address Mask Request messages received
icmpInAddrMaskReps: the number of ICMP Address Mask Reply messages received
icmpOutMsgs/sec: the rate at which ICMP messages are sent
icmpOutMsgs/sec: the number of ICMP messages not sent due to ICMP errors
icmpOutDestUnreachs: the number of ICMP Destination Unreachable messages sent
icmpOutTimeExcds: the number of ICMP Time Exceeded messages sent
icmpOutParmProbs: the number of ICMP Parameter Problem messages sent
icmpOutSrcQuenchs: the number of ICMP Source Quench messages sent
icmpOutRedirects/sec: the number of ICMP Redirect messages sent
icmpOutEchos/sec: the number of ICMP Echo messages sent
icmpOutEchoReps/sec: the number of ICMP Echo Reply messages sent
icmpOutTimestamps/sec: the number of ICMP Timestamp messages sent
icmpOutTimestampReps/sec: the number of ICMP Timestamp Reply messages sent
icmpOutAddrMasks: the number of ICMP Address Mask Request messages sent
icmpOutAddrMaskReps: the number of ICMP Address Mask Reply messages sent

IP

ipForwarding: the indication of whether this entity is acting as an IP router in respect to the forwarding of datagrams received by, but not addressed to, this entity.

ipDefaultTTL: the default value inserted into the Time-To-Live field of the IP header of datagrams originated at this entity, whenever a TTL value is not supplied by the transport layer protocol.

ipInReceives/sec: the rate of input datagrams received from interfaces, including those received in error.

ipInHdrErrors: the number of input datagrams discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, and so on.

ipInAddrErrors: the number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity.

ipForwDatagrams/sec: the rate of input datagrams for which this entity was not their final IP destination, as a result of which an attempt was made to find a route to forward them to that final destination.

ipInUnknownProtos: the number of locally-addressed datagrams receive successfully but discarded because of an unknown or unsupported protocol.

ipInDiscards: the number of input IP datagrams for which no problems were encountered to prevent their continued processing, but which were discarded (for example, for lack of buffer space).

ipInDelivers/sec: the rate of input datagrams successfully delivered to IP user-protocols (including ICMP).

ipOutRequests: the number of IP datagrams which local IP user-protocols (including ICMP) supplied to IP in requests for transmission.

ipOutDiscards: the number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but which were discarded (for example, for lack of buffer space).

ipOutNoRoutes: the number of IP datagrams discarded because no route could be found to transmit them to their destination.

ipReasmTimeout: the maximum number of seconds which received fragments are held while they are awaiting reassembling at this entity.

ipReasmReqds: the number of IP fragments received which needed to be reassembled at this entity.

ipReasmOKs: the number of IP datagrams successfully re-assembled.

ipReasmFails: the number of failures detected by the IP re-assembly algorithm (for whatever reason: timed out, errors, etc).

ipFragOKs: the number of IP datagrams that have been successfully fragmented at this entity.

ipFragFails: the number of IP datagrams that have been discarded because they needed to be fragmented at this entity but could not be, for example, because their Don't Fragment flag was set.

ipFragCreates/sec: the rate of IP datagram fragments that have been generated as a result of fragmentation at this entity.

ipRoutingDiscards: the number of routing entries which were chosen to be discarded even though they are valid.

SNMP

snmpInPkts/sec: the rate of messages delivered to the SNMP entity from the transport service.

snmpOutPkts/sec: the rate at which SNMP Messages were passed from the SNMP protocol entity to the transport service.

snmpInBadVersions: the number of SNMP messages which were delivered to the SNMP entity and were for an unsupported SNMP version.

snmpInBadCommunityNames: the number of SNMP messages delivered to the SNMP entity which used a SNMP community name not known to said entity.

snmpInBadCommunityUses: the number of SNMP messages delivered to the SNMP entity which represented an SNMP operation which was not allowed by the SNMP community named in the message.

snmpInASNParseErrs: the number of ASN.1 or BER errors encountered by the SNMP entity when decoding received SNMP messages.

snmpInTooBigs: the number of SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is tooBig.

snmpInNoSuchNames: the number of SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is noSuchName.

snmpInBadValues: the number of SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is badValue.

snmpInReadOnlys: the number valid SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is readOnly.
snmpInGenErrs: the number of SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is genErr.
snmpInTotalReqVars/sec: the rate of MIB objects which have been retrieved successfully by the SNMP protocol entity as the result of receiving valid SNMP Get-Request and Get-Next PDUs.
snmpInTotalSetVars/sec: the rate of MIB objects which have been altered successfully by the SNMP protocol entity as the result of receiving valid SNMP Set-Request PDUs.
snmpInGetRequests/sec: the rate of SNMP Get-Request PDUs which have been accepted and processed by the SNMP protocol entity.
snmpInGetNexts/sec: the rate of SNMP Get-Next PDUs which have been accepted and processed by the SNMP protocol entity.
snmpInSetRequests/sec: the rate of SNMP Get-Response PDUs which have been accepted and processed by the SNMP protocol entity.
snmpInGetResponses/sec: the rate of SNMP Set-Request PDUs which have been accepted and processed by the SNMP protocol entity.
snmpInTraps: the number of SNMP Trap PDUs which have been accepted and processed by the SNMP protocol entity.
snmpOutTooBigs: the number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status field is tooBig.
snmpOutNoSuchNames: the number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status is noSuchName.
snmpOutBadValues: the number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status field is badValue.
snmpOutGenErrs: the number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status field is genErr.
snmpOutGetRequests/sec: the rate of SNMP Get-Request PDUs which have been generated by the SNMP protocol entity.
snmpOutGetNexts/sec: the rate of SNMP Get-Next PDUs which have been generated by the SNMP protocol entity.
snmpOutSetRequests/sec: the rate of SNMP Set-Request PDUs which have been generated by the SNMP protocol entity.
snmpOutGetResponses/sec: the rate of SNMP Get-Response PDUs which have been generated by the SNMP protocol entity.
snmpOutTraps: the number of SNMP Trap PDUs which have been generated by the SNMP protocol entity.
snmpOutTraps: indicates whether the SNMP entity is permitted to generate authenticationFailure traps.

TCP

tcpRtoAlgorithm: the algorithm used to determine the timeout value used for retransmitting unacknowledged octets.
tcpRtoMin: the minimum value permitted by a TCP implementation for the retransmission timeout.
tcpRtoMax: the maximum value permitted by a TCP implementation for the retransmission timeout.
tcpMaxConn: the limit on the total number of TCP connections the entity can support.
tcpActiveOpens: the number of times TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state.
tcpAttemptFails: the number of times TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state.
tcpEstabResets: the number of times TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state.
tcpCurrEstab: the number of TCP connections for which the current state is either ESTABLISHED or CLOSE-WAIT.
tcpInSegs/sec: the rate at which segments are received, including those received in error.
tcpOutSegs/sec: the rate at which segments are sent, including those on current connections but excluding those containing only retransmitted octets.
tcpRetransSegs/sec: the rate at which segments are retransmitted.

tcpInErrs/sec: the rate at which segments are received in error.
tcpOutRsts/sec: the rate at which segments containing the RST flag are sent.
tcpPassiveOpens: the total number of times TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state.

UDP

udpInDatagrams/sec: the rate of UDP datagrams being delivered to UDP users.
udpNoPorts/sec: the rate of received UDP datagrams for which there was no application at the destination port.
udpInErrors: the number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.
udpOutDatagrams/sec: the rate at which UDP datagrams are sent.

Solaris: Sun System

Collisions/sec: the rate of output collisions.
CpuUser%: the percentage of non-idle processor time that is spent in user mode.
CpuNice%: the percentage of non-idle processor time that is spent in nice mode.
CpuSys%: the percentage of non-idle processor time that is spent in system mode.
CpuIdle%: the percentage of idle processor time.
IfInPackets/sec: the rate of input packets.
IfOutPackets/sec: the rate of output packets.
IfInErrors: the total number of input errors.
IfOutErrors: the total number of output errors.
Interrupts/sec: the rate of system interrupts.
PagesIn KBytes/sec: the rate of pages read in from disk.
PagesOut KBytes/sec: the rate of pages written to disk.
SwapIn KBytes/sec: the rate at which pages are being swapped in.
SwapOut KBytes/sec: the rate at which pages are being swapped out.

HP-UX: HP System

AvgJobs1: the average number of jobs in the last minute * 100.
AvgJobs5: the average number of jobs in the last 5 minutes * 100.
AvgJobs15: the average number of jobs in the last 15 minutes * 100.
CpuUser%: the percentage of non-idle processor time that is spent in user mode.
CpuNice%: the percentage of non-idle processor time that is spent in nice mode.
CpuSys%: the percentage of non-idle processor time that is spent in system mode.
CpuIdle%: the percentage of idle processor time.
FreeMemory KBytes: the amount of idle memory.
FreeSwap KBytes: the amount of free swap space on the system.
MaxProc: the maximum number of processes allowed.
MaxUserMem KBytes: the amount of maximum user memory on the system.
PhysMemory KBytes: the amount of physical memory on the system.
Users: the number of users logged on to the machine.

LINUX Memory

AvailableSwap KBytes: the available swap on the system.
Buffered KBytes: the amount of memory used as buffers.
Cached KBytes: the amount of memory cached.
FreeMemory KBytes: the amount of idle memory.
Shared KBytes: the amount of memory shared.
TotalMemory KBytes: the total amount of memory on the system.
TotalSwap KBytes: the total swap size for the system.

LINUX System

CpuUser%: the percentage of non-idle processor time that is spent in user mode.
CpuNice%: the percentage of non-idle processor time that is spent in nice mode.
CpuSys%: the percentage of non-idle processor time that is spent in system mode.
CpuIdle%: the percentage of idle processor time.

Windows HTTP Server

httpTotalFilesSent: the total number of files sent by this HTTP server.
httpTotalFilesReceived: the total number of files received by this HTTP server.
httpCurrentAnonymousUsers: the number of anonymous users currently connected to this HTTP server.
httpCurrentNonAnonymousUsers: the number of non-anonymous users currently connected to this HTTP server.
httpTotalAnonymousUsers: the total number of anonymous users that have ever connected to this HTTP server.
httpTotalNonAnonymousUsers: the total number of non-anonymous users that have ever connected to this HTTP server.
httpMaximumAnonymousUsers: the maximum number of anonymous users simultaneously connected to this HTTP server.
httpMaximumNonAnonymousUsers: the maximum number of non-anonymous users simultaneously connected to this HTTP server.
httpCurrentConnections: the current number of connections to the HTTP server.
httpMaximumConnections: the maximum number of simultaneous connections to the HTTP server.
httpConnectionAttempts: the total number of connection attempts to the HTTP server.
httpLogonAttempts: the total number of logon attempts to the HTTP server.
httpTotalOptions: the total number of requests made to this HTTP server using the OPTIONS method.
httpTotalGets: the total number of requests made to this HTTP server using the GET method.
httpTotalPosts: the total number of requests made to this HTTP server using the POST method.
httpTotalHeads: the total number of requests made to this HTTP server using the HEAD method.
httpTotalPuts: the total number of requests made to this HTTP server using the PUT method.
httpTotalDeletes: the total number of requests made to this HTTP server using the DELETE method.
httpTotalTraces: the total number of requests made to this HTTP server using the TRACE method.
httpTotalMove: the total number of requests made to this HTTP server using the MOVE method.
httpTotalCopy: the total number of requests made to this HTTP server using the COPY method.
httpTotalMkcol: the total number of requests made to this HTTP server using the MKCOL method.
httpTotalPropfind: the total number of requests made to this HTTP server using the PROPFIND method.
httpTotalProppatch: the total number of requests made to this HTTP server using the PROPPATCH method.
httpTotalSearch: the total number of requests made to this HTTP server using the MS-SEARCH method.
httpTotalLock: the total number of requests made to this HTTP server using the LOCK method.
httpTotalUnlock: the total number of requests made to this HTTP server using the UNLOCK method.
httpTotalOthers: the total number of requests made to this HTTP server not using the OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, MOVE, MKCOL, PROPFIND, PROPPATCH, MS-SEARCH, LOCK or UNLOCK methods.
httpCurrentCGIRequests: the number of Common Gateway Interface requests currently being serviced by this HTTP server.
httpCurrentBGIRequests: the number of Binary Gateway Interface requests currently being serviced by this HTTP server.
httpTotalCGIRequests: the total number of Common Gateway Interface requests made to this HTTP server.
httpTotalBGIRequests: the total number Binary Gateway Interface requests made to this HTTP server.
httpMaximumCGIRequests: the maximum number of Common Gateway Interface requests simultaneously processed by this HTTP server.
httpMaximumBGIRequests: the maximum number of Binary Gateway Interface requests simultaneously processed by this HTTP server.

httpCurrentBlockedRequests: the current number of requests being temporarily blocked by this HTTP server.
httpTotalBlockedRequests: the total number of requests that have been temporarily blocked by this HTTP server.
httpTotalRejectedRequests: the total number of requests that have been rejected by this HTTP server.

Windows FTP Server

ftpTotalFilesSent: the total number of files sent by this FTP server.
ftpTotalFilesReceived: the total number of files received by this FTP server.
ftpCurrentAnonymousUsers: the number of anonymous users currently connected to this FTP server.
ftpCurrentNonAnonymousUsers: the number of non-anonymous users currently connected to this FTP server.
ftpTotalAnonymousUsers: the total number of anonymous users that have ever connected to this FTP server.
ftpTotalNonAnonymousUsers: the total number of non-anonymous users that have ever connected to this FTP server.
ftpMaximumAnonymousUsers: the maximum number of anonymous users simultaneously connected to this FTP server.
ftpMaximumNonAnonymousUsers: the maximum number of non-anonymous users simultaneously connected to this FTP server.
ftpCurrentConnections: the current number of connections to the FTP server.
ftpMaximumConnections: the maximum number of simultaneous connections to the FTP server.
ftpConnectionAttempts: the total number of connection attempts to the FTP server.
ftpLogonAttempts: the total number of logon attempts to the FTP server.

Customizing SNMP Counter Discovery

*QA*Load currently has a standard list of Object Identifiers (OID) that it searches when discovering SNMP counters. You can create and import a list of additional OIDs in an XML file. The XML file, called the Custom OID File, introduces additional SNMP counters that you want to include during the counter discovery phase when you create or edit tasks and templates.

Note: You can include multiple categories and multiple counters in the custom OID file.

Once you create the custom OID file, you select it using the Custom OID File field in the Configure Monitor Dialog screen of the monitoring wizards. Since there is a one to one relationship between monitoring tasks and the custom file, multiple files can exist. These reside in a common, default location: `QALoad\Monitoring\SNMP`. The browse button for the Custom OID File field defaults to this location. This directory also contains two files to assist you in developing your Custom OID File:

> OID structure.xml - a template you can use for creating your custom supplemental file. This contains the basic XML structure you need to create your own Custom OID file. The Table of Custom OID File Elements describes and gives the rules for each of the XML elements in the structure.

> OID example.xml - a custom OID XML file structure. This is a sample Custom OID File.

Custom OID Template File

Use the following template, located in QALoad\Monitoring\SNMP\OID structure.xml, to create your custom OID file:

Table of Custom OID File Elements

The following table shows each element in the XML file structure, and describes its purpose and the rules that apply.

Tag	Description	Rules
<?xml version="1.0" ?>	XML file header indicating files structural version.	Must be first element of file. Only one per file.
<OIDCustom>	Begin bracket for entire OID custom information. Used to both identify content type and begin/end of file.	Only one per file. Must immediately follow XML version header.
<CategoryList>	Begin bracket for list of categories defined in this file. Used to help group all the contained categories.	Only one per file. Must immediately follow <OIDCustom> tag.
<Category>	Begin bracket for an individual category within the category list. Used to help group individual categories.	One required per category group. First instance must immediately follow <CategoryList> tag. Subsequent instances immediately follow </Category> end tag of previous group.
<CategoryName>?</CategoryName>	The display name for the category group. The "?" represents where the user defines the category name.	One required per category. Must immediately follow <Category> tag.
<Counter>	Begin bracket for individual counter information contained in a category. Multiple counters can exist per category.	One required per counter. First instance must immediately follow </CategoryName> tag. Subsequent instances immediately follow </Counter> tag.
<CounterName>?</CounterName>	The display name for the counter. The "?" represents where the user defines the counter name.	One required per counter. Only one allowed per counter.
<OID>?</OID>	The OID for the counter. The OID must start with a period "." (see example below). The "?" represents where the user defines the OID.	One required per counter. Only one allowed per counter.

<Units>?</Units>	The optional calculation units for the counter. The "?" represents where the user defines the calculation type which is one of the following: **"/sec"** – for counters that must take into account the time between samplings. **"Cpu%"** – for counters that must take into account the number of processors. A counter for which this applies is CpuIlde%. **"Kbits/sec"** – for converting counters that natively report in Kbit/sec to Kbytes/sec.	Optionally, one per counter. Only one allowed per counter.
<Description>?</Description>	The optional description for the counter. The "?" represents where the user defines the OID.	Optionally, one per counter. Only one allowed per counter.
</Counter>	The end bracket for an individual counter information set. Used to denote the end of individual counters attribute information.	One required per counter.
</Category>	The end bracket for an individual category, used to denote the end of a category and all of its associated counters.	One required per category.
</CategoryList>	End bracket for list of categories.	Only one per file. Must immediately precede <OIDCustom> tag.
</OIDCustom>	End bracket for entire OID custom information. Used to identify the end of file.	Only one per file. Must be last file element.

Custom OID Sample File

The sample Custom OID File shown here is located in QALoad\Monitoring\SNMP\OID example.xml:

```xml
<?xml version="1.0" ?>
- <OIDCustom>
  - <CategoryList>
    - <Category>
        <CategoryName>ip</CategoryName>
      - <Counter>
          <CounterName>ipDefaultTTL</CounterName>
          <OID>.1.3.6.1.2.1.4.2.0</OID>
          <Description>The default value inserted into the Time-To-Live field of the IP header.</Description>
        </Counter>
      - <Counter>
          <CounterName>ipRoutingDiscards</CounterName>
          <OID>.1.3.6.1.2.1.4.23.0</OID>
          <Description>The number of routing entries which were chosen to be discarded even though they
            are valid.</Description>
        </Counter>
      </Category>
    - <Category>
        <CategoryName>tcp</CategoryName>
      - <Counter>
          <CounterName>tcpActiveOpens</CounterName>
          <OID>.1.3.6.1.2.1.6.5.0</OID>
          <Description>The number of times TCP connections have made a direct transition to the SYN-SENT
            state from the CLOSED state.</Description>
        </Counter>
      - <Counter>
          <CounterName>tcpInSegs</CounterName>
          <OID>.1.3.6.1.2.1.6.10.0</OID>
          <Description>The total number of segments received, including those received in
            error.</Description>
        </Counter>
      </Category>
    - <Category>
        <CategoryName>Cpu Attributes</CategoryName>
      - <Counter>
          <CounterName>Cpu Idle%</CounterName>
          <OID>.1.3.6.1.4.1.2021.11.53.0</OID>
          <Units>Cpu%</Units>
          <Description>Cpu Idle% is the percentage of idle processor time.</Description>
        </Counter>
      </Category>
    </CategoryList>
  </OIDCustom>
```

WebLogic7/8 Remote Counters

The following dynamically discovered WebLogic remote extended counter categories are provided in
*QA*Load. Each category provides counters that extend the monitoring of your WebLogic system. The
categories, counter names, and parameters are all dynamically discovered by processing the set of MBeans
available in the WebLogic JMX Server.

WebLogic Application Runtime	WebLogic JMS Session Runtime
WebLogic Connector Service Runtime	WebLogic JTA Recovery Runtime
WebLogic Deployer Runtime	WebLogic JTA Runtime
WebLogic Domain Log Handler Runtime	WebLogic JVM Runtime
WebLogic Domain Runtime	WebLogic Log Broadcaster Runtime
WebLogic EJB Cache Runtime	WebLogic Message Driven EJB Runtime
WebLogic EJB Component Runtime	WebLogic Migratable Service Coordinator

WebLogic EJB Locking Runtime

WebLogic EJB Pool Runtime

WebLogic EJB Transaction Runtime

WebLogic Entity EJB Runtime

WebLogic Execute Queue Runtime

WebLogic JDBC Connection Pool Runtime

WebLogic JMS Connection Runtime

WebLogic JMS Consumer Runtime

WebLogic JMS Destination Runtime

WebLogic JMS Runtime

WebLogic JMS Server Runtime

Runtime

WebLogic Server Life Cycle Runtime

WebLogic Server Runtime

WebLogic Server Security Runtime

WebLogic Servlet Runtime

WebLogic Stateful EJB Runtime

WebLogic Stateless EJB Runtime

WebLogic Time Service Runtime

WebLogic Transaction Resource Runtime

WebLogic Web App Component Runtime

WebLogic Web Server Runtime

WebLogic9 Counters

ServerVantage provides the following dynamically discovered WebLogic9 remote counter categories. Each category provides counters that extend the monitoring of your WebLogic system. The categories, counters names, and parameters are all dynamically discovered by processing the set of MBeans available in the WebLogic JMX Server.

WebLogic App Client Component Runtime

WebLogic Application Runtime

WebLogic Cluster Runtime

WebLogic Component Runtime

WebLogic Connector Component Runtime

WebLogic Connector Connection Pool Runtime

WebLogic Connector Connection Runtime

WebLogic Connector Service Runtime

WebLogic EJB Cache Runtime

WebLogic EJB Component Runtime

WebLogic EJB Locking Runtime

WebLogic EJB Pool Runtime

WebLogic EJB Timer Runtime

WebLogic EJB Transaction Runtime

WebLogic Entity Cache Cumulative Runtime

WebLogic Entity Cache Current State Runtime

WebLogic Execute Queue Runtime

WebLogic Interception Component Runtime

WebLogic JVM Runtime

WebLogic Library Runtime

WebLogic Log Broadcaster Runtime

WebLogic Max Threads Constraint Runtime

WebLogic Message Driven EJB Runtime

WebLogic Min Threads Constraint Runtime

WebLogic NonXA Resource Runtime

WebLogic Persistent Store Connection Runtime

WebLogic Persistent Store Runtime

WebLogic Query Cache Runtime

WebLogic Request Class Runtime

WebLogic SAF Agent Runtime

WebLogic SAF Remote Endpoint Runtime

WebLogic Server Channel Runtime

WebLogic Server Life Cycle Runtime

WebLogic Server Life Cycle Task Runtime

WebLogic Server Runtime

WebLogic Server Security Runtime

WebLogic JDBC Data Source Runtime

WebLogic JDBC Data Source Task Runtime

WebLogic JMS Component Runtime

WebLogic JMS Connection Runtime

WebLogic JMS Consumer Runtime

WebLogic JMS Destination Runtime

WebLogic JMS Durable Subscriber Runtime

WebLogic JMS Pooled Connection Runtime

WebLogic JMS Producer Runtime

WebLogic JMS Remote Endpoint Runtime

WebLogic JMS Runtime

WebLogic JMS Server Runtime

WebLogic JMS Session Pool Runtime

WebLogic JMS Session Runtime

WebLogic Jolt Connection Pool Runtime

WebLogic Jolt Connection Service Runtime

WebLogic JRockit Runtime

WebLogic JTA Recovery Runtime

WebLogic JTA Runtime

WebLogic Servlet Runtime

WebLogic Task Runtime

WebLogic Thread Pool Runtime

WebLogic Transaction Name Runtime

WebLogic Transaction Resource Runtime

WebLogic User Lockout Manager Runtime

WebLogic WAN Replication Runtime

WebLogic Web App Component Runtime

WebLogic Web Server Runtime

WebLogic WLDF Archive Runtime

WebLogic WLDF Data Access Runtime

WebLogic WLDF Dbstore Archive Runtime

WebLogic WLDF File Archive Runtime

WebLogic WLDF Harvester Runtime

WebLogic WLDF Image Creation Task Runtime

WebLogic WLDF Instrumentation Runtime

WebLogic WLDF Watch Notification Runtime

WebLogic WLDF Wlstore ArchiveRuntime

WebLogic Work Manager Runtime

WebLogic Wsee Operation Runtime

WebLogic WSRM Remote Endpoint Runtime

WebSphere Remote Extended Counters

The following dynamically discovered WebSphere remote extended counter categories are provided in *QA*Load. Each category provides counters that extend the monitoring of your WebSphere system. The categories, counter names, and parameters are all dynamically discovered by processing data available from the WebSphere Performance Monitoring Infrastructure.

Remote monitoring supports WebSphere versions: 4.0+, 5.0, and 6.0. The counters supported vary by version.

WebSphere Alarm Manager Counters	WebSphere ORB Perf Module
WebSphere Bean Module	WebSphere Scheduler Module
WebSphere Cache Module	WebSphere Servlet Sessions Module
WebSphere Connection Pool Module	WebSphere System Module
WebSphere DCS Stack Counters	WebSphere Thread Pool Module
WebSphere High Availability Manager Counters	WebSphere Transaction Module
WebSphere J2C Module	WebSphere Web App Module
WebSphere JVM Runtime Module	WebSphere Web Services Counters

WebSphere MQ Remote Extended Counters

The following extended WebSphere MQ remote counters are provided in QALoad. These counters extend the monitoring of your WebSphere MQ system:

Channel Events	Queue Manager Connections
Channel Status	Queue Manager Events
Error Log Entries	Queue Manager Statistics
Percent Queue Depth	Queue Manager Up/Down
Performance Events	Queue Statistics
Queue Depth	

WMI Remote Extended Counters

The following extended WMI (Windows Management Instrument) remote counters are provided in *QA*Load. To display and use the extended counters in task configuration, you must configure user access with the MMC (Microsoft Management Console) and configure the WMI agent using the ServerVantage Agent Console (Reconfigure Agent). These procedures are described in the topic *Configuring WMI* in the ServerVantage Agent Configuration online help. Once configuration is complete, and you select WMI collector as your Server Type during task configuration on the Select Counters page, ServerVantage discovers the Windows registry counters and the extended counters for each WMI-configured server.

These counters extend the monitoring of your WMI system:

WMI WQL

WMI Top Ten Counters:

CPU Utilization % - Top Ten

Memory Utilization % - Top Ten

I/O Utilization % - Top Ten

Oracle Application Server Counters

ServerVantage provides the following dynamically discovered Oracle Application Server (AS) remote extended counter categories for remote monitoring of Oracle10g Application Server performance metrics. Each category provides counters and parameters that extend the monitoring of your Oracle AS system. The

Oracle AS agent dynamically discovers all available counters and parameter values. The available categories and metrics vary by installation. The Oracle AS agent supports wild-carded parameters and resource blackouts.

Supported platforms for Oracle AS include:

Solaris	Microsoft Windows Server 2003 (32-bit)
AIX	Microsoft Windows XP (not all components are supported)
HP	
Linux	

Oracle AS Counter Categories

Oracle AS EJB Method Metrics	Oracle AS JMS Session Metrics
Oracle AS Entity Bean Metrics	Oracle AS JMS Store Metrics
Oracle AS HTTP OC4J Metrics	Oracle AS JMS Temp Destination Metrics
Oracle AS HTTP Server Metrics	Oracle AS JServ JSP Metrics
Oracle AS HTTP Server Module Metrics	Oracle AS JServ Metrics
Oracle AS HTTP Server Response Metrics	Oracle AS JServ Servlet Metrics
Oracle AS HTTP Server Virtual Host Metrics	Oracle AS JServ Zone Metrics
Oracle AS JDBC Connection Metrics	Oracle AS JSP Metrics
Oracle AS JDBC Connection Source Metrics	Oracle AS JVM Metrics
Oracle AS JDBC Metrics	Oracle AS Notification Server Metrics
Oracle AS JDBC Statement Metrics	Oracle AS OC4J Transaction Manager Metrics
Oracle AS JMS Browser Metrics	Oracle AS PLSQL Metrics
Oracle AS JMS Connection Metrics	Oracle AS Portal Engine Metrics
Oracle AS JMS Consumer Metrics	Oracle AS Process Manager Metrics
Oracle AS JMS Durable Subscription Metrics	Oracle AS Servlet Metrics
Oracle AS JMS Metrics	Oracle AS Task Manager Metrics
Oracle AS JMS Persistence Metrics	Oracle AS Web Module Metrics
Oracle AS JMS Producer Metrics	

10g Release 2 Counter Categories

Oracle AS Portal Cache Metrics	Oracle AS Portal Page Metrics
Oracle AS Portal Cache Summary Metrics	Oracle AS Portal DB Repository Metrics
Oracle AS Portal DB Provider Metrics	Oracle AS Portal Web Provider Metrics

JVM Counters

ServerVantage provides the following statically discovered categories for monitoring a Java Virtual Machine (JVM). Each category provides counters that allow the monitoring of your JVM. ServerVantage utilizes the Java Monitoring and Management API which were introduced in J2SE 5.0 for counter data.

Using the Conductor

JVM Class Loading	JVM Memory
JVM Compilation	JVM Operating System
JVM Garbage Collection	JVM Threads

Guideline:

You must start your JVM as a "JMX-enabled JVM"by inserting the following properties:

```
C:\...\java -Dcom.sun.management.jmxremote.port=1095 –
Dcom.sum.management.jmxremote.ssl=false   –
Dcom.sun.management.jmxremote.authenticate=false
```

For more information, see http://java.sun.com/j2se/1.5.0/docs/guide/management/agent.html.

RStat Counters

ServerVantage Agent provides statically discovered categories for monitoring Unix using RStat. RStat is a Unix command to acquire statistics about the network including socket status, interfaces that have been auto-configured, memory statistics, and routing tables.

Collision Rate	Packets In
Context Switches	Packets Out
Disk Transfer Rate	Page In Rate
Fifteen Minute Load Average	Page Out Rate
Five Minute Load Average	Paging I/O Rate
Incoming Packet Error Rate	RPC Status
Interrupt Rate	Swap In Rate
Network Errors	Swap Out Rate
Nice Mode CPU Utilization Percentage	System Mode CPU Utilization Percentage
One Minute Load Average	System Uptime
Outgoing Packet Error Rate	Total CPU Utilization Percentage
	Total Packet Rate
	Use Mode CPU Utilization Percentage

Managing Counters

Adding Counters to a Task Using New Discovery Data

Add counters to a monitoring task by generating the available counter data and selecting the counters and instances to add to the task.

To add counters and instances to a monitoring task:

1. From the Conductor menu bar, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks dialog window.

2. Open or create a task.

3. In the menu bar, click **Actions>Add counter>Use new discovery counters**. The Edit Existing Monitor Wizard appears.

4. Follow the instructions for using the wizard to discover and add counters to the monitoring task.

Notes:

- (WebLogic and WebSphere) When the monitor to which you are adding counters is managed by an administrative server, the Edit Existing Monitor wizard appears and the counter discovery process begins immediately.

- (SNMP) You can supplement the counter discovery list by creating and specifying a custom OID file. See Customizing SNMP Counter Discovery for more information.

Adding Counters to a Task Using Cached Discovery Data

It is possible to add counters to monitor for a machine and monitor type using cached discovery data. To add counters to a task using cached discovery, do the following:

Step 1: Select the counter to add or modify

Step 2: Choose the instances of the counter to monitor

Step 3: Review the monitor definition

Step 4: Save the task

Step 1: Select the counter to add or modify:

1. From the Conductor menu bar, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks dialog window.

2. In the menu bar, click **Actions>Add counter>Use cached discovery counters**. This **Add/Edit Counters** dialog box appears.

3. From the **Available Items** pane, select the **Template** tab or the **Counter** tab.

4. To add an item, select a template or a counter, and click **Add,** or double-click the item to display it in the Selected Items pane. Click **Add All** to add all the items on the selected tab to the **Selected Items** pane.

5. To remove an item, double-click the item in the Selected Items pane or select the item and click **Remove**. The item is returned to the **Available Items** pane.

Note: Select multiple counters and templates by doing one of the following:

- To select nonadjacent counter items, click a counter item, and then hold down **Ctrl** and click each additional counter item.

- To select adjacent counter items, click the first counter item in the sequence, and then hold down **Shift** and click the last counter item.

6. Click **Next**. The **Choose Instances** dialog box displays.

Note: When you select a template, and some of the counters it contains are not present on the machine you are defining, you receive a message with a list of the counters that will not be added to the task.

Step 2: Choose the instances of the counter to monitor:

Using the Conductor

When clicking **Next** in the previous dialog box. The **Choose Instances** dialog box appears.

1. Review the counters selected. When a red dot appears next to a counter, select an instance of the counter.

2. Double-click the counter group to display the counters.

3. Select a counter and click **Edit**. The **Select instance for counter** dialog box appears.

4. In the **Available Instance** pane, select an instance and click **Add**.

5. Repeat until all instances of the counter that you want to apply to the task are selected.

6. Click **Save**. The **Choose Instances** dialog box appears.

7. Repeat this process for each designated counter.

8. Click **Next**. The **Summary** dialog box displays.

Step 3: Review the monitor definition:

1. In the Review Monitor Definition dialog box, review the information for the monitoring machine you defined.

2. Select one of the following:

   - **Set up another monitor for this task** - returns to the Enter properties of the monitoring machine dialog box so you can add another monitor to the monitoring task. When complete, proceed with Step 3 below.

   - **Continue without adding any more monitors** - continues in this dialog box.

   🔲 **Note**: (WebLogic and WebSphere) When you set up another monitor in a managed server environment, you only can add another monitor using a different administrative server.

   - To add a WebLogic monitor, you must use the same WebLogic jar files and WebLogic version as the current monitor.

   - To add a WebSphere monitor, you must use the same WebSphere Home, WebSphere client version, and WebSphere server version as the current monitor.

3. (Optional) In the **Monitors** pane, select the monitor type and click **Save as Template** to create a template for this monitoring task.

4. (Optional) In the **Monitors** pane, select a monitor and click **Remove Monitor** to delete a monitor from the task.

5. (Optional) Type a new value in the **Sample Interval** field. This is the frequency, in seconds, at which QALoad requests data during runtime data collection.

6. Click **Next** to proceed to the next step, where you review and save your selections.

Step 4: Save the task:

When clicking **Next** in the previous dialog box, the **Summary** dialog box appears.

1. On the **Summary** dialog box, review the monitors and counters selected for the template. Click **Back** to return to a dialog box and make changes to the information.

2. Click **Finish** to add the counters.

Removing a Monitor or a Counter from a Monitoring Task

Remove a monitor or a counter from a monitoring task, by following this procedure.

To remove a counter from a monitoring task:

1. In the **Monitors** pane of the Manage Monitoring Tasks window, select the monitor, counter, or counter family to delete.

2. Click **Actions>Remove Monitor/Counter**.

3. When the verification dialog box displays, click **OK**.

> **Note**: You cannot remove the last monitor on a machine, the last counter in the family, or the last family of counters in the task.

Monitoring Templates

About Monitoring Templates

Monitoring templates are designed to facilitate the configuration process. A monitoring template is a predefined group of counters not associated with a specific machine. You can create a new template for a monitoring task, or you can use one of QALoad's pre-defined templates.

When you create a custom template, QALoad's New Monitoring Template wizard guides you through the process of defining the type of template you want to create, configuring the monitor properties, and adding the counters and instances of counters to the template.

When you use one of QALoad's predefined templates, you select a stored template with the counters you want to monitor. The templates have counters grouped by functionality, such as Network Traffic, Response Time, or System Health. Where appropriate, the templates include the specific instances to monitor for each counter.

You can add or edit counters in either custom or pre-defined templates. When you open a template to edit it for the first time, Edit Monitoring Template wizard guides you through the process of discovering and adding new counters to a template. When you've just completed the counter discovery process for a template, either by creating a new template or by opening a template for editing, you can select counters from those already available in memory by using the cached discovery.

Custom Templates

You can create templates of the monitoring tasks that you develop so that all of the counters and instances for the task are saved. You can create new tasks and incorporate the template you created. Templates are saved as .xml files in the Templates directory.

You can create a template when you define a monitoring task, or you can use the New Monitor Template wizard to create and store a template for future use. Custom templates can be modified using either new discovery data or cached discovery data.

Pre-defined Templates

About Pre-defined Templates

*QA*Load provides pre-defined templates for each monitor type. Each template includes the counters most commonly used for particular task within each monitor type.

Using the Conductor

QALoad provides the templates form the following monitor types:

Oracle Application Server

JVM

SAP

SNMP

WebLogic

WebSphere

WebSphere MQ

Windows Registry

WMI

Note: You cannot modify pre-defined templates.

Viewing Pre-defined Templates

QALoad provides pre-defined templates for each monitor type. These include the counters most commonly used for particular task.

To access and review pre-defined templates:

1. In the Conductor, select **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. In the Manage Monitoring Tasks menu bar, click **Templates>Open Existing**. The Select a Monitor Template File dialog box appears and lists pre-defined templates and custom templates you created and saved.

3. Double-click a monitor type, then select a template file and click **Open**.

4. The template name displays in the Manage Monitoring Tasks dialog box.

JVM Templates

JVM Templates

ServerVantage has customized the following JVM Server templates:

JVM Availability

JVM Performance

Additionally, *QA*Load offers a customized JVM template for monitoring BlazeDS activity:

JVM BlazeDS

JVM Availability

This template monitors the availability of your JVM server.

The default sampling interval rate for this template is 5 minutes.

The JVM Availability template monitors the following counters and categories:

Category	Counters	Description
Class Loading	Current Loaded Class Count	Number of classes that are currently loaded in the JVM.

Memory	Total Used Memory	Amount of memeory (in bytes) currently in use by the JVM.
Operating System	Available Processor Count	Number of processors available to the JVM.
Threads	Live Thread Count	Number of live threads.

JVM Performance

This template monitors the performance of your JVM server.

The default sampling interval rate for this template is 5 minutes.

The JVM Performance template monitors the following counters and categories:

Category	Counters	Description
Compilation	Compilation Time	Approximate time (in milliseconds) spent in compilation by the JVM during the sample interval.
Garbage Collection	Total Garbage Collection Time	Total approximate time (in milliseconds) spent garbage collecting by the JVM during the sample interval.

JVM BlazeDS

This template monitors the BlazeDS server using JVM counters:

Category	Counters	Description
Class Loading	Current Loaded Class Count	Number of classes that are currently loaded in BlazeDS.
Compilation	Compilation Time	Approximate time (in milliseconds) spent in compilation by BlazeDS during the sample interval.
Garbage Collection	Total Garbage Collection Time	Total approximate time (in milliseconds) spent garbage collecting by the BlazeDS during the sample interval.
Memory	Total Used Memory	Amount of memory (in bytes) currently in use by BlazeDS.
Threads	Live Thread Count	Number of live threads.

Oracle Application Server Templates

Using the Conductor

Oracle Application Server Template Index

QALoad provides the following pre-defined Oracle Application Server (AS) 10g database templates:

Oracle AS Availability

Oracle AS Performance

Oracle Application Server Availability

This template monitors the availability of an Oracle Application Server (AS) 10g database.

> 🔹 **Note**: This template is only available if you have Oracle AS 10g installed on your monitored server.

This template includes the following counters and categories:

Category	Counters	Description
Oracle AS HTTP OC4J Metrics	OC4JUnavailable	Total number of times that an oc4j JVM could not be found to service requests.
	UnableToHandleReq	Total number of times mod_oc4j declined to handle a request.
Oracle AS HTTP Server Metrics	readyChildren	Number of child processes that are ready to run.
Oracle AS Process Manager Metrics	reqFail	Number of HTTP requests which fail.

Oracle Application Server Performance

This template monitors the availability of an Oracle Application Server (AS) 10g database.

> 🔹 **Note**: This template is only available if you have Oracle AS 10g installed on your monitored server.

This template includes the following counters and categories:

Category	Counters	Description
Oracle AS HTTP OC4J Metrics	ErrReqSess	Specifies the total number of session requests that mod_oc4j failed to route to an OC4J process.
	SucReqSess	Specifies the total number of session requests that mod_oc4j successfully routed to an OC4J process.
Oracle AS HTTP Server Metrics	busyChildren	Number of child processes active.
	connection.avg	Average time spent servicing HTTP connections.
	request.avg	Average time required to service an HTTP request.

RStat Templates

QALoad RStatd Health

This template monitors the status of the Remote Procedure Call (RPC) to the monitored Unix computer.

The QALoad RStatd Health template uses the following RStat extended counters:

Counters	Description
RPC Status	a Boolean rstat parameter that shows the status of the Remote Procedure Call (RPC) to the monitored Unix computer. A status of Down means that the RSM cannot communicate with or collect performance information about the computer from the rstatd server.

QALoad Server Availability

This template monitors the availability of UNIX servers.

The QALoad Server Availability template uses the following RStat extended counters:

Counters	Description
System Uptime	shows the amount of time since the monitored element was started.

QALoad Server Performance

This template monitors the performance of the UNIX server.

The QALoad Server Performance template uses the following RStat extended counters:

Counters	Description
Context Switches	shows the per-second number of CPU context switches.
Paging I/O Rate	shows the amount of CPU time across all processors that was spent waiting for input and output operations.
RPC Status	a Boolean rstat parameter that shows the status of the Remote Procedure Call (RPC) to the monitored Unix computer. A status of Down means that the RSM cannot communicate with or collect performance information about the computer from the rstatd server.
Total CPU Utilization Percentage	shows the percentage of CPU that currently is not in the idle state for all CPUs on the monitored computer.

QALoad Server Health

This template monitors the overall condition of the UNIX server.

The QALoad Server Health template uses the following RStat extended counters:

Counters	Description
Disk Transfer Rate	shows the per-second rate of disk transfers for each disk on the monitored element.
Paging I/O Rate	shows the amount of CPU time across all processors that was spent waiting for input and output operations.
Total CPU Utilization Percentage	shows the percentage of CPU that currently is not in the idle state for all CPUs on the monitored computer.

SAP Templates

SAP Templates

*QA*Load provides the following pre-defined SAP templates:

*QA*Load-SAP R3 Remote Availability

*QA*Load-SAP R3 Remote Performance

*QA*Load-SAP R3 Remote System Errors

QALoad-SAP R3 Remote Availability

This template monitors the availability of an SAP R/3 Instance. The SAP R/3 Availability template returns critical information about the availability of your SAP installation. One metric used to determine the availability of an SAP R/3 Instance is the status of the SAP collector.

The default event action assigned to this template issues an alarm if either the specified R/3 Instance or the collector goes down. The default instance is the first SAP Instance configured for monitoring during installation.

The SAP R/3 Availability template uses the following SAP R/3 extended counters:

Counters	Description
Active Servers	Returns the number of active SAP application servers for a given instance. It detects when a remote server is unavailable. Rule: IF 'SAP R/3 Remote Extended.Active Servers(SAP Instance: "**", Server Count: "10")' = 0 .

QALoad-SAP R3 Remote Performance

This template monitors the performance of your SAP R/3 Instance.

The default event action for this template raises an event if the number of alerts of critical status is greater than 0, or if the buffer hit ratio falls below 95%.

All the counters associated with this template require the instance number of your SAP installation. By default, this template uses the first instance configured for monitoring during ServerVantage installation. If you use the task configuration wizard to change the instance that the template monitors, you must also change the instance specified in the rule accordingly.

The SAP R/3 Performance template uses the following SAP R/3 extended counters:

Counters	Description
Buffer Statistic	Returns different buffer statistics for selected buffer name. This counter was chosen because buffering data is a key to the performance of SAP. Rule: IF 'SAP R/3 Remote Extended.Buffer Statistic(SAP Instance: "**", Buffer Name: "TTAB", Statistic Name: "Hit rate SAP buffer(%%)")' < 95.
Itemized Spool Queue	Return number of entries in the spool queue that match the specified criteria. Rule: IF 'SAP R/3 Remote Extended.Spool Queue(SAP Instance: "**", Request Status: "Processing")' > 10.
Memory Usage	Returns current memory usage. Rule: IF 'SAP R/3 Remote Extended.Memory Usage(SAP Instance: "**", Count: "10", Metrics: "MB")' > 10000.
Page/Roll Area	Returns Used Paging Area % statistic. This counter was chosen because roll memory is critical for work processes and page memory is critical for internal data processing.
Work Processes	Counter for monitoring SAP R/3 work processes. Returns number of stopped work processes. Rule: IF 'SAP R/3 Remote Extended.Work Processes(SAP Instance: "**", Process Type: "BGDDIAENQSPOUP2UPD", Process State: "Stopped")' > 2.

QALoad-SAP R3 Remote System Errors

This template monitors the errors and critical situations that occur on a SAP R/3 system. Rules and thresholds are preset to appropriate values for most sites.

The default sampling interval for this template is 5 minutes.

The SAP R/3 Performance template uses the following SAP R/3 Remote extended counters:

Counters	Description
Alerts	Counter for monitoring R/3 alerts. Returns number of alerts according to the specified criteria. This counter checks all alerts with error (red) status. Rule: IF 'SAP R/3 Remote Extended.Alerts(SAP Instance: "**", Monitor Set: "SAP CCMS Admin Workplace", Monitor: "Database", Severity: "Error - Red", Pattern: "*", Show Alert Text: "No")' > 0.
Itemized Spool Queue	Return number of entries in the spool queue that match the

	specified criteria.
Spool Queue	Return number of entries in the spool queue that match the specified criteria. This counter checks all spool entries with "Problem" status.
	Rule: IF 'SAP R/3 Remote Extended.Spool Queue(SAP Instance: "**", Request Status: "Problem")' > 0.
Work Processes	Counter for monitoring SAP R/3 work processes. Returns number of work processes according to the specified criteria. This counter checks stopped work processes.
	Rule: IF 'SAP R/3 Remote Extended.Work Processes(SAP Instance: "**", Process Type: "BGDDIAENQSPOUP2UPD", Process State: "Stopped")' > 0.

SNMP Templates

SNMP Templates

*QA*Load provides the following pre-defined SNMP templates:

*QA*Load-HP Performance

*QA*Load-Linux Performance

*QA*Load-SUN Performance

QALoad-HP Performance

This template includes the following counters and categories:

Category	Counters	Description
HP System	CpuIdle%	CpuSys% is the percentage of idle processor time.
	CpuSys%	CpuSys% is the percentage of non-idle processor time that is spent in system mode.
	CpuUser%	CpuUser% is the percentage of non-idle processor time that is spent in user mode.
	FreeMemory KBytes	FreeMemory KBytes is the amount of idle memory.
	FreeSwap KBytes	FreeSwap is the amount of free swap space on the system.
	MaxUserMem KBytes	MaxUserMem is the amount of maximum user memory on the system.
	Users	Users is the number of users logged on to the machine.
tcp	tcpInSegs/sec	tcpInSegs/sec is the rate at which segments are

		received, including those received in error.
	tcpOutSegs/sec	tcpOutSegs/sec is the rate at which segments are sent, including those on current connections but excluding those containing only retransmitted octets.
udp	udpInDatagrams/sec	udpInDatagrams/sec is the rate of UDP datagrams being delivered to UDP users.
	udpOutDatagrams/sec	udpOutDatagrams/sec is the rate at which UDP datagrams are sent.

QALoad-Linux Performance

This template includes the following counters and categories:

Category	Counters	Description
Linux System	CpuIdle%	CpuSys% is the percentage of idle processor time.
	CpuSys%	CpuSys% is the percentage of non-idle processor time that is spent in system mode.
	CpuUser%	CpuUser% is the percentage of non-idle processor time that is spent in user mode.
	Interrupts/sec	Interrupts/sec is the rate of system interrupts.
	PagesIn KBytes/sec	PagesIn KBytes/sec is the rate of pages read in from disk.
	PagesOut KBytes/sec	PagesOut KBytes/sec is the rate of pages written to disk.
	SwapIn KBytes/sec	SwapIn KBytes/sec is the rate at which pages are being swapped in.
	SwapOut KBytes/sec	SwapOut KBytes/sec is the rate at which pages are being swapped out.
tcp	tcpInSegs/sec	tcpInSegs/sec is the rate at which segments are received, including those received in error.
	tcpOutSegs/sec	tcpOutSegs/sec is the rate at which segments are sent, including those on current connections but excluding those containing only retransmitted octets.

udp	udpInDatagrams/sec	udpInDatagrams/sec is the rate of UDP datagrams being delivered to UDP users.
	udpOutDatagrams/sec	udpOutDatagrams/sec is the rate at which UDP datagrams are sent.

QALoad-SUN Performance

This template includes the following counters and categories:

Category	Counters	Description
Sun System	CpuIdle%	CpuSys% is the percentage of idle processor time.
	CpuSys%	CpuSys% is the percentage of non-idle processor time that is spent in system mode.
	CpuUser%	CpuUser% is the percentage of non-idle processor time that is spent in user mode.
	Interrupts/sec	Interrupts/sec is the rate of system interrupts.
	PagesIn KBytes/sec	PagesIn KBytes/sec is the rate of pages read in from disk.
	PagesOut KBytes/sec	PagesOut KBytes/sec is the rate of pages written to disk.
	SwapIn KBytes/sec	SwapIn KBytes/sec is the rate at which pages are being swapped in.
	SwapOut KBytes/sec	SwapOut KBytes/sec is the rate at which pages are being swapped out.
tcp	tcpInSegs/sec	tcpInSegs/sec is the rate at which segments are received, including those received in error.
	tcpOutSegs/sec	tcpOutSegs/sec is the rate at which segments are sent, including those on current connections but excluding those containing only retransmitted octets.
udp	udpInDatagrams/sec	udpInDatagrams/sec is the rate of UDP datagrams being delivered to UDP users.
	udpOutDatagrams/sec	udpOutDatagrams/sec is the rate at which UDP datagrams are sent.

WebLogic Templates

WebLogic Templates

*QA*Load provides the following pre-defined WebLogic templates:

*QA*Load-WebLogic Availability

*QA*Load-WebLogic EJB Performance

*QA*Load-WebLogic JDBC Performance

*QA*Load-WebLogic JMS Performance

*QA*Load-WebLogic Performance

*QA*Load-WebLogic Server Security

*QA*Load-WebLogic Servlet Performance

QALoad-WebLogic Availability

This template monitors the availability of a WebLogic server. The WebLogic Availability template returns critical information about the availability of your WebLogic installation.

The default sampling interval for this template is 5 minutes.

The WebLogic Availability template uses the following WebLogic extended counters:

Category	Counters	Description
ExecuteQueueRuntime	ExecuteQueueRuntime_PendingRequestOldestTime	Returns the time that the longest waiting request was placed in the queue. Rule: The Application Server is not in running mode if this counter value is > 50.
ServerRuntime	ServerRuntime_StateVal	Returns current state of the server. This counter provides a more detailed state than available or not. Rule: The Application Server is not in running mode if this counter value is <> 2.
ServerSecurityRuntime	ServerSecurityRuntime_LockedUsersCurrentCount	Returns the number of currently locked users on this server.

		Rule: There are a high number of users locked out if this counter value is > 5.

QALoad-WebLogic EJB Performance

This template monitors the EJB performance of a WebLogic server. The WebLogic EJB Performance template returns critical information about the performance of your WebLogic installation.

The default sampling interval for this template is 5 minutes.

The WebLogic EJB Performance template uses the following WebLogic extended counters:

Category	Counters	Description
EJBCacheRuntime	EJBCacheRuntime_ActivationCount	Returns the total number of times the EJB was activated. Rule: There is inefficient cache access if the number of activations is > 20.
	EJBCacheRuntime_CacheAccessCount	Returns the total number of attempts to access a bean from the cache.
	EJBCacheRuntime_CachedBeansCurrentCount	Returns the total number of beans from this EJB Home currently in the EJB cache.
	EJBCacheRuntime_CacheHitCount	Returns the total number of times an attempt to access a bean from the cache succeeded. The cacheHitCount value subtracting the cache miss count from the cache access count.
	EJBCacheRuntime_PassivationCount	Returns the total number of beans from this EJB Home that have been passivated. Rule: There is inefficient cache access if the number of passivations is > 20.
EJBLockingRuntime	EJBLockingRuntime_LockEntriesCurrentCount	Returns the number of currently locked users on this server.
	EJBLockingRuntime_TimeoutTotalCount	Returns the current number Threads

		that have timed out waiting for a lock on a bean.
	EJBLockingRuntime_WaiterTotalCount	Returns the number of objects waiting on the lock.
		Rule: There are a lot of objects waiting if the interval value of this counter is > 10.
EJBPoolRuntime	EJBPoolRuntime_BeansInUseCurrentCount	Returns the number of bean instances currently being used from the free pool.
	EJBPoolRuntime_IdleBeansCount	Returns the total number of available bean instances in the free pool.
	EJBPoolRuntime_TimeoutTotalCount	Returns the total number of Threads that have timed out waiting for an available bean instance from the free pool.
		Rule: There are a lot of objects timing out if the interval value of this counter is > 20.
	EJBPoolRuntime_WaiterTotalCount	Returns the total number of Threads currently waiting for an available bean instance from the free pool.
		Rule: There are a lot of objects waiting if the interval value of this counter is > 10.
EJBTransactionRuntime	EJBTransactionRuntime_TransactionsCommittedTotalCount	Returns the total number of transactions that have been committed for this EJB.
		Rule: There is high transaction overhead if the interval value of this counter is > 20.
	EJBTransactionRuntime_TransactionsRolledBackTotalCount	Returns the total number of transactions that have been rolled back for this EJB.
		Rule: There is high transaction overhead if the interval value of this counter is > 20.
	EJBTransactionRuntime_TransactionsTimedOutTotalCount	Returns the total number of transactions that have timed out for this EJB.
		Rule: There is high transaction overhead if the interval value of this counter is > 20.

| MessageDrivenEJBRuntime | MessageDrivenEJBRuntime_JMSConnectionAlive | Returns a boolean of the status of the connection. This counter displays the state of a JMS connection. |
| | | Rule: The JMS Connection is down if this counter value is = 0. |

QALoad-WebLogic JDBC Performance

This template monitors the JDBC performance of a WebLogic server. The WebLogic JDBC Performance template returns critical information about the performance of your WebLogic installation.

The default sampling interval for this template is 5 minutes.

The WebLogic JDBC Performance template uses the following WebLogic extended counters:

Category	Counters	Description
JDBC Connection Pool Runtime	ActiveConnectionsCurrentCount	Returns the current number of active connections.
	ActiveConnectionsHighCount	Returns the highest number of active current connections. The count starts at zero each time the JDBCConnectionPoolRuntimeMBean is instantiated.
	ConnectionDelayTime	Returns the number of milliseconds it takes to get a physical connection from the database. It is calculated as summary time to connect divided by summary number of connections.
	ConnectionsTotalCount	Returns the total number of JDBC connections in this DBCConnectionPoolRuntimeMBean since the pool was instantiated.
	FailuresToReconnectCount	Returns the number of attempts to refresh a connection to a database that failed. Failure may be due to the database being unavailable or a broken connection to the database.
		Rule: There are a high number of connection reconnect failures when this counter value is > 1.
	LeakedConnectionCount	Returns the number of connections that were checked out from the

		connection pool but were not returned to the pool by calling close (). Rule: There is a lot of connection pool leakage if this counter value is > 5.
	PoolState	Current state of the connection pool. Returns True if the pool is enabled, False if the pool is disabled.
	PrepStmtCacheMissCount	Returns a count of the cases when the cache does not have a cached statement to satisfy a request.
	WaitingForConnectionHighCount	The high water mark of waiters for a connection in this JDBCConnectionPoolRuntimeMBean. The count starts at zero each time the JDBCConnectionPoolRuntimeMBean is instantiated.
	WaitSecondsHighCount	Returns the highest number of seconds a connection waited. Rule: There is a long wait for the connection pool if this counter value is > 120.

QALoad-WebLogic JMS Performance

This template monitors the JMS performance of a WebLogic server. The WebLogic JMS Performance template returns critical information about the performance of your WebLogic installation.

The default sampling interval for this template is 5 minutes.

The WebLogic JMS Performance template uses the following WebLogic extended counters:

Category	Counters	Description
JMSConnectionRuntime	SessionsCurrentCount	Returns the current number of sessions for this connection.
	SessionsTotalCount	Returns the number of sessions on this connection since the last reset.
JMSRuntime	ConnectionsCurrentCount	Returns the current number of connections to this WebLogic Server.
	ConnectionsTotalCount	Returns the total number of

		connections made to this WebLogic Server since the last reset.
JMSServerRuntime	MessagesPendingCount	Returns the current number of messages pending (unacknowledged or uncommitted) stored on this JMS server. Pending messages are over and above the current number of messages. Rule: There are a large number of pending messages if this counter value is > 50.
	MessagesReceivedCount	Returns the number of messages received on this destination since the last reset.
JMSSessionRuntime	ConsumersCurrentCount	Returns the current number of consumers for this session.
	MessagesPendingCount	Returns the number of messages pending (uncommitted and unacknowledged) for this session. Rule: There are a large number of pending JMS Session messages if this counter value is > 50.
	MessagesReceivedCount	Returns the number of messages received on this destination since the last reset.
	MessagesSentCount	Returns the number of bytes sent by this session since the last reset.

QALoad-WebLogic Performance

This template monitors the performance of a WebLogic server. The WebLogic Performance template returns critical information about the performance of your WebLogic installation.

The default sampling interval for this template is 5 minutes.

The WebLogic Performance template uses the following WebLogic extended counters:

Category	Counters	Description

ConnectorServiceRuntime	ConnectionPoolCurrentCount	Returns the number of currently deployed connection pools.
ExecuteQueueRuntime	ExecuteThreadCurrentIdleCount	Returns the number of idle threads assigned to the queue.
	PendingRequestCurrentCount	Returns the number of waiting requests in the queue. Rule: There are a large number of pending requests if this counter value is > 50.
	ServicedRequestTotalCount	Returns the number of requests that have been processed by this queue.
JMSRuntime	ConnectionsCurrentCount	Returns the current number of connections to this WebLogic Server. Rule: There are a large number of JMS connections if this counter value is > 20.
JTARuntime	ActiveTransactionsTotalCount	Returns the number of active transactions on the server.
	SecondsActiveTotalCount	Returns the total number of seconds for all committed transactions.
	TransactionRolledBackResourceTotalCount	Returns the number of transactions that were rolled back due to a resource error.
	TransactionTotalCount	Returns the total number of transactions processed. This total includes all committed, rolled back and heuristic transaction completions.
JVMRuntime	HeapFreeCurrent	Returns the current amount of free memory (in bytes) in the JVM heap.
TimeServiceRuntime	ExceptionCount	Returns the total number of exceptions thrown while executing scheduled triggers. Rule: There are a large number of exceptions if the interval value of this counter is > 20.

	ExecutionsPerMinute	Returns the average number of triggers executed per minute.

QALoad-WebLogic Server Security

This template monitors the security of a WebLogic server. The WebLogic Server Security template returns critical information about the security status of your WebLogic installation.

The default sampling interval for this template is 5 minutes.

The WebLogic Server Security template uses the following WebLogic extended counters:

Category	Counters	Description
ServerSecurityRuntime	InvalidLoginAttemptsTotalCount	Returns the cumulative number of invalid login attempts made on this server. Rule: Multiple invalid login attempts have occurred when the interval value of this counter is > 5.
	LockedUsersCurrentCount	Returns the number of currently locked users on this server. Rule: There are multiple locked users if this counter value is > 5.
	LoginAttemptsWhileLockedTotalCount	Returns the cumulative number of invalid login attempts made on this server while the user was locked.
	UnlockedUsersTotalCount	Returns the number times users have been unlocked on this server.

QALoad-WebLogic Servlet Performance

This template monitors the performance of your WebLogic servlet. The WebLogic Servlet Performance template returns critical information about the servlet performance of your WebLogic installation.

The default sampling interval for this template is 5 minutes.

The WebLogic Servlet Performance template uses the following WebLogic extended counters:

Category	Counters	Description
ServletRuntime	ExecutionTimeAverage	Returns the average time all invocations of the servlet that has executed since the task was created. Rule: The servlet is averaging high execution times if this counter value average is > 10.
	ExecutionTimeHigh	Returns the amount of time the single longest invocation of the servlet that has executed since the task was created.
	ExecutionTimeTotal	Returns the total amount of time all invocations of the servlet that has executed since the task was created.
	InternalServlet	whether this is an Internal Servlet or not
	InvocationTotalCount	Returns the total number of times the servlet has been invoked. Gets the invocationTotalCount attribute of the ServletRuntimeMBean object.
	ReloadTotalCount	Returns the total number of times the servlet is reloaded. Gets the reloadTotalCount attribute of the ServletRuntimeMBean object.

WebSphere Templates

WebSphere Templates

*QA*Load provides the following pre-defined WebSphere templates:

*QA*Load-WebSphere 5.0 and Before JDBC Performance

*QA*Load WebSphere 5.0 and Before Performance

*QA*Load-WebSphere 5.0 and Before Web Application Performance

QAL-WS 6.0 and Later JDBC Performance

QAL-WS 6.0 and Later Performance

QAL-WS 6.0 and Later Web Application Performance

QALoad-WebSphere 5.x and Below JDBC Performance

This template monitors the performance of a WebSphere JDBC server. The WebSphere JDBC Performance template returns critical information about the JDBC performance of your WebSphere installation.

The default sampling interval for this template is 5 minutes.

The WebSphere JDBC Performance template uses the following WebSphere extended counters:

Category	Counters	Description
JDBC Connection Pool Module	connectionPoolModule.avgWaitTime	Average waiting time in milliseconds until a connecti
	connectionPoolModule.concurrentWaiters	WebSphere extended counter for monitoring connectionPoolModule.concurrentWaiters
	connectionPoolModule.faults	Average waiting time in milliseconds until a connecti
	connectionPoolModule.percentMaxed	Average percent of the time that all connections are i Rule: IF 'WebSphere connectionPoolModule.connectionPoolModule.perce "**", Server: "**", Data Source: "all")' > 25.
	connectionPoolModule.percentUsed	Average percent of the pool that is in use.

QALoad-WebSphere 5.x and below Web Application Performance

This template monitors the performance of a WebSphere 5.0 and earlier Web Application server. The WebSphere 5.0 and earlier Web Application Performance template returns critical information about the Web Application performance of your WebSphere installation.

The default sampling interval for this template is 5 minutes.

The WebSphere 5.0 and earlier Web Application Performance template uses the following WebSphere extended counters:

Category	Counters	Description
WebSphere servletSessionsModule	servletSessionsModule.activateNonExistSessions	Number of requests for a session that no longer exists, presumably because the session timed out. This counter may indicate a high number of timeout conditions.
	servletSessionsModule.activeSessions	The number of concurrently active sessions. A session is active if WebSphere is currently processing a request, which uses that session. This counter may indicate high activity.
	servletSessionsModule.cacheDiscards	Number of session objects that have been forced out of the cache. This counter may

Category	Counters	Description
		indicate a need for more memory in the cache.
	servletSessionsModule.invalidatedSessions	Number of sessions invalidated. This counter may indicate a high number of invalidated sessions.
	servletSessionsModule.invalidatedViaTimeout	Number of requests for a session that no CountStatistic exists, presumably because the session timed out. This counter may indicate a high number of timeout conditions.
WebSphere webAppModule	webAppModule.servlets.concurrentRequests	Number of requests that are concurrently processed. This counter may indicate high activity for an application.
	webAppModule.servlets.numErrors	Total number of errors in a servlet or Java Server Page (JSP). This counter may indicate a high number of error incidents.
	webAppModule.servlets.responseTime	Response time, in milliseconds, of a servlet request. This counter may indicate a slow response time of a request.

QALoad WebSphere 5.x and below Performance

This template includes the following counters and categories:

Category	Counters	Description
WebSphere jvmRuntimeModule	jvmRuntimeModule.freeMemory	WebSphere extended counter for monitoring jvmRuntimeModule.freeMemory
	jvmRuntimeModule.usedMemory	WebSphere extended counter for monitoring jvmRuntimeModule.usedMemory

WebSphere orbPerfModule	orbPerfModule.concurrentRequests	WebSphere extended counter for monitoring orbPerfModule.concurrentRequests
	orbPerfModule.interceptors.processingTime	WebSphere extended counter for monitoring orbPerfModule.interceptors.processingTime
	orbPerfModule.referenceLookupTime	WebSphere extended counter for monitoring orbPerfModule.referenceLookupTime
WebSphere systemModule	systemModule.avgCpuUtilization	WebSphere extended counter for monitoring systemModule.avgCpuUtilization
	systemModule.freeMemory	WebSphere extended counter for monitoring systemModule.freeMemory
WebSphere threadPoolModule	hreadPoolModule.activeThreads	WebSphere extended counter for monitoring threadPoolModule.activeThreads

QALoad-WebSphere 6.x and Above Performance

This template includes the following counters and categories:

Category	Counters	Description
WebSphere jvmRuntimeModule	jvmRuntimeModule.freeMemory	WebSphere extended counter for monitoring jvmRuntimeModule.freeMemory
	jvmRuntimeModule.heapSize	Total memory in JVM run time.
	jvmRuntimeModule.usedMemory	WebSphere extended counter for monitoring jvmRuntimeModule.usedMemory
WebSphere systemModule	systemModule.CPUUsageSinceLastMeasurement	Average system CPU utilization taken over the time interval since the last reading. Because the first call is required to perform initialization, an invalid value such as 0 is returned. All subsequent calls return the expected value. On SMP machines, the value returned is the utilization averaged over all

		CPUs.
WebSphere threadPoolModule	threadPoolModule.activeCount	Number of concurrently active threads.
WebSphere transactionModule	transactionModule.activeCount	Number of concurrently active global transactions.
	transactionModule.globalTimeoutCount	Number of global transactions timed out.
	transactionModule.localTimeoutCount	Number of local transactions timed out.
	transactionModule.rolledbackCount	Total number of global transactions rolled back.

QALoad-WebSphere 6.x and Above Web Application Performance

This template monitors the performance of a WebSphere 6.0 and later Web Application server. The WebSphere 6.0 and later Web Application Performance template returns critical information about the Web Application performance of your WebSphere installation.

The default sampling interval for this template is 5 minutes.

The WebSphere 6.0 and later Web Application Performance template uses the following WebSphere extended counters:

Category	Counters	Description
WebSphere servletSessionsModule	servletSessionsModule.externalReadTime	Time (in milliseconds) taken in reading the session data from persistent store. For multi-row sessions, the metrics are for the attribute; for single-row sessions, the metrics are for the whole session. Applicable only for persistent sessions. When using a JMS persistent store, you have the choice of whether to serialize the data being replicated. If you choose not to serialize the data, the counter is not available.
	servletSessionsModule.externalWriteTime	Time (milliseconds) taken to write the

		session data from the persistent store. Applicable only for (serialized) persistent sessions. Similar to externalReadTime described above.
	servletSessionsModule.liveCount	Number of sessions that are currently cached in memory.
	servletSessionsModule.noRoomForNewSessionCount	Applies only to session in memory with AllowOverflow=false. The number of times that a request for a new session cannot be handled because it would exceed the maximum session count.
WebSphere webAppModule	webAppModule.servlets.concurrentRequests	Number of requests that are concurrently processed. This counter may indicate high activity for an application.
	webAppModule.servlets.errorCount	Total number of errors in a servlet or Java Server Page (JSP). This counter may indicate a high number of error incidents.
	webAppModule.requestCount	Total number of requests a servlet processed.
	webAppModule.serviceTime	Response time, in milliseconds, of a servlet request.

QALoad-WebSphere 6.x and Above JDBC Performance

This template monitors the performance of a WebSphere JDBC server. The WebSphere JDBC Performance template returns critical information about the JDBC performance of your WebSphere installation.

The default sampling interval for this template is 5 minutes.

The WebSphere JDBC Performance template uses the following WebSphere extended counters:

Category	Counters	Description
WebSphere	connectionPoolModule.faultCount	Total number of faults, such

Connection Pool Module		as, timeouts, in connection pool.
	connectionPoolModule.percentUsed	Average percent of the pool that is in use.
	connectionPoolModule.prepStmtCacheDiscardCount	Total number of statements discarded by the LRU algorithm of the statement cache.
	connectionPoolModule.waitingThreadCount	Number of threads that are currently waiting for a connection.
	connectionPoolModule.waitTime	Average waiting time in milliseconds until a connection is granted.

WebSphere MQ Templates

WebSphere MQ Templates

*QA*Load provides the following pre-defined WebSphere MQ templates:

*QA*Load-WebSphere MQ Availability

*QA*Load-WebSphere MQ Performance

QALoad-WebSphere MQ Availability

This template monitors the availability of a WebSphere MQ server. The WebSphere MQ Availability template returns critical information about the availability of your WebSphere MQ installation.

The default sampling interval for this template is 5 minutes.

The WebSphere MQ Availability template uses the following WebSphere MQ extended counters:

Counters	Description
Channel Events	Return the number of channel events for the current interval.
Queue Manager Events	Reports the number of queue manager events for the current interval.
Queue Manager Up/Down	Monitors the running state of a queue manager.

QALoad-WebSphere MQ Performance

This template monitors the performance of a WebSphere MQ server. The WebSphere MQ Performance template returns critical information about the performance of your WebSphere MQ installation.

The default sampling interval for this template is 5 minutes.

The WebSphere MQ Performance template uses the following WebSphere MQ extended counters:

Counters	Description
Performance Events	This counter reports the number of performance events for the current interval.

WMI Templates

WMI Templates

QALoad provides the following pre-defined WMI templates:

*QA*Load-Active Monitoring Availability

*QA*Load-Citrix IMA Networking

*QA*Load-Citrix Metaframe All

*QA*Load-Citrix MetaFrame IMA

*QA*Load-Citrix MetaFrame Zone

*QA*Load-Cold Fusion

*QA*Load-Generic Application Availability and Performance

*QA*Load-MS IIS Availability

*QA*Load-MS IIS Performance

QALoad-Active Monitoring Availability

This template includes the following counters and categories:

Category	Counters	Description
Memory	Available MBytes	
Processor	% Processor Time	
System	System Up Time	

QALoad-Citrix IMA Networking

This template includes the following counters and categories:

Category	Counters	Description
Citrix IMA Networking	Bytes Received/sec	

	Bytes Sent/sec	
	Network Connections	
Network Interface	Bytes Total/sec	

QALoad-Citrix Metaframe All

This template includes the following counters and categories:

Category	Counters	Description
Citrix MetaFrame XP	Application Enumerations/sec	
	Application Resolution Time (ms)	
	Application Resolutions/sec	
	Data Store Connection Failure	
	DataStore bytes read/sec	
	DataStore bytes written/sec	
	DataStore reads/sec	
	DataStore writes/sec	
	Dynamic Store bytes read/sec	
	DynamicStore bytes written/sec	
	DynamicStore reads/sec	
	DynamicStore writes/sec	
	Filtered Application Enumerations/sec	
	LocalHostCache bytes read/sec	
	LocalHostCache bytes written/sec	
	LocalHostCache reads/sec	
	LocalHostCache writes/sec	
	Zone Elections	

	Zone Elections Won	
Memory	Page Reads/sec	
PhysicalDisk	% Disk Time	
Processor	% Processor Time	

QALoad-Citrix MetaFrame IMA

This template includes the following counters and categories:

Category	Counters	Description
Citrix MetaFrame XP	Application Enumerations/sec	
	Application Resolution Time (ms)	
	Application Resolutions/sec	
	Data Store Connection Failure	
	DataStore bytes read/sec	
	DataStore bytes written/sec	
	DataStore reads/sec	
	DataStore writes/sec	
	Filtered Application Enumerations/sec	
	LocalHostCache bytes read/sec	
	LocalHostCache bytes written/sec	
	LocalHostCache reads/sec	
	LocalHostCache writes/sec	
Terminal Services	Active Sessions	
	Total Sessions	

QALoad-Citrix MetaFrame Zone

This template includes the following counters and categories:

Category	Counters	Description
Citrix MetaFrame XP	Dynamic Store bytes read/sec	
	DynamicStore bytes written/sec	
	DynamicStore reads/sec	
	DynamicStore writes/sec	
	LocalHostCache bytes read/sec	
	LocalHostCache bytes written/sec	
	LocalHostCache reads/sec	
	Zone Elections	
	Zone Elections Won	
Network Interface	Bytes Total/sec	
	Current Bandwidth	
Terminal Services	Active Sessions	
	Total Sessions	

QALoad-Cold Fusion

This template includes the following counters and categories:

Category	Counters	Description
ColdFusion MX Server	Avg DB Time (msec)	
	Avg Queue Time (msec)	
	Avg Req Time (msec)	
	Bytes In / Sec	
	Bytes Out / Sec	
	DB Hits / Sec	

	Page Hits / Sec	
	Queued Requests	
	Running Requests	
	Timed Out Requests	
Memory	% Committed Bytes In Use	
	Available Bytes	
	Page Faults/sec	
Process	% Processor Time	

QALoad-Generic Application Availability and Performance

This template includes the following counters and categories:

Category	Counters	Description
Process	% Processor Time	
System	System Up Time	

QALoad-MS IIS Availability

This template includes the following counters and categories:

Category	Counters	Description
System	System Up Time	
Web Service	Current Anonymous Users	
	Current Connections	
	Logon Attempts/sec	
	NonAnonymous Users/sec	
	Not Found Errors/sec	
	Total Delete Requests	
	Total Files Sent	

86

	Total Get Requests	
	Total NonAnonymous Users	
	Total Not Found Errors	

QALoad-MS IIS Performance

This template includes the following counters and categories:

Category	Counters	Description
Internet Information Services Global	Current Blocked Async I/O Requests	
	Total Blocked Async I/O Requests	
	Total Rejected Async I/O Requests	
	URI Cache Flushes	
	URI Cache Hits	
	URI Cache Hits %	
	URI Cache Misses	
PhysicalDisk	% Disk Time	
Process	% Processor Time	
Redirector	Current Commands	
	Network Errors/sec	
Server	Work Item Shortages	
Server Work Queues	Queue Length	
Web Service	Not Found Errors/sec	

Windows Registry Templates

Windows Registry Templates

Using the Conductor

QALoad provides the following pre-defined Windows Registry templates:

QALoad-Active Monitoring Availability

QALoad-Citrix IMA Networking

QALoad-Citrix Metaframe all

QALoad-Citrix Metaframe IMA

QALoad-Citrix Metaframe Zone

QALoad-Cold Fusion

QALoad-Generic Application Availability and Performance

QALoad-MS IIS Availability

QALoad-MS IIS Performance

QALoad-Server Health

QALoad-Windows Availability

QALoad-Windows Performance

QALoad-Active Monitoring Availability

This template includes the following counters and categories:

Category	Counters	Description
Memory	Available MBytes	This counter monitors the Active Monitoring client site and notifies you when it is low on resources, where Processor time is > 95% for more than 3 intervals. The parameter for this counter is **Instance**. The default is _Total.
Processor	% Processor Time	Raise an event when Active Monitoring client site is low on memory resources, where Available Memory is at or below 1MB for more than 3 intervals.
System	System Up Time	This counter tests the network connection between two machines and monitors the communication status of the machine that receives communication.

QALoad-Citrix IMA Networking

This template includes the following counters and categories:

Category	Counters	Description
Citrix IMA Networking	Bytes Received/sec("_Total")	This counter monitors the total bytes received per second.
	Bytes Sent/sec("_Total")	This counter monitors the total bytes sent per second.
	Network Connections	This counter monitors the network connections.
Network Interface	Bytes Total/sec	This counter monitors the network connection total bytes/sec.

QALoad-Citrix Metaframe all

This template includes the following counters and categories:

Category	Counters	Description
Citrix MetaFrame XP	Application Enumerations/sec	This counter monitors application enumerations / sec.
	Application Resolution Time (ms)	This counter monitors application resolution time.
	Application Resolutions/sec	This counter monitors application resolutions.
	Data Store Connection Failure	This counter monitors datastore connection failure.
	DataStore bytes read/sec	This counter monitors datastore bytes reads per second.
	DataStore bytes written/sec	This counter monitors datastore bytes written per second.
	DataStore reads/sec	This counter monitors datastore reads per second.
	DataStore writes/sec	This counter monitors datastore writes per second.
	Dynamic Store bytes read/sec	This counter monitors DynamicStore bytes read per

		second.
	DynamicStore bytes written/sec	This counter monitors DynamicStore bytes written per second.
	DynamicStore reads/sec	This counter monitors DynamicStore reads per second.
	DynamicStore writes/sec	This counter monitors DynamicStore writes per second.
	Filtered Application Enumerations/sec	This counter monitors Filtered Application Enumerations per second.
	LocalHostCache bytes read/sec	This counter monitors LoadHostCache bytes read per second.
	LocalHostCache bytes written/sec	This counter monitors LoadHostCache bytes written per second.
	LocalHostCache reads/sec	This counter monitors LoadHostCache reads per second.
	LocalHostCache writes/sec	This counter monitors LoadHostCache writes per second.
	Zone Elections	This counter monitors zone elections.
	Zone Elections Won	This counter monitors zone elections won.
Memory	Page Reads/sec	This counter monitors page reads per second.
PhysicalDisk	% Disk Time	This counter monitors % disk time.
Processor	% Processor Time	This counter monitors % processor time.

QALoad-Citrix Metaframe IMA

This template includes the following counters and categories:

Category	Counters	Description
Citrix MetaFrame XP	Application Enumerations/sec	This counter monitors the application enumeration per second.
	Application Resolution Time (ms)	This counter monitors the application resolution time.
	Application Resolutions/sec	This counter monitors the application resolution.
	Data Store Connection Failure	This counter monitors the datastore connection failure.
	DataStore bytes read/sec	This counter monitors the datastore bytes read per second.
	DataStore bytes written/sec	This counter monitors the datastore bytes written per second.
	DataStore reads/sec	This counter monitors the datastore reads per second.
	DataStore writes/sec	This counter monitors the datastore writes per second.
	Filtered Application Enumerations/sec	This counter monitors filtered application enumerations per second.
	LocalHostCache bytes read/sec	This counter monitors LoadHostCache bytes read per second.
	LocalHostCache bytes written/sec	This counter monitors LoadHostCache bytes written per second.
	LocalHostCache reads/sec	This counter monitors LoadHostCache reads per second.
	LocalHostCache writes/sec	This counter monitors LoadHostCache writes per second.
Terminal Services	Active Sessions	This counter monitors active sessions.
	Total Sessions	This counter monitors total sessions.

QALoad-Citrix Metaframe Zone

This template includes the following counters and categories:

Category	Counters	Description
Citrix MetaFrame XP	Dynamic Store bytes read/sec	This counter monitors the dynamic store bytes read / sec.
	DynamicStore bytes written/sec	This counter monitors the dynamic store bytes written / sec.
	DynamicStore reads/sec	This counter monitors the dynamic store reads / sec.
	DynamicStore writes/sec	This counter monitors the dynamic store writes / sec.
	LocalHostCache bytes read/sec	This counter monitors the LocalHostCache bytes read / sec.
	LocalHostCache bytes written/sec	This counter monitors the LocalHostCache bytes written / sec.
	LocalHostCache reads/sec	This counter monitors the LocalHostCache reads / sec.
	Zone Elections	This counter monitors the zone elections.
	Zone Elections Won	This counter monitors the zone elections won.
Network Interface	Bytes Total/sec	This counter monitors network connection total bytes.
	Current Bandwidth	This counter monitors network connection current bandwidth.
Terminal Services	Active Sessions	This counter monitors active sessions.
	Total Sessions	This counter monitors total sessions.

QALoad-Cold Fusion

This template includes the following counters and categories:

Category	Counters	Description
ColdFusion MX Server	Avg DB Time (msec)	

	Avg Queue Time (msec)	
	Avg Req Time (msec)	
	Bytes In / Sec	
	Bytes Out / Sec	
	DB Hits / Sec	
	Page Hits / Sec	
	Queued Requests	
	Running Requests	
	Timed Out Requests	
Memory	% Committed Bytes In Use	
	Available Bytes	
	Page Faults/sec	
Process	% Processor Time	

QALoad-Generic Application Availability and Performance

This template includes the following counters and categories:

Category	Counters	Description
Process	% Processor Time	This counter returns the percentage of elapsed time that all threads of a process use the processor to execute instructions. This process could include code executed to handle certain hardware interrupts or trap conditions.
System	System Up Time	This counter monitors critical tasks by verifying the existence of processes. You can monitor single or multiple tasks running on the system by selecting the Processes tab from the task manager and

		then selecting processes that you want to monitor. You can also monitor only certain instances of a task by specifying a Process ID to monitor. If you do not specify a Process ID, this counter monitors all instances of the task.

QALoad-MS IIS Availability

This template includes the following counters and categories:

Category	Counters	Description
System	System Up Time	
Web Service	Current Anonymous Users	
	Current Connections	
	Logon Attempts/sec	
	NonAnonymous Users/sec	
	Not Found Errors/sec	
	Total Delete Requests	
	Total Files Sent	
	Total Get Requests	
	Total NonAnonymous Users	
	Total Not Found Errors	

QALoad-MS IIS Performance

This template includes the following counters and categories:

Category	Counters	Description
Internet Information Services Global	Current Blocked Async I/O Requests	

	Total Blocked Async I/O Requests	
	Total Rejected Async I/O Requests	
	URI Cache Flushes	
	URI Cache Hits	
	URI Cache Hits %	
	URI Cache Misses	
PhysicalDisk	% Disk Time	
Process	% Processor Time	
Redirector	Current Commands	
	Network Errors/sec	
Server	Work Item Shortages	
Server Work Queues	Queue Length	
Web Service	Not Found Errors/sec	

QALoad-Server Health

This template includes the following counters and categories:

Category	Counters	Description
Memory	% Committed Bytes In Use	
	Pages/sec	
PhysicalDisk	% Disk Time	
	Avg. Disk Queue Length	
Processor	% Processor Time	
System	Processor Queue Length	

QALoad-Windows Availability

This template monitors the availability of the Windows operating system, focusing on:

Logons

Security

Up time

The default sampling interval for this template is 5 minutes.

This template includes the following counters and categories:

Category	Counters	Description
Server	Errors Access Permissions Errors Granted Access Errors Logon Errors System Logon Total	The Microsoft Windows Availability template uses these Server registry counters to monitor errors due to logon problems. To enable these counters, you must configure your Windows system to audit logon and logoff events. You can do this by configuring the Audit Policy in the User Manager for Domains program.
	Server Sessions Sessions Errored Out Sessions Forced Off Sessions Logged Off Sessions Timed Out	The Microsoft Windows Availability template uses these Server registry counters to monitor how well users' sessions are running. If there is a large number of session errors, it is usually due to systems rebooting often or network errors.
System	System Up Time	This counter returns the number of seconds that a system was available for use. If this number continues to reset to zero, it means that the system is rebooting often. For a report that lists the number of times that the system has rebooted over a period of time, see the Microsoft Windows Availability Report topic.

QALoad-Windows Performance

This template monitors the performance of the Microsoft Windows system, focusing on:

CPU

Disk I/O

Disk space

Memory

Network

The default sampling interval for this template is 5 minutes.

This template includes the following counters and categories:

Category	Counters	Description
LogicalDisk	% Disk Time	This counter monitors the percentage of elapsed time that the disk services read and write requests, including the time that the disk driver waits in the disk queue. If this value is consistently near 100%, the disk is in very heavy use. You can determine which processes are making the majority of the disk requests by monitoring them individually.
	% Free Space	This counter monitors low free-space situations.
	Avg. Disk Queue Length	This counter indicates the number of pending I/O service requests. If the returned value is greater than 2, there is a disk problem. On a multi-disk subsystem, such as a striped set or striped with parity, you can perform a calculation to determine the presence of a disk problem. The basic formula is (Disk Queue Length) - (Number of Physical Disk Drives in the multi-disk configuration). For example, if you have a striped set with 3 disk drives and a queue length of 5, then you get an acceptable value of 2 (5 - 3 = 2).
Memory	Available Bytes	If the value returned by this counter falls under 10 MB,

		virtual memory is running low. To resolve this, close some applications or increase the memory settings. If this counter is consistently low after an application is running, it usually indicates a system memory leak.
		As the value returned by this counter decreases, the value returned by the Committed Bytes counter increases. This indicates that a process is allocating memory from the virtual address space but might not be using it. Because the virtual address space is a limited resource, use these counters to check for applications that allocate memory but do not use it. To resolve this, add more physical memory. When an application finishes processing, note the last value. If this counter does not return to the original value, the application has a memory leak or a hidden process that has not properly terminated.
		The acceptable range for committed bytes should be less than the physical RAM. The default value is 64 MB.
	Cache Faults/sec	If the value returned by this counter is less than the value returned by the Page faults/sec counter, the system is paging too much for a normal system. To resolve this, add more physical memory.
	Committed Bytes	This counter returns the amount of virtual memory (in bytes) that was committed, as opposed to memory that has was reserved.
	Page Faults/sec	If the value returned by this counter is greater than 5, the system is paging too much. Add more physical memory. A consistent value of 10 or later needs immediate attention.

	Page Reads/sec	
	Pages/sec	If this counter returns a high peak value, the system is experiencing a lot of paging activity. A high value also indicates that your system does not contain enough physical memory to handle the demands placed on it by the application. To resolve this, add more physical memory. To calculate the % disk time used for paging, use the following calculation: (% Disk Time used for paging) = (Memory, Pages/sec) * (Average Disk Transfer/sec) * 100
	Transition Faults/sec	
Paging File	% Usage Peak	This counter returns the maximum use of your page file. If the value the counter returns consistently reaches 90%, the virtual address space is too small. You should increase the size of your paging file. When the value returned by the counter exceeds 75%, a significant system performance degradation becomes noticeable.
PhysicalDisk	% Disk Time	
	Avg. Disk Queue Length	
	Avg. Disk sec/Transfer	
	Disk Reads/sec	
	Disk Writes/sec	
Processor	% Interrupt Time	This counter monitors the percentage of time that the processor spent receiving and servicing hardware interrupts during the sample interval.
	% Processor Time	On single processor systems, if the value returned by this counter is consistently higher than 90%, the CPU probably has a bottleneck. You should

		examine each process in the system to determine which one is using more of the processor than it should. The process with the highest peak is generally the performance bottleneck.
	% User Time	This counter monitors non-idle processor time spent in User mode as a percentage of the sample interval.
Redirector	Network Errors/sec	This counter indicates how many serious network errors have occurred. These errors are generally logged in the system event log, so you can check there for more information. If an error occurs, take immediate action to resolve the problem.
Server	Bytes Received/sec	
	Bytes Total/sec	
	Bytes Transmitted/sec	
	Errors Logon	This counter determines if an unauthorized user is trying to access your system.
	Work Item Shortages	This counter monitors the number of times that a work item was not allocated. You might need to increase the InitWorkItems and MaxWorkItems parameters for the LanMan Server if this number continues to increase.
System	Context Switches/sec	If the value returned by this counter value is high, assign a higher priority to the use of critical sections or semaphores by the program. This achieves a higher throughput and reduces task switching.
	Processor Queue Length	

Creating a New Template

To open the New Monitoring Template wizard:

1. In Conductor, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Click **Templates>New Template**. The New Monitoring Template Wizard appears.

3. Click **Next** to start the procedure.

To create a new template:

Use the following steps in the New Monitoring Template wizard to create a new monitoring template:

> **Note**: (WebLogic and WebSphere) When QALoad detects a managed server environment, you must also select the individual server on which to model the template.

1. Enter the template properties

2. Configure the monitor

3. Server discovery (WebLogic and WebSphere)

4. Choose the server (WebLogic and WebSphere)

5. Process the server (WebLogic and WebSphere)

6. Counter discovery

7. Choose the counters

8. Choose the instances

9. Review, save, and create the template

Next: Enter the Template Properties

Opening an Existing Template

Use the following steps to apply a previously created or pre-defined template.

To open and review an existing template:

1. In the Conductor, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. In the Manage Monitoring Tasks window, click **Templates>Open Existing**. The Select a Monitor Template File dialog box displays.

3. In the **Look in** field, select a template type, then select a template and click **Open**. The template and its counters display in the Manage Monitoring Tasks window.

> **Note**: To apply a template to a task, use the New Monitoring Task wizard.

Editing Instances for Templates

Use the following steps to modify the instances to monitor in a custom template:

Step 1: Open the Edit Template Instances Wizard

Step 2: Choose the instances to monitor

Step 3: Save the template

Step 1: Open the Edit Template Instances wizard:

1. In the Conductor, click **Tools>Monitor Tasks** to display the Manage Monitoring Tasks window.

2. Open the template to edit.

3. Click **Templates>Edit instances**. The Edit Template Instances Wizard appears.

Step 2: Choose the instances of the counter to monitor:

Review the counters you selected. When a red dot appears next to a counter, you must select an instance for the counter.

1. Double-click the counter group to display the counters.

2. Select an instance for a counter and click **Edit**. The **Select instance for counter** dialog box appears.

3. To add an instance: In the Available Instance pane, select an instance and click **Add**.

4. To remove an instance: In the Selected instances pane, select an instance and click **Remove**.

5. Repeat until you select all instances of the counter that you want to apply to the task.

6. Click **Save**. You return to the **Choose Instances** dialog box.

7. Repeat this process for each designated counter.

8. Click **Next**. The Summary dialog box displays.

Step 3: Save the template:

1. On the **Summary** dialog box, review the monitors and counters you have selected for the template. Click **Back** to return to a dialog box and make changes to the information.

2. Click **Back** to return to the previous step and edit the instances.

3. Click **Finish** to create the template.

Modifying Template Counters Using New Discovery Data

When you want to add or edit counters in a custom template, you can generate the discovery data that you add to the template. The **Edit Monitoring Template** wizard guides you through the process of generating and applying new discovery data. Use the following steps to modify template counters using new discovery:

Step 1: Open the Edit Monitoring Template Wizard

Step 2: Enter properties of the template

Step 3: Configure the monitor

Step 4: Counter Discovery

Step 5: Choose the counters

Step 6: Choose the instances of the counter

Step 7: Save the template

Step 1: Open the Edit Monitoring Template wizard:

1. In the Manage Monitoring Tasks window, open the template to edit.

2. Click **Templates>Add/Edit Counter>Use new discovery data**. The **Edit Monitoring Template** wizard appears.

3. Click **Next** in the **Welcome** dialog box. The **Enter properties of the template** dialog box displays.

Note: (WebLogic and WebSphere) You can change the Java Settings field, if necessary. If the Java file location has changed, click the **Browse** button and select the new location.

Step 2: Enter properties of the template:

In the Enter properties of the template dialog box, do the following:

1. Review the template information.

2. Type or edit the description for the template in the **Description** field

3. Click **Next**. The **Configure Monitor** dialog box displays.

Note: (WebLogic and WebSphere) You can modify any field in the Configure Monitor dialog box except the Admin Server field.

Step 3: Configure the monitor:

In the Configure Monitor Dialog, do the following:

1. Type the configuration data for the host machine, if necessary. This data is used to connect to the host machine and to the host database during counter discovery and runtime data collection. The required configuration data varies depending on the monitor type selected. Click a link below to view the required configuration details for your monitor type.

   - Oracle Application Server
   - JVM
   - SAP
   - ServerVantage
   - SNMP
   - WebLogic
   - WebSphere
   - WebSphere MQ
   - Windows Registry
   - WMI

2. Click **Next**.

   - WebLogic and WebSphere) The Processing Servers dialog box displays. Follow the procedure for selecting a server. Once you select the server, the automatic counter discovery process begins.

   - (All other monitor types) The automatic counter discovery process begins.

Step 4: Counter discovery:

QALoad automatically performs the counter discovery. The default maximum time for counter discovery is 300 seconds. When counter discovery is complete, the Choose Counters dialog box displays.

Step 5: Choose the counters:

When the counter discovery process completes, the **Add the desired counter to this template** dialog box appears.

1. From the **Available Items** pane in the **Choose Counters** dialog box, select the **Template** tab or the **Counter** tab.

2. To add an item, select a template or a counter to monitor in the task for this machine and monitor type, and click **Add**, or double-click the item to display it in the **Selected Items** pane. Click **Add All** to add all the items on the selected tab to the **Selected Items** pane.

3. To remove an item, double-click the item in the **Selected Items** pane or select the item and click **Remove**. The item is returned to the **Available Items** pane.

Note: Select multiple counters and templates by doing one of the following:

- To select nonadjacent counter items, click one counter item, and then hold down **CTRL** and click each additional counter item.

- To select adjacent counter items, click the first counter item in the sequence, and then hold down **SHIFT** and click the last counter item.

4. Click **Next**. The **Choose Instances** dialog box displays.

Note: When you select a template containing counters that are not present on the machine you are defining, you receive a message with a list of the counters that will not be added to the task.

Step 6: Choose the instances of the counter to monitor:

Review the selected counters. When a red dot appears next to a counter, select an instance of the counter.

1. Double-click the counter group to display the counters.

2. Select a counter and click **Edit**. The **Select instance for counter** dialog box appears.

3. In the **Available Instance** pane, select an instance and click **Add**.

4. Repeat until you select all instances of the counter that you want to apply to the task.

5. Click **Save**. The **Choose Instances** dialog box appears.

6. Repeat this process for each designated counter.

7. Click **Next**. The **Summary** dialog box displays.

Step 7: Save the template:

1. On the **Summary** dialog box, review the counters and instances you have selected for the template. Click **Back** to return to a dialog box and make changes to the information.

2. Click **Finish** to create the template.

Modifying Template Counters Using Cached Discovery

When you need to add or edit counters in a template that you created, you can use the cached counter discovery data to modify the template.

Note: You cannot modify the counters in pre-defined templates.

Follow these steps to modify template counters using cached discovery:

Step 1: Select the counter to add or remove

Step 2: Choose the instances of the counter

Step 3: Save the template

Step 1: Select the counter to add or remove:

1. In Conductor, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Open the template to edit, then click **Templates>Add/Edit counter>Use cached discovery data**. The **Edit Template Counters** wizard appears with the **Add/Edit/Remove Template Counters** dialog box displayed.

3. From the **Available Items** pane, select the **Template** tab or the **Counter** tab.

4. To add an item, select a template or a counter to monitor for this machine and monitor type, and click **Add**, or double-click the item to display it in the **Selected Items** pane. Click **Add All** to add all the items on the selected tab to the **Selected Items** pane.

5. To remove an item, select the item in the **Selected Items** pane and click **Remove,** or double-click the item to return it to the **Available Items** pane.

Note: Select multiple counters and templates by doing one of the following:

- To select nonadjacent counter items, click one counter item, and then hold down **Ctrl** and click each additional counter item.

- To select adjacent counter items, click the first counter item in the sequence, and then hold down **Shift** and click the last counter item.

6. Click **Next**. The **Add/Edit/Remove Template Instances** dialog box displays.

Note: When you select a template that contains counters not present on the machine you are defining, a message displays with a list of the counters that will not be added.

Step 2: Choose the instances of the counter to monitor:

1. Review the selected counters. When a red dot appears next to a counter, select an instance of the counter.

2. Double-click the counter group to display the counters.

3. Select a counter and click **Edit**. The **Select instance for counter** dialog box appears.

4. In the **Available Instance** pane, select an instance and click **Add**.

5. Repeat until you select all instances of the counter that you want to apply.

6. Click **Save**. The **Choose Instances** dialog box appears.

7. Repeat this process for each designated counter.

8. Click **Next**. The **Summary** dialog box displays.

Step 3: Save the template:

1. On the **Summary** dialog box, review the selected monitors and counters for the template. Click **Back** to return to a dialog box and make changes to the information.

2. Click **Finish** to create the template.

Removing a Counter from a Template

Remove a counter from a template by following this procedure.

Using the Conductor

To remove a counter from a template:

1. In Conductor, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Select the counter or counter family to delete.

3. Click **Templates>Remove counter**.

4. When the verification dialog box displays, click **OK**.

📘 **Note:** You cannot remove the only counter family in a template or the last counter in a family.

Creating and Editing Monitoring Tasks

Creating a New Monitoring Task

To open the New Monitoring Task wizard:

1. From the Conductor Start Page, click **Configure Monitoring** in the Tasks area.

   OR

   In the Conductor's Visual Designer, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Click **File>New**. The **New Monitoring Task** wizard appears.

3. Click **Next** to start the procedure.

📘 **Note**: You can open the Manage Monitoring Tasks window to edit or create a task by clicking the browse button next to the **Monitor task** field in the Session node.

To create a new monitoring task:

Use the following steps in the **New Monitoring Task** Wizard to create a new monitoring task:

1. Define the monitor
2. Configure the monitor
3. Discover the servers (WebLogic and WebSphere)
4. Choose the servers (WebLogic and WebSphere)
5. Process the Server (WebLogic and WebSphere)
6. Discover the counters
7. Choose the counters for the monitoring task
8. Choose the instances of the counter to monitor
9. Review the monitor definition
10. Save and create the monitoring task

Next: Define the Monitor

Using an Existing Monitoring Task

To select an existing monitoring task:

1. From the Conductor Start Page, click **Configure Monitoring** in the Tasks area.

   OR

In the Conductor's Visual Designer, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Click **File>Open**. The Choose an Existing Task dialog box appears.

3. Select a task and click **OK**. The task displays in Manage Monitoring Tasks window.

4. Select **Enable runtime monitoring** at the bottom of the window to enable the monitoring task.

Note: You also can enable monitoring in the Session node of the Visual Designer. Use the drop-down arrow in the **Monitor task** field to select an existing task, then select **Enable monitoring**. You can open the Manage Monitoring Tasks window to edit or create a task by clicking the browse button next to the **Monitor task** field.

See also:

Choose an Existing Monitoring Task Dialog Box

Adding a Monitoring Machine

Use this procedure to add a monitor to an existing task.

Note: (WebLogic and WebSphere) In a managed server environment, you only can add a monitor to a task from a different administrative server.

▪ For WebLogic, you must use the same WebLogic jar files and WebLogic version as the current monitor.

▪ For WebSphere, you must use the same WebSphere Home, WebSphere client version, and WebSphere server version as the current monitor.

To add a monitor to an existing task under the same administrative server, use Edit an Existing Server Group.

To open the New Monitoring Task wizard:

1. In the Conductor, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Click **Actions>Add monitor**. The Add Monitoring Machine wizard appears. Click **Next** to start the procedure.

Note: You can open the Manage Monitoring Tasks window to edit or create a task by clicking the browse button next to the **Monitor task** field.

To add a monitoring machine:

Use the following steps in the **Add Monitoring Machine** wizard to add a monitoring machine to the task:

Note: (WebLogic and WebSphere) When QALoad detects a managed server environment, you must also select the individual servers to monitor.

1. Enter properties of the monitoring machine

2. Configure the monitor

3. Discover the Servers (WebLogic and WebSphere)

4. Choose the Servers (WebLogic and WebSphere)

5. Process the Server (WebLogic and WebSphere)

6. Discover the counters

7. Choose the counters for the monitoring task

8. Choose the instances of the counter to monitor

9. Review the monitor definition

10. Save and create the monitoring task

Using the Conductor

Editing a Monitoring Machine

To open the Edit Monitoring Machine wizard:

1. In Conductor, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Select a monitor in the Monitors panel, then click **Actions>Edit monitor**. The **Edit Existing Monitor** wizard appears.

3. Click **Next** to start the procedure.

Notes:

- You can open the Manage Monitoring Tasks window to edit or create a task by clicking the browse button next to the **Monitor task** field in the Session node.

(WebLogic and WebSphere) In a managed server environment, the Edit Monitor dialog box appears. Do one of the following:
- o Select Edit Server Group and click OK. The Edit Existing Monitor Group wizard appears. Use this option to add, edit, or remove monitors under the same administrative server.
- o Select Edit This Server and click OK. The Edit Existing Monitor wizard appears. Use the procedure for Editing a Single Server in a Managed Server Environment to edit the counters or instances of a single monitor in the managed server group.

To edit a monitoring machine:

Use the following steps in the **Edit Existing Monitor** wizard to change the properties of a monitoring machine:

1. Enter properties of the monitoring machine

2. Configure Monitor Dialog

3. Discover the Counters

4. Choose Counters

5. Choose Instances

6. Review Monitor Definition

7. Summary

Editing an Existing Server Group

(WebLogic and WebSphere) In a managed server environment, you can add, edit, or remove monitors managed under the same administrative server using this procedure.

To open the Edit Existing Monitor Group wizard:

1. In the Conductor, click **Tools>Monitor Tasks**. The Manage Monitoring Tasks dialog box appears.

2. Select a monitor, then click **Actions>Edit Monitor**. The Edit Monitor dialog box appears.

3. Select **Edit Server Group**, then click **OK**. The Edit Existing Monitor Group wizard appears.

4. Click **Next**.

> **Note**: You can open the Manage Monitoring Tasks window to edit or create a task by clicking the browse button next to the **Monitor task** field.

To add, edit, or remove a monitor:

Use the following steps in the Edit Existing Monitor Group Wizard:

1. Enter Properties of the Monitoring Machine
2. Configure Monitor Dialog
3. Discover Servers
4. Choose Servers
5. Process Server
6. Discover Counters
7. Choose Counters
8. Choose Instances
9. Review Monitor Definition
10. Summary

Editing a Single Server in a Managed Server Environment

(WebLogic and WebSphere) Use this procedure to edit the counters and instances for a single server in a managed server group.

To open the Edit Monitoring Machine wizard:

1. In the Conductor, click **Tools>Monitor Tasks**. The Manage Monitoring Tasks dialog box appears.

2. Select a monitor, then click **Actions>Edit Monitor**. The Edit Monitor dialog box appears.

3. Select **Edit This Server**, then click **OK**. The Edit Existing Monitor wizard appears.

4. Click **Next**.

> **Note**: You can open the Manage Monitoring Tasks window to edit or create a task by clicking the browse button next to the **Monitor task** field.

To edit the server:

Use the following steps in the **Edit Existing Monitor** wizard:

1. Discover the counters
2. Choose counters
3. Choose instances
4. Review Monitor Definition
5. Summary

Editing Instances

Use the following procedure to edit instances if the counters you are monitoring:

Step 1: Open the Edit Instances dialog box

Step 2: Choose the instances to monitor

Step 3: Review the monitor definition

Step 4: Save the task

Step 1: Open the Edit Instances dialog boxes:

1. In the Conductor, click **Tools>Monitor Tasks** to open the Manage Monitoring Tasks window.

2. Select the machine, the counter, or the instance to edit.

3. Click **Tools>Monitoring>Edit instances**. The Edit Instances dialog box displays.

Step 2: Edit the instances of a counter:

1. In the **Choose Instances** dialog box, double-click the counter group in the left-hand pane to display the counters.

2. Select a counter and click **Edit**. The **Select instance for counter** dialog box appears.

   Note: When a counter can not be edited, the Edit button is unavailable.

3. Perform the necessary edits. You can do the following:

   - In the Available Instances pane, select an instance and click **Add**. The instance is added to the Selected Instances pane. Repeat until you select all instances of the counter that you want to apply to the task.

   - In the Selected Instances pane, select an instance and click **Remove**. The instance is removed from the Selected Instances pane and added to the Available Instances pane.

4. Click **Save**. The **Choose Instances** dialog box displays again.

5. Repeat this process for each counter you want to edit.

6. Click **Next**. The **Review Monitor Definition** dialog box displays.

Step 3: Review the monitor definition:

1. Review the information for the monitoring machine you defined.

2. Select one of the following:

   Set up another monitor for this task - returns to the Define Monitor dialog box so you can add another monitor to the monitoring task.

   Continue without adding any more monitors - continues in this dialog box.

3. (Optional) Click **Save as Template** to create a template for this monitoring task.

4. (Optional) Select a monitor in the Monitors pane and click **Remove Monitor** to delete a monitor from the task.

5. (Optional) Type a new value in the **Sample Interval** field. This is the frequency, in seconds, at which QALoad requests data from ServerVantage during runtime data collection.

6. Click **Next**. The Summary dialog box displays.

Step 4: Save the task:

1. Review the monitors and counters you have selected for the task. Click Back to return to a dialog box and make changes to the information.

2. In the **Monitoring task name** field, type a name for the monitoring task.

3. In the **Description** field, type a description for the task.

4. Select a monitor in the **Monitors** pane, and click **View Monitor Details**. The Properties of dialog box displays with detailed information about the monitor configuration and the counters you selected.

5. Click **Finish** to create the monitoring task.

6. Select **Enable runtime monitoring** at the bottom of the window to enable the monitoring task.

Note: You also can enable monitoring in the Session node of the Visual Designer. Use the drop-down arrow in the **Monitor task** field to select an existing task, then select **Enable monitoring**. You can open the Manage Monitoring Tasks window to edit or create a task by clicking the browse button next to the **Monitor task** field.

Removing a Monitor or a Counter from a Monitoring Task

Remove a monitor or a counter from a monitoring task, by following this procedure.

To remove a counter from a monitoring task:

1. In the **Monitors** pane of the Manage Monitoring Tasks window, select the monitor, counter, or counter family to delete.

2. Click **Actions>Remove Monitor/Counter**.

3. When the verification dialog box displays, click **OK**.

Note: You cannot remove the last monitor on a machine, the last counter in the family, or the last family of counters in the task.

Monitoring Managed Server Environments

Monitoring Managed Server Environments

WebLogic and WebSphere servers can be configured into managed server environments, where a multi-server group is managed by an administrative server. You can create tasks and templates for managed server environments using the procedures for creating and editing monitoring tasks and selecting the servers from which to extract data.

When you select a WebLogic or WebSphere server to monitor and QALoad detects a managed server environment, it automatically queries the administrative server for the individual servers it manages. All servers that QALoad discovers are listed as available servers that you can select for monitoring. You also can create a template in a managed server environment by selecting an individual server on which to model the template.

Edit the counters and instances for a single server in a managed server group using the procedures for creating and editing monitoring tasks. You also can add, edit, or remove monitors managed under the same administrative server.

Creating Monitoring Tasks for Managed Servers

Use the New Monitoring Task wizard to create monitoring tasks for individual servers or groups of servers in a managed server environment. When you create a monitoring task for a managed server environment,

you select an administrative server to monitor. *QA*Load queries the administrative server for the individual servers it manages. All servers discovered are listed as available for monitoring.

> 🔷 **Note**: When the information returned by the administrative server indicates that it does not manage any other servers, the counter discovery process begins for the individual administrative server.

Once you select the servers to monitor, *QA*Load begins the counter discovery process for each server in turn. Select the counters and instances of counters to monitor on the first server. QALoad includes these in the task, and then begins the counter discovery process for the next server you selected.

Creating Monitoring Templates for Managed Servers

You can select counters and instances and save them to a template that you can use for other monitoring tasks.

When you create a template in a managed server environment, you select a server on which to model the template and adding the counters and instances of counters to the template.

*QA*Load queries the administrative server for all the servers it manages. From this list, select the server to use as the model when creating the template. *QA*Load's New Monitoring Template wizard guides you through the process of adding the counters and instances of counters to the template.

> 🔷 **Note**: You can select only one server as a model for the template. If the server you select is unavailable, you are returned to the managed server selection dialog box to choose another server.

Editing Monitors in a Managed Server Environment

In a managed server environment, you can edit a single server or modify an existing group of servers in a task.

Edit an individual server using the Edit Existing Monitor wizard. You can add or remove counters and instances to monitor on the server. When you edit a server group, you use the Edit an Existing Server Group wizard. You can add monitors, edit the properties of a monitor, or remove monitors managed under the same administrative server. You also can change the counters and instances to monitor on an individual machine.

ServerVantage

Overview of Server Monitoring with ServerVantage

If you are currently a licensed user of Compuware ServerVantage, you can integrate data from your existing ServerVantage deployment directly into a *QA*Load timing file.

For this method to be successful, the following conditions must be met:

> ServerVantage must be installed and configured correctly on your system.

> ServerVantage must be scheduled to monitor the specified performance counters at a time that coincides with a running *QA*Load test.

> You must configure the port to use for the SQL database. The port must be open on the ServerVantage database server so that QALoad can retrieve the counter data at the conclusion of the test. The default SQL port is1433.

> QALoad must be able to access the ServerVantage database server on port 139 or 445 via tcp to obtain time stamps at the beginning and end of the test.

QALoad must be able to access the ServerVantage agent using an ICMP ping during the monitor setup. If security restrictions prevent pinging the agent, an entry can be added to the host's file on the Conductor machine mapping the domain name of the agent to the IP address of a machine that can be pinged, such as the Conductor.

About ServerVantage

ServerVantage (formerly EcoTOOLS) monitors the availability and performance of applications, databases and servers, allowing users to centrally manage events across all application components— Web servers, firewalls, application servers, file systems, databases, middleware, and operating systems. ServerVantage simultaneously monitors these components, analyzes both historical and real-time events, and correlates monitored information for problem detection.

Integration with ServerVantage is configured from the *QA*Load Conductor. Performance counters collected during a load test are included in the test's timing file and can be sorted and displayed in *QA*Load Analyze in much the same way as *QA*Load timing data. For more information about installing or configuring ServerVantage, refer to its product documentation.

Setting Up Integration with ServerVantage

Use the following steps to set up integration with Compuware ServerVantage:

Step 1: Open the New Monitoring Task Wizard

Step 2: Define and Configure the Monitor

Step 3: Review the Monitor Definition

Step 4: Review the Summary and Create the Task

Step 1: To open the New Monitoring Task wizard:

1. Click **Tools>Monitor Tasks**.

2. Click the **Set up monitoring** link, then select **Set up a new monitoring task,** then click **OK** to open the New Monitoring Task wizard. Click **Next**.

Step 2: To define and configure the monitor:

1. In the **Define Monitor** dialog box, click the arrow in the **Monitor Type** box and select **ServerVantage**.

2. In the **Control Server Database Host** field, click the down arrow and select the hostname of the machine where the ServerVantage server is located.

3. Click **Next**. The **Configure Monitor** dialog box displays.

4. In the **Username** field, type a valid user name to access the ServerVantage server, if necessary.

5. In the **Password** field, type the password that corresponds to the user name above, if necessary.

6. Select the **Override Default Database** check box to provide the ServerVantage database name. When this option is not selected, *QA*Load uses the default ServerVantage database name. If you provided a different name during the installation of ServerVantage, select this option and type the name in the **Database Name** field.

7. In the **Name** field in the Vantage Agent area, type the hostname of a machine(s) where a ServerVantage Agent is installed, and click the **Add** button to add it to your load test.

8. Click **Next** to proceed to the next step, Review Monitor Definition.

Step 3: To review the monitor definition:

1. Review the information for the monitoring machine you defined.

2. Select one of the following:

> **Set up another monitor for this task** - returns to the Define Monitor dialog box so you can add another monitor to the monitoring task.
>
> **Continue without adding any more monitors** - continues in this dialog box.

3. (Optional) In the Monitors pane, select the monitor, then click **Save as Template** to create a template for this monitoring task.

4. (Optional) In the Monitors pane, select the monitor type, then click **Remove Monitor** to delete a monitor from the task.

5. (Optional) Type a new value in the **Sample Interval** field. This is the frequency, in seconds, at which QALoad requests data during runtime data collection.

6. Click **Next** to proceed to the next step, where you review the summary and create the task.

Step 4: To review the summary and create the task:

1. Review the monitors and counters you have selected for the task in the Summary dialog box. Click **Back** to return to a dialog box and make changes to the information.

2. In the **Monitoring task name** field, type a name for the monitoring task. The task is saved so you can reuse this configuration of counters and instances.

3. In the **Description** field, type a description for the task.

4. Select a monitor in the **Monitors** pane, and click **View Monitor Details**. The Properties of dialog box displays with detailed information about the monitor.

5. Click **Finish** to create the monitoring task. The Manage Monitoring Tasks window displays.

Displaying ServerVantage Agent Data

If you set options to integrate Compuware ServerVantage resource utilization data before running a test, that data is included in the resulting timing file. It can be sorted and displayed in *QA*Load Analyze in much the same way as *QA*Load timing data. ServerVantage data provides a summary of all the Agents that ServerVantage monitored during the load test and details aggregate statistics for Agent data points including minimum, maximum, and mean data values.

When you open a timing file containing ServerVantage Agent data, *QA*Load Analyze displays test data with QALoad timing data two ways:

> ServerVantage Agent workstations are listed in the Server Monitoring group in the Workspace tree-view, under the Resource Trends (ServerVantage) branch. From the Workspace, select **Agent workstations** to create detail or graphical views of the Agent data points. Specifically, you can:

- Display Agent data point details.

- Graph Agent data point details.

> Detailed data point information is displayed in the Data window. The ServerVantage detail view includes data such as the name of the machine where you ran the ServerVantage Agent; the Agent name; and the minimum, maximum, and mean data values for the Agent.

Note: ServerVantage resource utilization data is available only if you set the ServerVantage integration options on the *QA*Load Conductor's Test Information window before executing a load test.

ApplicationVantage

Overview of ApplicationVantage

*QA*Load integrates with Compuware ApplicationVantage to help you analyze network performance during a load test. ApplicationVantage provides granular thread details that allow network managers to identify poorly performing applications. *QA*Load also provides test data that you can open in ApplicationVantage.

Before *QA*Load can collect network data during a load test, the following must be true:

>The ApplicationVantage Agent is installed on the same machine as the *QA*Load Conductor. You can install either the ApplicationVantage Agent or the ApplicationVantage Remote Agent.

>You have specified on which NIC to capture in the Manage Players/Groups dialog box in Conductor. How?

At test time when a transaction is started, the Player configured to capture ApplicationVantage data starts an ApplicationVantage trace. The trace stops when the transaction completes. When a Player is running a script that is set to run in ApplicationVantage mode, every transaction generates a new trace file. At the end of the test, these files are packaged into the test's timing file.

> **Hint**: For information about ApplicationVantage, refer to the documentation you received with your purchase of this tool.

Configuring a test to use ApplicationVantage

Integration with ApplicationVantage enables you to study network problems in detail. You can set up one or more ApplicationVantage (AV) Player machines for the load test. These AV Player machines run a QALoad script on a periodic basis while the AV Agent captures the network traffic that the script produces. The resulting AV trace files (*.opx) are sent back to the Conductor with the regular QALoad timing file for analysis after the test is complete.

To enable ApplicationVantage, you must be running ApplicationVantage 10.0 or greater. You must enable ApplicationVantage in the Properties window, and set the Network Interface Card (NIC) Name used by the machine on which the data is captured.

Enabling ApplicationVantage

You can enable or disable the ApplicationVantage for each load test on a script. To enable ApplicationVantage, you must select the option, and then set the Network Interface Card (NIC) Name.

To enable ApplicationVantage:

1. Click the script icon for the appropriate script in the Visual Designer window. The Script Properties panel appears on the right-hand side of the window.
2. In the Script Properties panel, click the **ApplicationVantage** field, then select **True**.
3. Click **Tools>Manage Players**.
4. If necessary, set the NIC Name.

Setting up Network Interface Card Name

To use the ApplicationVantage Agent to collect data for ApplicationVantage, it is necessary to specify which Network Interface Card (NIC) to capture on. This is the network information for the workstation where your ApplicationVantage Remote Agent is installed.

To set up NIC Name:

Using the Conductor

1. On the Conductor's toolbar, select **Tools>Manage Player**. The Manage Players/Groups dialog box displays with names of available Player machines listed in the **Players** area.

2. Click the Player machine that will be running the virtual user to be captured. The information for that Player machine displays in the Player Information area.

3. If necessary, click the ⊞ button next to **Application Vantage Settings** to expand the information.

4. From the drop-down list in the **NIC Name** field, select the NIC that is used by the machine.

6. Click **Save**, then click **OK.**

ClientVantage

Overview of ClientVantage

Compuware ClientVantage manages end-user application performance and availability. Problems can be diagnosed by powerful fault detection and analysis capabilities as well as resource monitoring.

ClientVantage must be installed on the same Windows workstation as the *QA*Load Conductor and the *QA*Load Player.

Vantage Analyzer

Vantage Analyzer Integration

Compuware's Vantage Analyzer is designed for easy resolution of complex application performance issues. It enables you to easily drill into specific problem transactions to determine the cause of bottlenecks in your production applications. It also enables you to find Java code and SQL statements that are consuming excessive resources. Troublesome memory leaks that are observed in your actual production servers can be quickly resolved.

If you are currently a licensed user of Vantage Analyzer, you can integrate data from your existing Vantage Analyzer deployment directly into a *QA*Load timing file.

For this method to be successful, the following conditions must be met:

The supported version of Vantage Analyzer must be installed and configured correctly on the same machine as *QA*Load Analyze. For more information about installing or configuring Vantage Analyzer, refer to its product documentation.

Time has to be synchronized between the *QA*Load Conductor machine and the Vantage Analyzer Nucleus Server machine to make testing data more meaningful. The difference of the time between the two machines is saved in a timing file.

Setting Up Integration with Vantage Analyzer

To set up integration with Compuware Vantage Analyzer:

1. Open or create a session in the Conductor. From the **Tools** menu, choose **Monitor Tasks**. The Manage Monitoring Tasks dialog box displays.

2. Click the **Set up monitoring** link and select **Set up a new monitoring task**. Click **OK**. If the Welcome to the New Monitoring Task Wizard appears, click **Next**.

3. In the Define Monitor dialog box, click the arrow in the **Monitor type** box and select **Vantage Analyzer**.

4. In the Nucleus Server Name or IP address field, type or select the machine host name or IP address of the machine where the Vantage Analyzer Nucleus server runs.

5. Click **Next**. The Configure Monitor Dialog box displays.

6. In the **Username** field, type the login user ID for the Vantage Analyzer Nucleus server machine (not the Nucleus server itself).

7. In the Password field, type the login password for the Vantage Analyzer Nucleus server machine (not the Nucleus server itself).

8. Click **Next**. The Review Monitor Definition dialog box displays.

Capturing Vantage Analyzer Metrics

Capture several levels of Compuware Vantage Analyzer metrics during a conductor performance tests.

To capture Vantage Analzyer metrics:

1. Set up capture of J2EE/ASP.Net web application for testing within Vantage Analzyer.

2. Click Retrieve Vantage Analzyer Data button. The Retrieve Vantage Analyzer Data dialog box appears.

3. Configure Vantage Analyzer monitoring tasks in the dialog box and click OK.

4. Click the Capture Mode button to configure capture mode. The Vantage Analzyer Capture Mode-Settings dialog box appears.

5. Configure capture mode and click **OK**.

6. Run the test.

Test Setup Interface

Overview of the Test Setup Interface

The Conductor provides two methods for designing a test session: the Visual Designer and the Grid View.

Visual Designer

The Conductor's main window is the Visual Designer. The Visual Designer consists of three parts that you use to create the test session:

Visual Designer window - contains a series of nodes displayed in a tree view. The session displays as the top-level node, while the scripts in the session are represented as nodes underneath the session.

Players/Groups panel - This is a dockable panel the appears on the left-hand side of the window. It displays all Players and Groups available for the session.

Properties panel - This is a dockable panel the appears on the right-hand side of the window. Select a session node, a script icon, or a Player machine in a script node to display information and set options for each one.

You can assign player machines to a script by dragging a player or a player group from the Player/Groups panel and dropping it into the script in the Visual Designer window. Review and update properties of the test elements in the Properties panel.

In addition, the Visual Designer's toolbar provides access to standard Windows functionality, such as Print and Copy, as well as quick access to Conductor setup options and to *QA*Load Analyze.
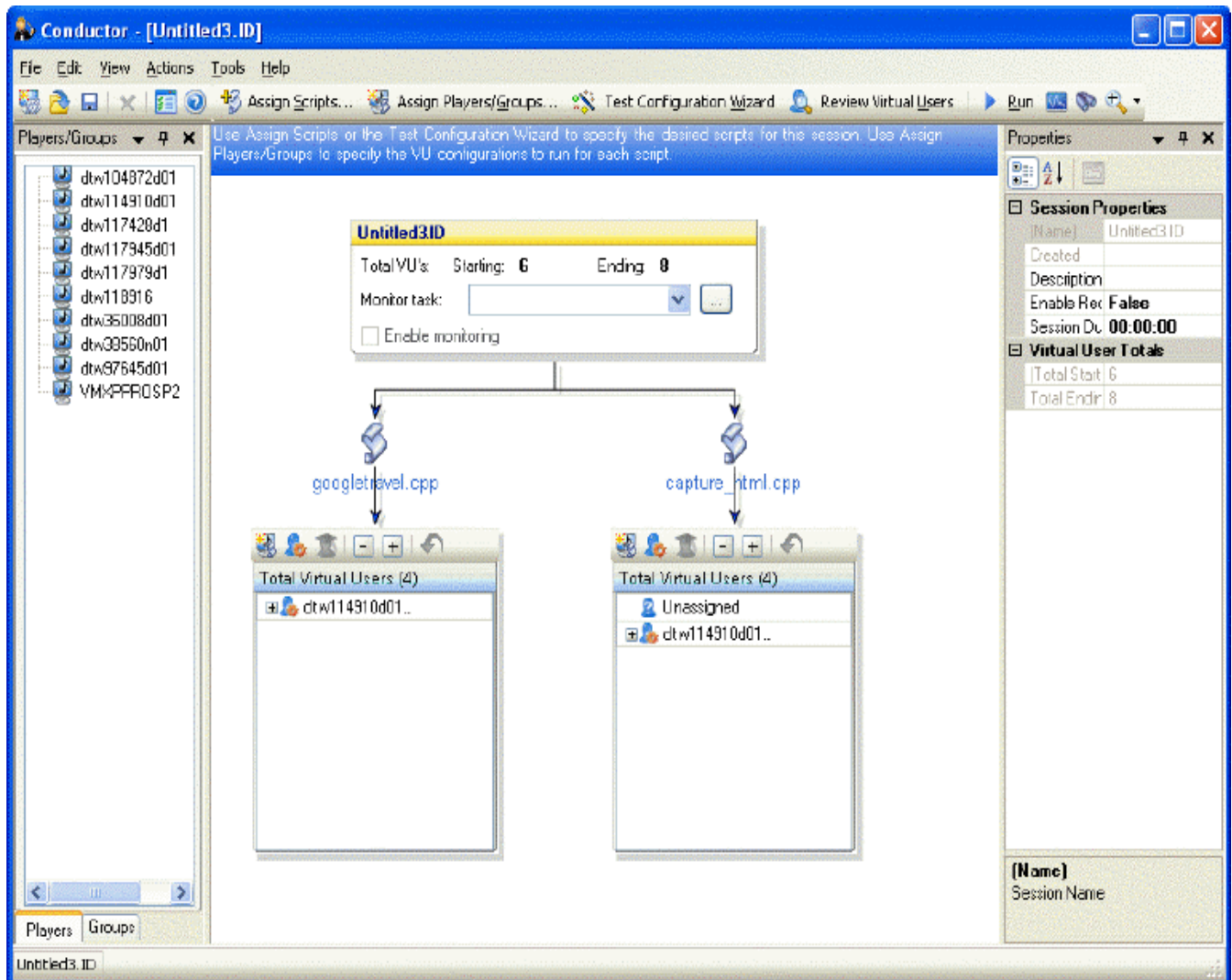
Grid View

Alternately, you can open the Grid View to enter test information and set up the machines and scripts for the test. The Grid View is a dockable window that appears at the bottom of the Visual Designer window. Changes made in the Grid View appear in the Visual Designer.

The fields you use to set options for your load test are displayed in two tabs:

Script Assignment tab - where you set up options for any scripts that have previously been recorded and compiled.

Machine Assignment tab - where you assign scripts to specific Player workstations, starting and ending virtual users (VU), VU increments, and timing intervals.

The Grid View toolbars provides access to the main functions of assigning scripts and players to your load test session.
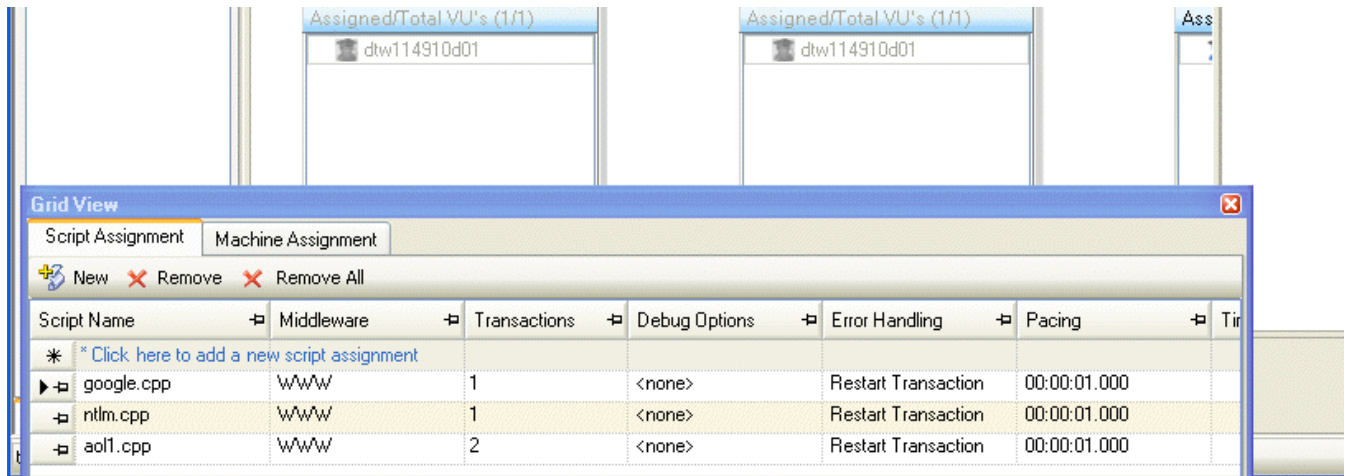
Using the Visual Designer

The Visual Designer is the main window, the test setup interface, that displays a collection of icons and nodes that represent your test session. The top level node is the current test session, and the child nodes represent the scripts assigned to the session.

Use the Visual Designer to enter information about your test and set up the machines and scripts for the test using the two panels in the Visual Designer window:

> Players/Groups panel - This is a two-tabbed window that list the installed QALoad Players and any groups in which they are members. Assign Players and groups to individual scripts by dragging and dropping them into the script nodes.
>
> Properties panel - This is a dynamic panel on the right-hand side of the Visual Designer that changes content depending on the view you choose. Use the Properties panel to setup and review configurations for session, script, and Player properties.
>
> o Session properties - Click the top-level Session node to display session properties in the Session Properties panel. Use the Session Properties panel to enable recording and set session duration.
>
> o Script properties - Click the script icon for any individual script to display its properties in the Script Properties panel. Use the Script Properties panel to set options for Application Vantage, external files and datapools, and additional script properties, such as debug options, error handling, number of transactions, and timing options.
>
> o Player properties - Click an individual Player in a script node to view it's properties in the Player Properties panel. You can set starting and ending virtual users (VU), VU increments, mode and timing intervals, and set expert user options.

Use the toolbar buttons to assign scripts, assign players and groups, or to open the Test Configuration Wizard, where you can easily configure your test session.

Using the Grid View

You can use the Grid View to enter test information about your test and set up the machines and scripts for the test. The Grid View contains two tabs:

Script Assignment tab - Use this tab to set up options for any scripts that have previously been recorded and compiled. Any scripts you add here are included in your load test, and one virtual user is automatically assigned to your script on the **Machine Assignment** tab. After setting up your scripts here, you must assign additional virtual users to your script from the **Machine Assignment** tab.

Machine Assignment tab - Use the **Machine Assignment** tab in the Grid View to assign scripts to specific Player workstations. You also assign starting and ending virtual users (VU), VU increments, and timing intervals.

To open the Grid View:

Click **View>Grid Window**. The Grid View pane appears below the Visual Designer window.

Runtime Window

When you start a test, the Conductor's interface changes to an interactive test control station called the Runtime Window. From the Runtime Window, you can observe the progress of individual scripts and Player machines, create real-time graphs, and change the behavior of scripts and Players from a running test to better simulate the unpredictability of real users. This window has two unique areas:

Runtime main window - The information in the main Runtime window depends on the view you select in the Active View field. You can view details for all test scripts, individual test scripts, all player machines, and individual player machines.

Runtime Options Panel - The lower pane, called the Runtime Options Panel, displays data for the current view and the individual script you select. This is a dockable control station that allows you to change virtual user options and data transfer options while your test is running. The information in the Runtime Options panel is displayed in three tabs: Virtual User Options Tab, Script Options Tab, Global Options Tab.

Running a Test

Running a Load Test

After validating a script, it is safe to run a load test with that script.

To start a load test:

In the Conductor, click the **Run** button on the configuration and setup toolbar, or from the **Actions** menu, choose **Run**. While a test is running, the Conductor's Interface changes to provide you with real-time test options. For more information, see Runtime Window Interface.

> **Note**: While any window on the desktop is re-sizing or re-positioning, all Windows applications pause. Do not click and hold on a window caption or border for extended periods during a load test because it delays message handling and may have an impact on test results.

While a load test is running, the Conductor's toolbar changes from the Configuration and Setup Toolbar to the Runtime Toolbar. The Runtime Toolbar buttons let you control the test and access detailed information about the test while it is running. For more information, see Monitoring a Load Test. This gives detailed information about what to expect from the *QA*Load Conductor while a test is running — including descriptions of the Runtime Toolbar buttons

Running a Series of Tests

You can also run a series of tests — a batch test. A batch test comprises multiple session ID files. When you run a batch test, the session files are executed sequentially until all of them are executed. The Conductor enables you to run multiple batch tests without operator intervention. For more information, see Running a Batch Test.

Running a Load Test

After validating a script, it is safe to run a load test with that script.

To start a load test:

In the Conductor, click the **Run** button on the configuration and setup toolbar, or from the **Actions** menu, choose **Run**. While a test is running, the Conductor's Interface changes to provide you with real-time test options. For more information, see Runtime Window Interface.

> **Note**: While any window on the desktop is re-sizing or re-positioning, all Windows applications pause. Do not click and hold on a window caption or border for extended periods during a load test because it delays message handling and may have an impact on test results.

While a load test is running, the Conductor's toolbar changes from the Configuration and Setup Toolbar to the Runtime Toolbar. The Runtime Toolbar buttons let you control the test and access detailed information about the test while it is running. For more information, see Monitoring a Load Test. This gives detailed information about what to expect from the *QA*Load Conductor while a test is running — including descriptions of the Runtime Toolbar buttons

Running a Series of Tests

You can also run a series of tests — a batch test. A batch test comprises multiple session ID files. When you run a batch test, the session files are executed sequentially until all of them are executed. The Conductor enables you to run multiple batch tests without operator intervention. For more information, see Running a Batch Test.

Dialing Up/Down Virtual Users

*QA*Load's dial-up/dial-down feature in Conductor allows you to dynamically add or reduce virtual users to your test at the script or Player level while your test is running. This enables you to adjust your running test according to test behavior on-the-fly, rather than stopping to re-configure playback criteria.

To use the dial-up/dial-down feature, you must:

> be licensed for at least the number of virtual users requested
>
> configure a dial-up/dial-down session before running the test

Notes:

- ▪ If you have not configured a dial-up/dial-down session, you will not be allowed to add or suspend virtual users while the test is running. For more information, see Configuring a Dial-up Session.

- ▪ Dial-up/dial-down is enabled only after all Virtual Users configured for the test session are ramped up.

- ▪ Dial-up/dial-down is not supported for a machine assignment entry that is using a player group.

When your test is running, the bottom of the Test Information window turns into the dockable Runtime Options panel, a portion of which is shown below:
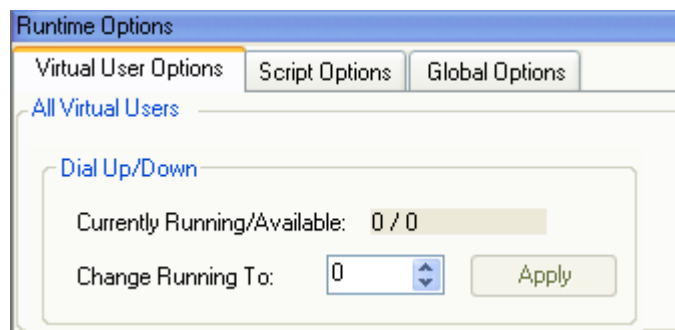
When you click on a Player or script in the test's tree-view, the Runtime Options panel indicates how many virtual users are currently running on the selected Player machine or script. You can change the number of running virtual users per script or per Player by selecting the appropriate script or Player machine in the tree-view, and then typing a new number in the Change Running To field (or by using the up/down arrows).

To dial up or down (add or subtract) virtual users during a test:

1. When your test is running, click on the script or Player workstation in the Runtime Window's tree-view for which you want to add or subtract virtual users. On the Virtual User Options tab, the **Currently Running/Available** field shows how many virtual users are currently running on that script or Player.

2. In the **Change Running To** field, type a new number or use the up/down arrows to change the number.

3. When you are done, click **Apply**. The Conductor will release or suspend the specified number of virtual users.

Notes:

> Your changes do not take effect until you click **Apply**.
> When you dial down a virtual user, the virtual user finishes the current transaction before going into a suspended state.

Increase/Decrease Runtime Timing Updates

While a test is running, you can change the frequency at which timing updates are sent from the Players to the Conductor in the Runtime window. Decreasing the update interval reduces the amount of overhead incurred in large load tests due to the communications between the Conductor and large numbers of virtual users.

To change the Runtime Timing updates:

1. On the **Global Options** tab of the Runtime Options panel (bottom pane), choose from the following options:

   - **No Updates**: Choose this option to stop sending timing data while the test is running. Data will still be collected at the end of the test.

   - **Send All**: Choose this option to send all timing data as it is compiled.

   - **Periodic Updates**: Choose this option to specify a time interval for sending updates, then type the time interval (in seconds) below.

2. Click **Apply**. Any change takes effect immediately, and applies to all scripts in the test.

Stopping a Load Test

A load test is complete when all virtual users exit. A virtual user automatically exits when one of the following occurs:

> A script encounters an EXIT command.
>
> A script completes its transaction loop.
>
> A *QA*Load function fails and Abort on Error is set in Error Handling

To stop a load test:

Click the **Exit All Virtual Users** button or click the **Quit Current Test** button. The Virtual User icon changes to ❌ and the message, "Session aborted by User", displays.

Adding Post-test Comments

If you selected the **Display post test comments dialog** option when you configured the Conductor, the **Post Test Comments** window opens when you click the **Quit Current Test** button. Type any comments, which are saved to the test's Summary Report that you can view in *QA*Load Analyze.

Adding Post-test Comments

By setting the appropriate options when you configure the Conductor, you can add comments to a completed test. The comments appear in the test's Summary Report in *QA*Load Analyze.

To configure the Conductor for adding post-test comments:

1. Select **Tools>Options**. The Options dialog box appears.

2. In the **Dialog** section of the Conductor Sessions page , select **Display post test comments dialog**.

3. Click **OK**. The Conductor is now configured so that you can add comments when a test completes.

To add post-test comments:

1. In the Test Completed dialog box, click **Exit Test**.

   OR

   In the Runtime toolbar, click the **Quit current test** button. The Post Test Comments dialog box displays.

2. Type any comments in the dialog box, then click **OK**. Your comments are saved in the **Post Test Comments** field of the Summary Report in

3. *QA*Load Analyze.

Monitoring a Load Test

When you start a test, the *QA*Load Conductor's interface changes to an interactive test control station, referred to as the Runtime Window. The Runtime Window displays information about the scripts, machines, and virtual users that are executing the load test. From the Runtime Window, you can observe the progress of individual scripts and Player machines, create and view real-time graphs, and start or suspend scripts and Players from a running test to better simulate the unpredictability of real users. For more information, see Runtime Window Interface.

In addition to the test data shown by default on the Runtime Window, you can access detailed test information using the *QA*Load Conductor's Runtime Toolbar Buttons. You can:

    View statistics for a single virtual user

    View the activities of a virtual user in a browser-like window (WWW only)

    Step to the next request (WWW only)

    View the current datapool record

    Display the script running on a single virtual user

    Display messages sent from a Player workstation to the *QA*Load Conductor

    Display statistics about Conductor/Player communication

    Show/hide the Runtime Tree or Runtime Control Panel

    Exit, abort, or quit the test

For more information, see Runtime Toolbar Buttons.

Viewing Test Statistics

Once all workstations stop executing, click the **Quit Current Test** button on the toolbar to complete the test and automatically create the timing file (.tim). To compute and view test statistics from the timing file in QALoad Analyze, open Analyze from Conductor.

To access Analyze from the Conductor:

From the Conductor's **Tools** menu, choose **Analyze**.

Conductor Runtime Interface

Runtime Window Interface

When you start a test, the Conductor's interface changes to an interactive test control station called the Runtime Window. The Runtime Window displays information about the scripts, machines, and virtual users that are executing the load test. The Runtime window is divided into two areas – the main Runtime window, and the dockable Runtime Options panel at the bottom of the window.

On the Runtime Window, you can observe the progress of individual scripts and Player machines, view real-time graphs, and start or suspend scripts and Players from a running test to better simulate the unpredictability of real users. The Runtime main window dynamically displays test details according to the type of information you select in the **Active View** field.

The lower pane, the Runtime Options panel, displays data for the current view and the individual script you select. This is a dockable control station that allows you to change virtual user options and data transfer options while your test is running.

Runtime Window

When you start a test, the Conductor's interface changes to an interactive test control station called the Runtime Window. From the Runtime Window, you can observe the progress of individual scripts and Player machines, create real-time graphs, and change the behavior of scripts and Players from a running test to better simulate the unpredictability of real users. This window has two unique areas:

Runtime main window - The information in the main Runtime window depends on the view you select in the Active View field. You can view details for all test scripts, individual test scripts, all player machines, and individual player machines.

Runtime Options Panel - The lower pane, called the Runtime Options Panel, displays data for the current view and the individual script you select. This is a dockable control station that allows you to change virtual user options and data transfer options while your test is running. The information in the Runtime Options panel is displayed in three tabs: Virtual User Options Tab, Script Options Tab, Global Options Tab.

Active View Details

Details for a running load test are displayed in the Runtime main window. The details displayed depend on the option you select in the **Active View** field. You can choose:

All Virtual Users

Virtual Users by Script

Virtual Users by Machine

All Scripts

All Player Machines

Graphs View

Sessions View

VantageAnalyzer MReports View

All Virtual Users

Lists the script and player machine involved in the test and other detail information by each virtual user. Double-click a virtual user to display the Virtual User Info window. Select this option to view details about each virtual user running a script in the test:

Status icon: The first column displays an icon that indicates the status of the virtual user. A moving icon represents a running virtual user; a still icon represents a virtual user that hasn't yet started or is suspended; and an icon with a checkmark through it represents a virtual user that has exited. A red circle with an X indicates errors have occurred on that virtual user, or that the test session was manually terminated using the Exit, Abort, or Quit Current Test buttons.

User: Virtual User's identification number assigned by QALoad.

Script: Name of the script the virtual user is running.

Machine: Name of the Player machine on which the script is running.

Pass: Number of transactions successfully completed.

Fail: Number of transaction that have failed to complete successfully.

Min: Lists the shortest response time recorded for a transaction on the virtual user.

Max: Lists the longest response time recorded for a transaction on the virtual user.

Last: Lists the most recent response time recorded on the virtual user.

Status: Lists a status for the virtual user if any errors have been encountered, or lists the name of the checkpoint the user has encountered if the option Send all timing data including Checkpoint information was selected on the Runtime window of the Session Options dialog box during test setup. Double-click a virtual user for more information, or if the error message is to long to read.

Errors:

Error Message:

> ⬛ **Note**: Double-click in any field to display the Virtual User Info dialog box with the details about each virtual user.

back to top

Virtual Users by Script

Lists Virtual User and Player machine information within each assigned script. Double-click a virtual user to display the Virtual User Info window. Select this option to view the following details for each virtual user sorted by script:

User: Virtual User's identification number assigned by QALoad.

Machine: Name of the Player machine on which the script is running.

Pass: Number of transactions successfully completed.

Fail: Number of transaction that have failed to complete successfully.

Min: Lists the shortest response time recorded for a transaction on the virtual user.

Max: Lists the longest response time recorded for a transaction on the virtual user.

Last: Lists the most recent response time recorded on the virtual user.

Status: Lists a status for the virtual user if any errors have been encountered, or lists the name of the checkpoint the user has encountered if the option Send all timing data including Checkpoint information was selected on the Runtime window of the Session Options dialog box during test setup. Double-click a virtual user for more information, or if the error message is to long to read.

Errors:

Error Message:

> ⬛ **Note**: Double-click in any field to display the Virtual User Info dialog box with the details about each virtual user.

back to top

Virtual Users by Machine

Lists Virtual User and script information within each assigned player machine. Double-click a virtual user to display the Virtual User Info window. Select this option to view the following details for each virtual user sorted by Player machine:

User: Virtual User's identification number assigned by QALoad.

Script: Name of the script the virtual user is running.

Pass: Number of transactions successfully completed.

Fail: Number of transaction that have failed to complete successfully.

Min: Lists the shortest response time recorded for a transaction on the virtual user.

Max: Lists the longest response time recorded for a transaction on the virtual user.

Last: Lists the most recent response time recorded on the virtual user.

Status: Lists a status for the virtual user if any errors have been encountered, or lists the name of the checkpoint the user has encountered if the option Send all timing data including Checkpoint information was selected on the Runtime window of the Session Options dialog box during test setup. Double-click a virtual user for more information, or if the error message is to long to read.

Using the Conductor

Errors:

Error Message:

> ⚑ **Note**: Double-click in any field to display the Virtual User Info dialog box with the details about each virtual user.

back to top

All Scripts

Lists detail information for each script assigned to the test. Select this option to display the following summary details about every script in the test:

Script: The name of the script.

Total Users: The total number of virtual users assigned to run the script.

Running: The total number of virtual users currently running the script. This number may vary at different times in a single test if you are configured for dial-up virtual users, or have configured the test as a ramp-up test.

Pass: The number of transactions successfully completed.

Fail: The number of transactions that have failed to complete successfully.

Response Time: The average response time of all virtual users currently running the test script.

Throughput: The average number of transactions per second this script is running.

back to top

All Player machines

Lists detail information for each player machine assigned to the test. Select this option to display the following summary information about each Player machine running the test.

Machine: The machine name, preceded by an icon indicating whether the test is currently running on the machine or finished. A blue checkmark indicates a successfully completed test.

Total Users: The total number of virtual users controlled by that Player Agent.

Running: The total number of virtual users currently running on that Player Agent machine.

% Processor: The percentage of the Player Agent machine's processor that is currently in use.

% Memory: The percentage of the Player Agent machine's memory that is currently in use.

% Disk: The percentage of the Player Agent machine's disk space that is currently in use.

Status: Lists a general status for the test. For example, *Test is running*.

back to top

Graphs View

Displays real-time graphs for checkpoints, performance counters, and Player machine health statistics. You can control which types of data are graphed in addition to how the graphs appear. For detailed information, refer to Graphs View.

Sessions View

Displays summary information about the test and the Player machine. For more information, refer to Session View

VantageAnalyzer MReports View

Displays the latest retrieved VantageAnalyzer reports. If there are no VantageAnalyzer monitoring task associated with the tests, the active view is not available. For more information, refer to VantageAnalyzer MReports View.

Session View

When you select **Session** in the Active View field, the Conductor Runtime Window provides summary information about the test session that is currently running. The Session view can be printed as a report by right-clicking and choosing **Print** from the shortcut menu.

> **Note**: The Session view below has been cropped to better fit this help topic, while still representing what a real Session view might look like.

Click on the sections in the following graphic for more information about the Session view.

Graphs View

The Graphs view in the Conductor Runtime Window displays graphs of data collected during the test. By default, the Graphs view displays graphs for response times, test status, and player machine health.

Other graphs, such as user-defined checkpoints and Remote Monitoring counters, can also be plotted in the right pane of the Graphs view if they were enabled for the session.

To display graphs:

1. Right-click on a counter or other data type in the tree view that you want to plot in a graph.

2. Choose **Add Graph** or **Add Plot To**.

You can also modify a graph's appearance by right-clicking on the graph and choosing one of the formatting options, such as colors and axes properties. To increase the visibility of a plot when you have multiple plots on a graph, click on a plot (or that plot's number in the legend) to highlight it.
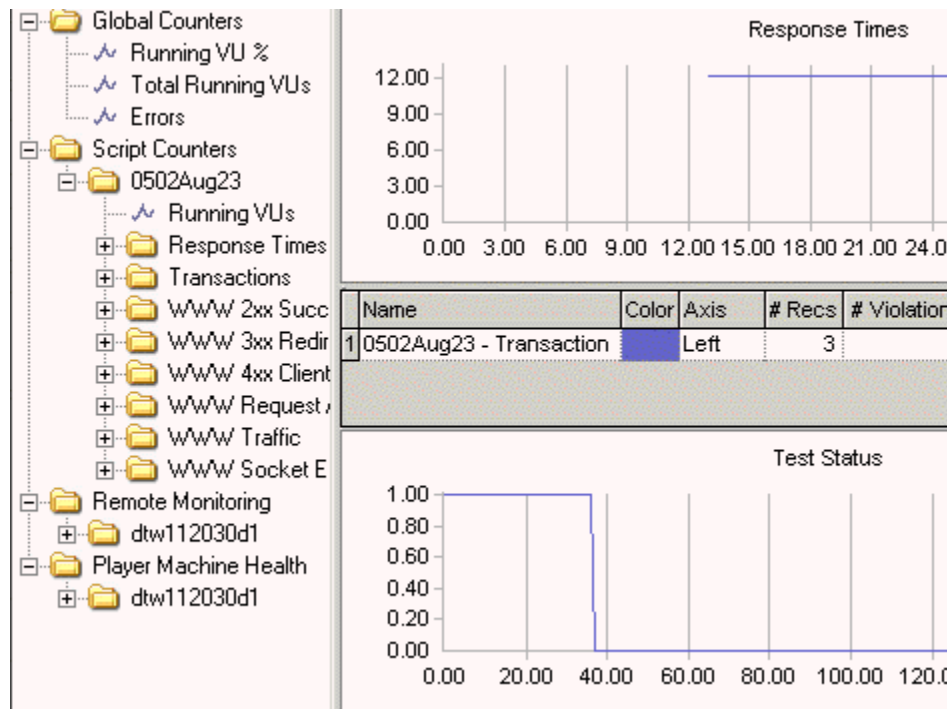
VantageAnalyzer MReports View

When selecting VantageAnalyzer MReports from the **Active View** field, the Conductor Runtime Window provides the last retrieved VantageAnalyzer MReport. The three toolbar buttons available in this view allow you to display the VantageAnalyzer Configuration dialog box, to refresh the VantageAnalzyer report, and to launch the VantageAnalyzer Performance Console.

This view is not available if the VantageAnalyzer reports retrieval is not configured. To set up VantageAnalyzer reports retrieval, refer to Retrieve VantageAnalyzer Data.

Runtime Options Panel

This dockable control station enables you to change virtual user options and data transfer options while your test is running. The information in the Runtime Options panel is displayed in three tabs:

Virtual User Options Tab

Currently Running/Available: Displays the number of currently running Virtual Users.

 Change Running To: Click the arrows to change the number of Virtual Users in the test.

Apply: Click to apply your changes.

Script Options Tab

Changes made in this window will override options set during the original Script Assignment.

> **Note**: Options on this tab are available only when **Virtual Users by Script** or **All Scripts** are selected in the Active View.

Error Handling: Choose how to respond when an error occurs during execution of the transaction. During large load tests, errors can sometimes indicate that the test is straining the limits of the hardware/software in the testing environment. Options are:

> **Abort Transaction** — If an error occurs while a transaction is being executed, the Player should abort the current transaction and the virtual user who encountered the error should exit the test. Use this option when errors will make the virtual user invalid for executing more transactions.

> **Continue Transaction** — If an error occurs while a transaction is being executed, the Player should continue executing the transaction as if the error didn't occur. Select this option when errors are not critical to the performance of the load test and can be safely ignored.

> **Restart Transaction (WWW, SAPGUI, and Citrix scripts only)** — If an error occurs while a transaction is being executed, the Player should abort the current transaction entirely and restart a new transaction from the beginning. Note that the transaction count will increase for each transaction that is restarted.

Pacing: Enter a value in this field to change the rate of pacing. Pacing is the time interval between the start of a transaction and the beginning of the next transaction on each workstation running the script. For example: if a transaction is designed to duplicate the process of someone handling incoming telephone calls and those calls arrive at a rate of 40 per hour/per person, set the pacing rate at 90 seconds.

Sleep: *QA*Load records the actual delays between requests and inserts the DO_SLEEP command in the script to mimic those delays when the script is played back in a test. You can maintain the exact length of the recorded delays at playback, or shorten them by entering a smaller percentage of the originally recorded delay to play back. For example, if you recorded a delay of 10 seconds then DO_SLEEP (10); is written to your script. Then, if a Sleep Factor of 50% is specified here, the Player will sleep for 5 seconds at that statement when the test is executed.

Apply: Click to apply your changes to the running script.

Cancel: Click to cancel any changes you have not yet applied to the running script.

Global Options Tab

Global Options apply to all scripts.

Timing Updates: Select when the Players in your test should send timing information to the Conductor. You can choose:

> **No Updates** - No timing updates are sent to the Conductor

> **Send All** - Sends all timing updates to the Conductor

> **Periodic Updates** - If you chose Periodic Updates, type how often, in seconds, timing updates should be sent to the Conductor in the **(1 - 1000 Sec.)** field.

Apply: Click to apply your changes to the running script.

Cancel: Click to cancel any changes you have not yet applied to the running script.

Monitoring a Running Test

Monitoring a Load Test

When you start a test, the *QA*Load Conductor's interface changes to an interactive test control station, referred to as the Runtime Window. The Runtime Window displays information about the scripts, machines, and virtual users that are executing the load test. From the Runtime Window, you can observe the progress of individual scripts and Player machines, create and view real-time graphs, and start or suspend scripts and Players from a running test to better simulate the unpredictability of real users. For more information, see Runtime Window Interface.

In addition to the test data shown by default on the Runtime Window, you can access detailed test information using the *QA*Load Conductor's Runtime Toolbar Buttons. You can:

> View statistics for a single virtual user
>
> View the activities of a virtual user in a browser-like window (WWW only)
>
> Step to the next request (WWW only)
>
> View the current datapool record
>
> Display the script running on a single virtual user
>
> Display messages sent from a Player workstation to the *QA*Load Conductor
>
> Display statistics about Conductor/Player communication
>
> Show/hide the Runtime Tree or Runtime Control Panel
>
> Exit, abort, or quit the test

For more information, see Runtime Toolbar Buttons.

Monitoring CPU Usage

To help you monitor the impact of running a load test on a server, *QA*Load can collect data from selected Players about CPU usage during a load test. The statistics collected during the test are merged into the test's timing file so you can view them in *QA*Load Analyze after the test.

> **Note**: During a load test, if the CPU idle time of your machine falls below 25%, check the individual processes on your machine. If the Players and virtual users are utilizing most of the active CPU time, you should use additional Player machines and fewer virtual users per Player to conduct your load test.

Watching a Script Execute

Use the Debug tab in the Conductor Runtime window to view the executing script. Note that it is possible that you will not see the execution of every statement. In order to minimize network traffic between the Conductor and the Players, the Player sends its script debug status to the Conductor once per second, so that the Player can execute several statements without sending a debug message to the Conductor.

To open the Debug tab:

Select a Player in the Runtime window, then click the **Debug Virtual User** toolbar button.

> ⓘ **Note**: The Conductor highlights the script line that it is currently executing.

Graphing Checkpoints in Conductor

Use the Graphs view of the runtime Conductor to create real-time graphs of checkpoint response times during script execution.

> ⓘ **Note**: Similar graphs are also available for post-test analysis in *QA*Load Analyze.

Checkpoints are listed in the tree view on the left side of the Graphs view of the runtime Conductor, as shown in the example below. Both automatic and user-defined checkpoints appear in the **Response Times** folder of each running script.
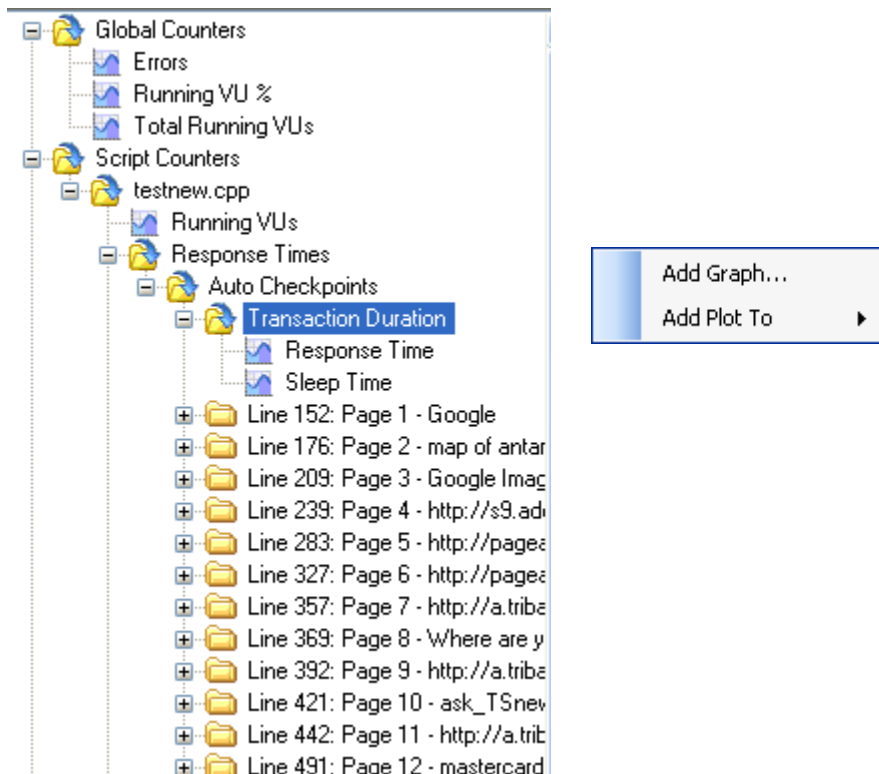
Creating a Graph of Checkpoint Response Times

Before you can review checkpoint response times in graph form, you must select the checkpoint counters to include. To choose a checkpoint that should appear in a graph, highlight the checkpoint name, right-click and choose either **Add Graph** to create a new graph or **Add Plot To** to add a data plot to an existing graph.

If you choose the **Add Graph** option, the Add Graph dialog box appears. Select the options for how the graph should appear and click **OK**.

Adding Thresholds

To better identify problem checkpoints, you can set thresholds on plots or graphs that indicate the number of times the data record for that checkpoint has gone above or below the number you set. Thresholds can be set from the Advanced tab of the Add Graph dialog box or by right-clicking on an existing graph and choosing **Thresholds**.

Highlighting Individual Plots

If you create several plots on a single graph, it may become difficult to see individual plots. To increase a plot's visibility, click on a plot in the graph or a plot's number in the graph's legend. When highlighted, the plot appears thicker and darker on the graph.

Saving Checkpoint Graphs to a Session ID

Checkpoint graphs that are created in the Conductor are automatically saved to the current session ID file. To remove all graphs you added, click **Graph>Restore Default Graph Layout**.

Graphing Remote Monitoring

Use the Graphs view of the runtime Conductor to create real-time graphs of counter data during script execution. Similar graphs are also available for post-test analysis in *QA*Load Analyze.

Selecting Counters to /*Graph

All counter data that is available for graphing is located in the tree view on the left side of the of the Graphs view Data window, as shown below.
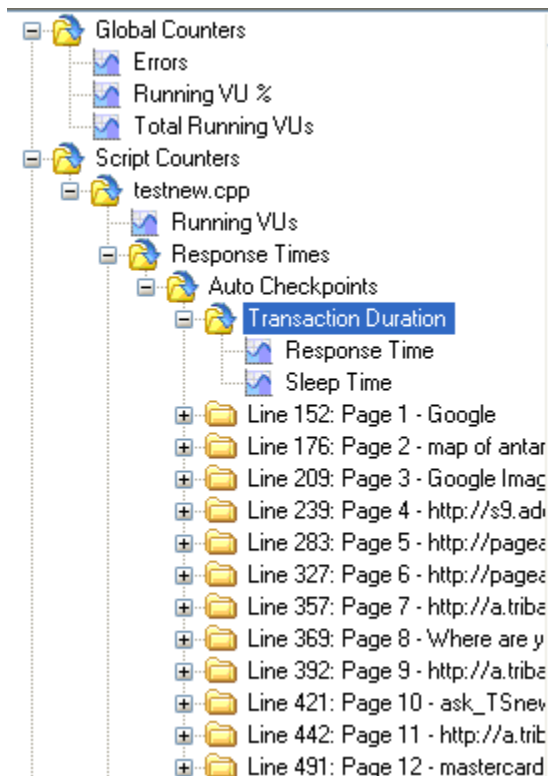
Scripts of any middleware type collect the following default counter data, which is available in the Conductor for real-time graphing:

   Global counters: Running VU%, total running VUs, and errors

   Script counters: Running VUs, response times, and transactions

   Player machine health: % processor, % memory used, % disk space, %disk time, % paging file

Additional middleware-based graphs are also generated by default and vary by middleware. For example, for the WWW middleware, several performance-based counters are automatically collected and available for graphing, including server responses and WWW traffic. You can monitor this data to determine the optimum rate of performance of the application that is running.

Graphing Counter Statistics

To choose a counter that should appear in a graph, highlight the checkpoint counter name or group of counters (folder), right-click and choose either **Add Graph** to create a new graph or **Add Plot To** to add a data plot to an existing graph.

If you choose the **Add Graph** option, the Add Graph dialog box appears. Select the options for how the graph should appear and click **OK**.

To better identify problems in the test, you can set thresholds on plots or graphs that indicate the number of times the data record for that counter has gone above or below the number you set. Thresholds can be set from the Advanced tab of the Add Graph dialog box or by right-clicking on an existing graph and choosing **Thresholds**.

Highlighting Individual Plots

If you create several plots on a single graph, it may become difficult to see individual plots. To increase a plot's visibility, click on a plot in the graph or a plot's number in the graph's legend. When highlighted, the plot appears thicker and darker on the graph.

Saving Counter Data Graphs to a Session ID

Counter data graphs that are created in the Conductor are automatically saved to the current session ID file. To remove all graphs you added, click **Graph>Restore Default Graph Layout**.

Running a Series of Tests (Batch)

Running a Batch Test

By setting the appropriate options in the Conductor, you can elect to run a series of tests as a batch, rather than one at a time. A batch test comprises multiple session ID files that are executed sequentially.

You can create a batch test by adding a number of session ID files to a batch file. Before you can add a session ID to a batch file, the following conditions must be true:

> The session must include a defined number of transactions. Sessions of unlimited transactions cannot be used in a batch test.

> All scripts to be included must exist before starting the batch test.

To run a batch test:

1. Select **Actions>Batch Test**. The Configure Batch Test dialog box appears.

2. Select the required session ID files in the **Available Session Files** list and click **Add** to add them to the **Selected Sessions** list.

3. If you want to run a previously defined batch, click the **Load Batch File** button in the toolbar to navigate to the directory where the batch file (.run) resides. Select it, and click **OK**.

4. In the **Delay Between Tests** field, click the up or down arrow to set the number of seconds to wait before starting the next test.

5. Click **Save** to save the current batch file, or click **Save As...** to save the batch file under a new name.

6. Click **OK** to return to the main Visual Designer window

   OR

   Click **Start** to begin running the batch test.

The Conductor then executes each of the session ID files in sequence.

Adding Sessions to a Batch Test

Before a session is added, the following conditions must be true:

> The session must include a defined number of transactions. Sessions of unlimited transactions cannot be used in a batch test.

> All scripts must exist prior to starting the batch test. This means that the files referenced in the selected session ID files are present in the script directory.

A session can be placed in a batch multiple times. This feature might be used to re-run a test or to perform housekeeping chores, such as logging users in or out of a host or database.

To add a session:

1. From the **Actions** menu, choose **Batch Test**. The Configure Batch Test dialog box displays.

2. In the **Available Session Files** box, highlight the session you want to add, and click the **Add** ⮕ Add button.

   If you want to run a previously defined batch, click the **Load** button to navigate to the directory where the batch file (.run) resides. Select it, and click **OK**.

   The session is added to the Selected Sessions list on the right side of the dialog box.

3. In the **Delay Between Tests** field, click the up or down arrow to set the number of seconds to wait before starting the next test.

4. Click **Save** to save the current batch file, or click **Save As...** to save the batch file under a new name.

5. Click **Start** to begin running the batch test, or click **OK** to return to the main Visual Designer window.

Setting Delays Between Tests

You can set a fixed delay or pause between tests by specifying a value in the **Delay Between Tests** field on the Configure Batch Test dialog box. After each test is complete, the Conductor delays for the specified amount of time before starting the next test. To set up a series of tests, see Running a Batch Test.

Removing a Session from a Batch Test

To remove a session from a batch test:

1. Select **Actions>Batch Test**. The Configure Batch Test dialog box appears.

2. Click **Load Batch File** and select the file you want to modify.

3. In the **Selected Sessions** pane, highlight the session to remove and click **Remove**.

4.   Click **OK**.

Terminating a Batch Test

Stop a batch of tests the same way you would stop a single session test, by clicking the **Abort All Virtual Users** or **Exit All Virtual Users** on the toolbar. The Virtual User icon changes to ⊗ and the message, "Session aborted by User", displays. When the Conductor process stops for any reason during a load test, the associated Player processes automatically terminate.

Troubleshooting

Conductor pre-test checks

Before a test begins, the Conductor completes the following pre-test checks of the parameter files and Players. If any of these checks fail, the Conductor displays an error message.

Are there enough Players configured to support the number of users specified in the session ID file?

Does the number of users specified in the session ID file exceed the maximum number of users defined by your authorization key?

Can the specified compiled script files be accessed?

Are all Players communicating with the Conductor? (The Conductor sends a request message to all the Players to verify that they are up and running.)

Executing SSL scripts that use client certificates

If you are executing SSL scripts that use client certificates, you must manually copy the client certificates in use to the Player machine(s) executing the script(s).

Manually copy the client certificates from the `\Program Files\Micro Focus\QALoad\Certificates` directory to the same default directory on the Player machine.

> **Note**: On the Unix player platform, you must create the `Certificates` sub-directory in the `QALoad` directory. The directory name is case sensitive.

See also:

Create a Client Certificate in QALoad

Heartbeat message failure on a virtual user

When a Player machine crashes or experiences a loss of communication, the heartbeat message that the Conductor sends out (if enabled) fails. This situation is indicated in the runtime Conductor through a message on each virtual user that is affected. When the heartbeat message fails for a virtual user, the Status column of the Details view of a script displays the following message: "The Player running this user failed to respond to a heartbeat message."

The option for enabling a heartbeat message is located on the Player page of the Options dialog box in the Conductor.

Timing file is too big

Depending on the length of the load test and the amount of data that was collected, timing files can grow to excessively large sizes that become difficult to handle. To prevent timing files from becoming too large, try modifying the following settings:

Disable automatic middleware checkpoint timings in the Conductor

Use the Conductor's timing data thinning options

Both of these settings are located on the Timing page of the Script Properties dialog box.

To access the Script Properties dialog box:

1. In the Visual Designer window, click the script icon   to display the script Properties panel on the right-hand side of the window.

2. In the Script Properties section, click the **Timing Options** field to display the **browse [...]** button.

3. Click the **browse [...]** button.

Tips for running QALoad tests on UNIX systems

To successfully run large *QA*Load tests on UNIX systems, you may need to make adjustments to your settings as described below:

General (AIX, Solaris, and RedHat Linux)

When you attempt to run a large number of virtual users on UNIX platforms, the virtual users do not always synch. If virtual users do not synch, try increasing the Virtual User Startup Delay. By default, *QA*Load Conductor sets the VU Startup Delay to 1 millisecond. This default is not high enough for UNIX platforms. If the UNIX Player receives a value less than 15 milliseconds, the delay will be 15 milliseconds or more.

To increase the delay:

1. In the *QA*Load Conductor, click **Tools>Options**.

2. Click **Player** in the tree view to display the Player page.

3. In the **VU Startup Delay** field, type the number of milliseconds to delay virtual user startup.

Solaris

The default file descriptor limit on Solaris has a "soft" limit of 64, and a "hard" limit of 1024 (Solaris 2.6). Per the Solaris 2 FAQ (refer to http://www.wins.uva.nl/pub/solaris/solaris2.html), the file descriptor limit is described in the getrlimit() manual page as: "One more than the maximum value that the system may assign to a newly created descriptor. This limit constrains the number of file descriptors that a process may create."

To increase this limit, system administrators can modify the /etc/system file and reboot the system. For example:

```
* set hard limit on file descriptors

set rlim_fd_max - 4096

*set soft limit on file descriptors

set rlim_fd_cur = 1024
```

Index